Computing Education in African Countries: A Literature Review and Contextualised Learning Materials

Sally Hamouda*
Virginia Tech
Blacksburg, VA, USA
shamouda@vt.edu

Linda Marshall*
University of Pretoria
Pretoria, South Africa
lmarshall@cs.up.ac.za

Kate Sanders*
Rhode Island College
Providence, RI, USA
KSanders@ric.edu

Ethel Tshukudu*
San Jose State University &
University of Botswana
San Jose, CA, USA
ethel.tshukudu@sjsu.edu

Oluwatoyin Adelakun-Adeyemo
Bingham University
Karu, Nigeria
toyin@sure-impact.com

Brett A. Becker
University College, Dublin,
Dublin, Ireland
brett.becker@ucd.ie

Emma R. Dodoo
University of Michigan
Ann Arbor, MI, USA
edodoo@umich.edu

G. Ayorkor Korsah
Ashesi University
Berekuso, E/R, Ghana
akorsah@ashesi.edu.gh

Sandani Luvhengo
University of Pretoria
Pretoria, South Africa
luvhengos@gmail.com

Oluwakemi Ola
University of British Columbia
Vancouver, BC, Canada
kemiola@cs.ubc.ca

Jack Parkinson
University of Glasgow
Glasgow, Scotland, United Kingdom
jack.parkinson@glasgow.ac.uk

Ismaila Temitayo Sanusi
University of Eastern Finland
Joensuu, Finland
ismaila.sanusi@uef.fi

Abstract

This report begins with a literature review of computing education in Africa. We found a substantial body of work, scattered over more than 80 venues, which we have brought together here for the first time. Several important themes emerge in this dataset, including the need to contextualise computing education.

In the second part of this report we investigate contextualisation further. We present a pilot study, grounded in the literature review, of the development of course materials, sample code, and programming assignments for introductory programming, contextualised for six African countries: Botswana, Egypt, Ghana, Nigeria, South Africa, and Zambia. We include the materials, report on a preliminary evaluation of the materials by fellow educators in African countries, and suggest a process by which other educators could develop materials for their local contexts.

CCS Concepts

• **Social and professional topics** → **Computing education**; **Computer science education**; **CS1**; **Cultural characteristics**; **Geographic characteristics**; **Race and ethnicity**; *Student assessment*; *Computing literacy*; • **General and reference** → **Surveys and overviews**; • **Human-centered computing** → **Contextual design**; **Ethnographic studies**.

Keywords

Africa; African; Computer Science Education; Computing Education; contextualisation; CS1; CS1 materials; introductory programming; literature review; Algeria; Angola; Botswana; Burundi; Cameroon; Egypt; Ethiopia; Ghana; Kenya; Lesotho; Liberia; Libya; Mauritius; Morocco; Mozambique; Namibia; Nigeria; Rwanda; Senegal; South Africa; Sudan; Tanzania; Uganda; Zambia

ACM Reference Format:
Sally Hamouda, Linda Marshall, Kate Sanders, Ethel Tshukudu, Oluwatoyin Adelakun-Adeyemo, Brett A. Becker, Emma R. Dodoo, G. Ayorkor Korsah, Sandani Luvhengo, Oluwakemi Ola, Jack Parkinson, and Ismaila Temitayo Sanusi. 2024. Computing Education in African Countries: A Literature Review and Contextualised Learning Materials. In *2024 Working Group Reports on Innovation and Technology in Computer Science Education (ITiCSE-WGR 2024), July 8–10, 2024, Milan, Italy.* ACM, New York, NY, USA, 33 pages. https://doi.org/10.1145/3689187.3709606

*Working Group Leaders

ITiCSE-WGR 2024, Milan, Italy

ACM ISBN 979-8-4007-1208-1/24/07
https://doi.org/10.1145/3689187.3709606

1 Introduction

This report begins with a literature review of computing education research in Africa. One might easily get the impression from attending computing education conferences that little or no computing education research has been done in African countries. For example, recent analyses have shown that contributions from Africa to the ACM SIGCSE Technical Symposium [113] and the ACM ITiCSE Working Groups [126] are rare. We wanted to find out if there was a computing education research literature from African countries, and if so, what it was like.

We found no previous comprehensive literature review. A bibliometric analysis of computing education research in the Global South [9] concluded that South Africa has a notable number of computing education publications. A recent study by Sanusi and Deriba [155] has some preliminary results but was limited to major international computing education research outlets. An analysis of the publications in the *South African Computer Journal* through 2009 [102] included only a few about computing education. Finally, individual papers' related work sections sometimes contain useful information, but they are, by definition, focused on a particular topic (for example, [183]).

As a result, we have investigated the following research questions:

- RQ1 How many computing education research papers based on data from one or more African countries have been published?
- RQ2 In which years were the papers published?
- RQ3 In which countries were the papers' data gathered?
- RQ4 In which venues (i.e., conferences and journals) were the papers published?
- RQ5 What were the major topics addressed in the papers?

As we will demonstrate, our findings reveal a substantial body of work distributed across more than 80 different venues. We do not attempt to synthesise the papers in our dataset into a larger statement about "what Africa thinks" about the topics they address. Africa includes many very diverse countries, most of which themselves include a variety of regions, languages, and cultures. Our argument, instead, is that these papers should be taken into account as part of the broader computing education research discourse.

To make it easier for readers to explore this literature, a complete list of the papers in our dataset is given in the references to this report (indicated by an asterisk before the first author's name). The answers to our research questions provide an outline of the dataset (Section 3), and selected papers are discussed in more detail (Section 4). By integrating African experiences, challenges, and innovations into the broader discourse of computer science education, these works contribute to a more inclusive and globally relevant body of knowledge.

In the course of this literature review, efforts to contextualise computing education emerged as an important theme. Building on this finding, we developed new contextualised materials for computing education in African countries. The process of developing these materials, the materials themselves, and a preliminary evaluation are described in the second part of this report. In Sections 6 through 13, we present our pilot study in the development of new contextualised materials for computing education in Africa: sample code and programming assignments for introductory programming. Finally, in Section 14, we conclude with a summary of the whole report and suggestions for future work.

This report extends previous work in several ways. It presents:

- A dataset of computing education papers that are based on data from African countries.
- The country affiliations of the papers' authors, the venues and years in which the papers were published, and the major topics the papers address.
- Supplementing this descriptive information, more detailed discussions of selected papers from the dataset.
- Contextualised programming activity examples for six different African countries (Egypt, Botswana, Ghana, Nigeria, South Africa, and Zambia), designed by educators from those countries and explicitly linked to the CS2023 knowledge units and learning objectives.
- A preliminary evaluation of those materials by fellow educators in those countries.
- A practical starting point and a suggested process by which lecturers across African countries, and beyond, could develop additional materials for their own local context. This structured approach to contextualisation lays the groundwork for ongoing research and continuous improvement in contextualised computing education, supporting broader goals of inclusion and diversity in the field.

1.1 Researcher Positionality

We are a team of twelve educators and researchers with a combined 115 years of computing education research experience. In addition, seven of us have 76 years of teaching experience in Africa, and, beyond the classroom, ten of us have collectively spent 297 years living in various African countries, namely, Botswana, Egypt, Ghana, Nigeria, South Africa, and Zambia. Our collective experience guides this research.

2 Literature Review: Methods

In this section, we discuss the query we used, the databases we searched, which papers we included out of those retrieved, and how we coded the papers in the resulting dataset. (Note: Our dataset includes conference papers, journal articles, and one PhD thesis; for readability, we refer to them all as "papers"). This process is summarised in Figure 1.

2.1 The Query

Our query was adapted from the query reported in Ezugwu et al.'s survey of literature related to machine learning in Africa [73]. The original query included a list of the countries in Africa, plus terms specifically related to machine learning. We modified the query to focus on computing education, rather than machine learning, and streamlined the list of countries. For example, since a search on "Sudan" should retrieve both "Sudan" and "South Sudan", we decided to search on "Sudan" alone. Similarly, we searched on "Guinea" alone, rather than "Equatorial Guinea", "Guinea", "Guinea Bissau". (This last change, we later discovered, led to the inclusion of two papers about "Papua New Guinea", but those were easy to exclude.)

We then spent the first two weeks exploring the databases to investigate what query would be most effective. After exploration, we agreed on the following logic for our query:

> ("computing education" OR "computer science education" OR "CS education" OR "robotics education" OR "machine learning education" OR "software engineering education" OR "programming education") AND ("Algeria" OR "Angola" OR "Benin" OR "Botswana" OR "Burkina Faso" OR "Burundi" OR "Cameroon"

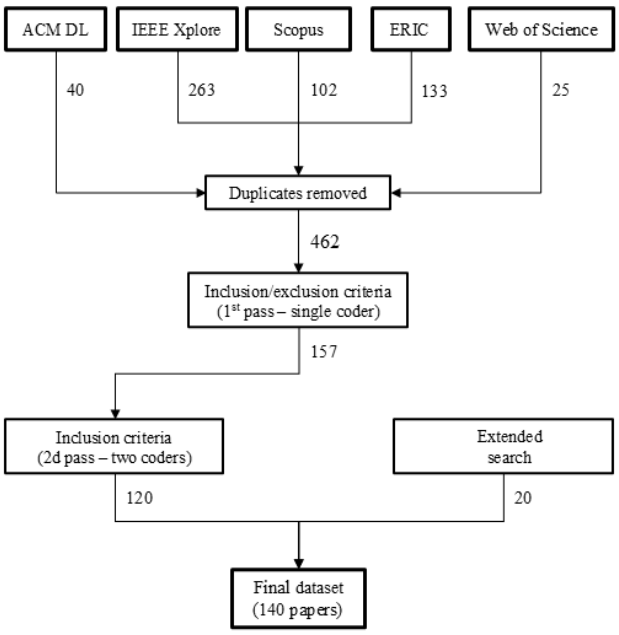

Figure 1: A dataflow diagram showing the number of papers in our dataset at each step in the analysis.

> OR "Cape Verde" OR "Central African Republic" OR "Chad" OR "Comoros" OR "Congo" OR "Cote d'Ivoire" OR "Djibouti" OR "Egypt" OR "Eritrea" OR "Eswatini" OR "Ethiopia" OR "Gabon" OR "Gambia" OR "Ghana" OR "Guinea" OR "Kenya" OR "Lesotho" OR "Liberia" OR "Libya" OR "Madagascar" OR "Malawi" OR "Mali" OR "Mauritania" OR "Mauritius" OR "Morocco" OR "Mozambique" OR "Namibia" OR "Niger" OR "Nigeria" OR "Rwanda" OR "Sao Tome" OR "Senegal" OR "Seychelles" OR "Sierra Leone" OR "Somalia" OR "South Africa" OR "Sudan" OR "Tanzania" OR "Togo" OR "Tunisia" OR "Uganda" OR "Zambia" OR "Zimbabwe")

We also decided that because full-text searches retrieved thousands of results, we would limit our search to the titles, abstracts, and keywords for each paper.

2.2 Databases

We searched the ACM Digital Library ("the DL"), IEEE Xplore, Scopus, ERIC (eric.ed.gov, an online library of education research and information), and Web of Science. The total number of papers retrieved per database can be seen in Figure 1. After removal of duplicates 462 papers remained in the dataset.

Because we were working with more than one database, the first challenge was how to implement our logical query in different databases. For the benefit of readers who might want to do a similarly broad search on this or other topics, we include details of the challenges raised by working with such a variety of different databases.

2.2.1 The ACM Digital Library. The DL's interface provides no obvious support for entering an SQL-type query such as the one given above. Instead, the primary "Advanced Search" interface of the DL offers one or more textboxes. In each box, one or more words or phrases can be entered; these are combined implicitly with "OR", and the textboxes are connected to each other by "AND". Next to each textbox, there is a menu that allows the user to select the breadth of the search from several options, including Title, Abstract, Keyword – but not more than one option per textbox. Unlike Scopus, IEEE Xplore, or ERIC, the DL does not offer the option of searching title, abstract *and* keywords.

The DL does provide a way to limit a search by modifying individual terms in the query. So we revised our query using separate terms for title, abstract, and keyword. The new search looked like this:

> (Abstract:"computing education" OR Title:"computing education" OR Keyword:"computing education" OR ...) AND (Abstract:"Algeria" OR Title:"Algeria" OR Keyword: "Algeria" OR ...)

substituting three terms (with the prefixes "Abstract:", "Title:", and "Keyword:") for each of the terms in our original query. The resulting query had a total of 183 terms.

The next question was where to enter this new expanded query. In the end, we had the best results by pasting our query into a single textbox, and selecting the menu item "Search Anywhere" for that textbox, hoping that the Title:, Abstract:, and Keyword: restrictions in the actual query would over-ride the full-text search. This did indeed seem to be the case since we retrieved a combined total of 40 papers from searching both the ACM Full-Text Collection and the ACM Guide to Computing Literature.

2.2.2 IEEE Xplore. IEEE XPlore does allow a search on abstract/title/ keywords, but limits the number of search terms per search clause to a maximum of 25. So three separate searches were performed:

(1) "All Metadata":"computational thinking" OR "All Metadata":"programming education" OR "All Metadata":"computing education" OR "All Metadata":"computer science education" OR "All Metadata":"CS education" OR "All Metadata":"robotics education" OR "All Metadata":"machine learning education" OR "All Metadata":"software engineering education") AND ("All Metadata":"Algeria" OR "All Metadata":"Angola" OR "All Metadata":"Benin" OR "All Metadata":"Botswana" OR "All Metadata":"Burkina Faso" OR "All Metadata":"Burundi" OR "All Metadata":"Cameroon" OR "All Metadata":"Cape Verde" OR "All Metadata":"Central African Republic" OR "All Metadata":"Chad" OR "All Metadata":"Comoros" OR "All Metadata":"Congo" OR "All Metadata":"Cote d'Ivoire" OR "All Metadata":"Djibouti" OR "All Metadata":"Egypt". ***Note:*** *"All Metadata" includes: abstract, index terms, and bibliographic citation data (such as document title, publication title, author, etc.).*[1]
(2) Same as above but countries are Eritrea through Mauritius, inclusive

[1] ieeexplore.ieee.org/Xplorehelp/searching-ieee-xplore/command-search#summary-of-data-fields

(3) Same as above but countries are Morocco through Zimbabwe, inclusive

2.2.3 Scopus. The search in Scopus was refreshingly simple: the first search page displayed in Scopus has a search box with "abstract/title/keywords" selected (the default scope for the search). The search box took our entire query with no modifications.

2.2.4 ERIC. By default, ERIC searches title, author, source, abstract and descriptor. Judging by the output of our searches, "source" refers to the name of the venue (e.g., *Computer Science Education* or EDUCON), and "descriptor" refers to keywords. Thus, ERIC's search is equivalent to what we did in the other databases – with the minor addition of the venue title – and a lot easier than the ACM Digital Library's.

There is an ERIC keyword "location:" that seems to be required for all papers, so we used an unrestricted search for the topics and a "location:" search for the countries:

> ("computing education" OR ...) AND (location:"Algeria" OR ...)

In other words, we did a title-abstract-keywords search for the topics, and a "location:" search for the countries. The only catch was that ERIC citations are exported in a file format called "nbib"; we wrote a program to convert the nbib entries to bibtex.

2.2.5 Web of Science. We used the "Topic" search box, which automatically searches title, abstract, keyword plus, and author keywords, without modification to our query.

2.3 Analysis: Overview

In the first phase of our analysis, before the working group met in person at ITiCSE, we applied the criteria described below in Sections 2.4 and 2.5 and developed a tentative list of paper themes or topics. Five researchers each read approximately 90 papers. While there was some discussion among the researchers, due to the size of the dataset, there was only one coder for each paper.

In the second phase, after ITiCSE, each of the papers from the initial dataset was re-analysed by two people, with one researcher coding all of the papers (for continuity) and eight researchers each coding approximately 20 papers, independently from the first coder. For each paper, we re-checked the inclusion criteria, removing papers that we agreed did not satisfy them. We then deductively coded the remaining papers using the topics list from the first pass, while remaining open to new possible codes. All differences were resolved through discussion.

In parallel, two other researchers extended our original search by identifying the authors whose names occurred on the most papers in our dataset, searching for their works on Google Scholar and ResearchGate, and coding the resulting papers. ResearchGate's recommendation feature was useful in finding "similar papers" once one was downloaded. After applying the exclusion criteria (Section 2.4), they followed the same process as the researchers re-analysing the preliminary dataset (refer to Figure 1).

More details of the analysis are given in the remainder of this section.

2.4 Analysis: Exclusion Rules

We began with some objective exclusion rules, removing from the dataset:

- Any paper with fewer than 4 pages, in order to focus on more substantial articles and conference papers, rather than posters or short work-in-progress papers.
- Citations to entire conference proceedings. These were excluded on the grounds that relevant items would be retrieved as individual papers. We applied a similar rule to "collections": books made up of chapters written by different authors on a variety of topics.
- Any papers where the full text was not available in English. There is potentially a very interesting literature available in Arabic and French, and perhaps additional languages, but it is outside the scope of this report.
- Any papers that were not peer-reviewed.

2.5 Analysis: Inclusion Rules

Our two inclusion criteria are derived from our first research question:

- Is the paper about computing education?
- Is the paper based on data gathered in one or more African countries?

2.5.1 Computing Education. The core examples of computing education research were easy to recognise: for example, papers about how to teach introductory programming. In general, we included papers about the teaching and learning of one or more computing concepts or topics (see CS2023 [4]), regardless of whether they were taught to majors or non-majors. We also included papers about teamwork, if grounded in the context of teaching and learning computing, and issues such as student motivation to study computing and gender ratios in computing classes. Similarly, the decision *not* to include papers about topics such as biodiversity and electrical engineering was an easy one.

Borderline cases that we decided were out of scope for this report included papers about advising or mentoring computing majors and the job market for computing graduates. We also decided not to include papers about statistics, mathematics and writing for computing majors. While those topics are undoubtedly important for computing students, this report focuses specifically on computing topics.

Other borderline cases were sometimes in scope and sometimes not: papers about using technology in teaching and papers about infrastructure. These were included only if they made an explicit connection to the teaching and learning of computing. The decision regarding teaching with technology is supported by the scope of ACM *Transactions on Computing Education* which states [3]:

> The journal does not publish on learning technologies unless the technology is specifically about teaching and/or learning computing concepts and reveals insights about teaching/and or learning computing. It is not enough for learners to be using computers to learn, or for learning to be set in a learning context

about computing concepts. The work must reveal insights about teaching and/or learning computing in particular to be in scope.

2.5.2 Data Gathered in One or More African Countries. For this report, all papers were required to have some data from one or more African countries. Papers based on data entirely from outside of Africa were out of the scope of our report, even if all the researchers were affiliated with an institution in an African country.

We defined "data" very broadly. Besides human participant data such as interviews with or survey data from students or instructors, samples of student work, student grades, student demographics, student journals, and classroom observations, we included experience reports, curriculum documents, education policy documents, and overviews of the past or current state of some aspect of computing education.

Two cases where we did not include the papers were:

(1) Papers where the data from an African country were aggregated with data from other countries, with no separate analysis or discussion.
(2) "Approach-design-roadmap" papers (so called from typical first words of their titles). These papers present an idea about how to do something that could generally be applied anywhere (for example, a new programming language) and has not been evaluated in an African context.

Even though some of these papers have an author who is affiliated with an institution in an African country – and some of the papers are quite interesting – we decided that both types of paper were outside the scope of this report.

Finally, after a paper was ruled out for any reason, we did not apply the rest of the inclusion/exclusion rules to that paper. Thus, for example, if a paper was too short, or not in English, we did not check whether it was an approach-design-roadmap paper.

2.6 Limitations

One limitation of the literature review is that we only looked at literature where the full text is in English. This only meant deleting one item from our dataset. Nevertheless, there is relevant literature in Arabic and French, and possibly other languages (see, for example, [2, 171]). We conjecture that the failure of our searches to retrieve more of this work is likely because the databases we searched, and the search terms we used, are biased towards English. Investigating the computing education literature in other languages would be a very interesting area for future work.

We looked only at papers that had been published up to the dates of our searches, in mid-March 2024. Since this is a lively area of research, more relevant papers have probably already been published. We aim here only to provide a baseline for further investigation.

We searched only on the title, abstract, and keywords. It is possible, therefore, that there are relevant papers that we did not retrieve. In addition, we did not look at all possible databases. In particular, the databases we chose do not generally index PhD theses, and there are definitely several PhD theses that would have fit our inclusion criteria that our query did not retrieve (See, for example [8, 43, 134, 144, 152, 185]). Further, it is possible that, during the first pass when papers were reviewed by one person each, we may have inadvertently excluded papers from the dataset that should have been included.

Finally, although the thematic analysis was done by two researchers for each paper, other reviewers might reach different conclusions – indeed, probably would, given different backgrounds and experience. We have identified all the papers in the dataset in our bibliography. Another analysis of these papers (and more) would be a welcome addition to the conversation about computing education in African countries.

3 Literature Review: Results

In this section, we characterise the dataset in terms of the answers to our research questions. In Section 4, we expand on these results with a more detailed discussion of several example papers from the dataset.

3.1 RQ1: Number of Papers

After the analysis described in Section 2, our dataset contained 140 papers. These papers were used in our analysis and are indicated with an asterisk (*) at the start of their listing in the references of this report.

3.2 RQ2: Years of Publication

Figure 2 shows a clear growth over time in the number of papers about computing education in African countries. The first paper in our dataset is an outlier, published in 1978 [19], a paper by a Nigerian researcher that described computing education in Nigeria including the history of computing education in Nigeria and the details of computing courses at several Nigerian universities. This is perhaps not entirely surprising since Nigeria was Africa's most populous nation in 1977 [1], as it still is, and obtained its first computer for educational purposes in 1963 – an IBM 1401 – a result of the IBM World Trade Corporation establishing the IBM African Education Centre at the University of Ibadan in Western Nigeria. That year, 52 students from Nigeria and other English-speaking African nations were enrolled in courses in FORTRAN and other programming courses [1].

Following that, there was a gulf in publications and then slow growth until recent years, most of which have ten or more papers. Since 2024 already has 11 papers, even though our data were collected in March, the current year is on track to match or exceed 2023's record of 15 papers.

3.3 RQ3: Countries Where Data Were Gathered

Table 1 shows the 24 African countries from which the papers in our dataset gathered data and indicates the number of papers based on data from each of those countries (in the Total (Data) row). The list includes almost half of the countries in Africa, spanning the continent north to south from Egypt to South Africa, and west to east from Ghana to Kenya. For comparison, we also analysed how many of those papers also included data from countries outside of Africa. These countries are shown in Table 2, again in the Total (Data) row.

Finally, to further explore the connection between the papers in our dataset and African countries, we examined the institutional

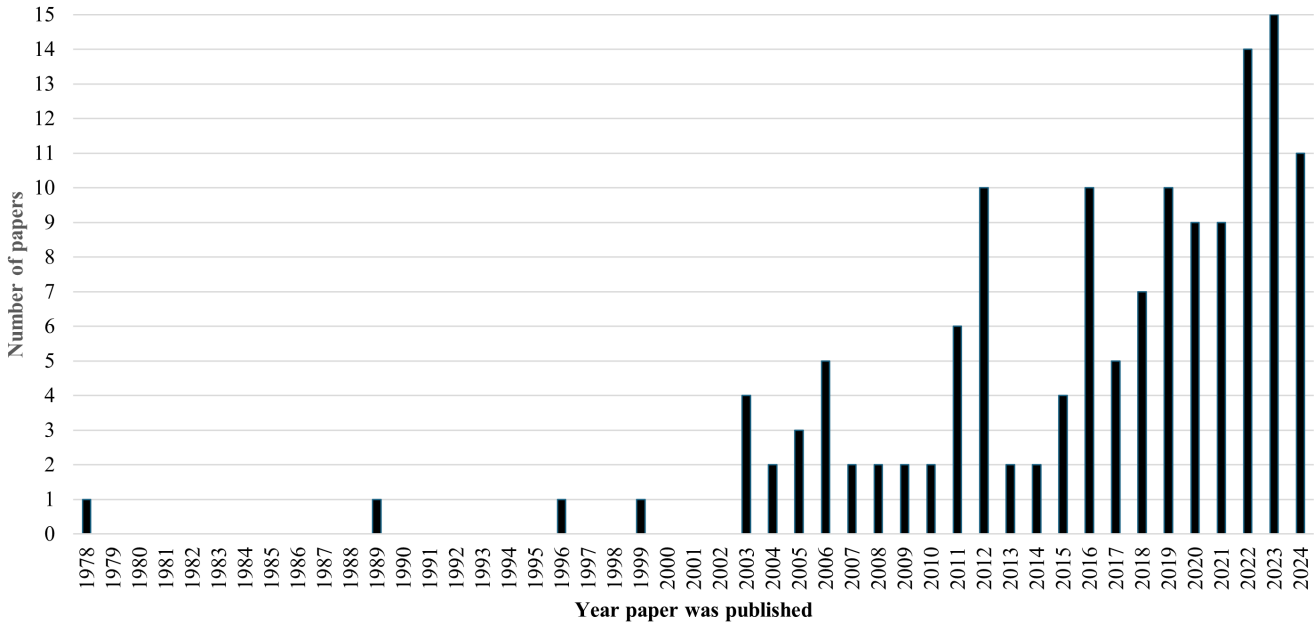

Figure 2: Number of papers published per year

location of the authors of these papers. The way we counted authorship was by paper. For instance, if a given paper had three authors (e.g., Nigeria, Finland, and United States affiliations) and gathered data from two countries (e.g., Nigeria and Finland), we recorded this as one authorship from Nigeria, one authorship from Finland, and one authorship from the United States. Similarly, we would have recorded one instance of data gathered from Nigeria and one instance of data gathered from Finland. Thus, this example paper would result in five data points – three authorships and two locales from which data were gathered.

These results are given in the Total (Authorship) row of Table 1 (for authors with affiliations to an African institution) and in the Total (Authorship) row of Table 2 (for authors with affiliations outside of Africa). When considering the author affiliation results, it should be taken into account that author affiliation is only a rough proxy for the author's connection with Africa: even within our dataset, there are authors who go back and forth between affiliations inside and outside of Africa.

South Africa and Nigeria are quite prominent in both categories of data and authorships. With regard to data from African countries, they are tied at 33 papers each. South Africa is ahead on authorships with 47 papers, as opposed to 21 for Nigeria, indicating that papers about South Africa are more likely to be written by authors currently in South Africa.

Among non-African countries, Finland stands out with 43 papers with authors from Finland reporting on data from Africa (which may or may not be co-authored with authors currently in an African country). This is due to the fact that many Finnish researchers have spent time in Africa and many African researchers are working on, or have completed PhDs in Finland. Finland's number of authorships is second only to South Africa's overall, and among countries outside of Africa, the country with the next highest number is the United States, with 17.

3.4 RQ4: Publication Venues

The 140 papers in our dataset were published in a total of 88 different venues (counting venues with similar but slightly different names, such as Elsevier's *Computers and Education: Artificial Intelligence* and *Computers and Education Open*, as distinct items).

The venues with the most papers were IEEE Frontiers in Education (8 papers); *African Journal of Research in Mathematics, Science and Technology Education* and *Computer Science Education* (7 papers each); *Education and Information Technologies* (6 papers); IEEE IST Africa (5 papers); and IEEE Africon and the SIGCSE Symposium (4 papers each). Of the 88 venues, 76 published only one paper each.

Only 28 of the 140 papers were published in what might be called "core" computing education venues: the conferences ACM ICER, ACM ITiCSE, ACM SIGCSE Symposium, IEEE Frontiers in Education, WiPCSE, and Koli Calling, and the journals *IEEE Transactions on Education*, *ACM Transactions on Computer Science*, and *Computer Science Education*. There was one paper in our dataset from ACM Inroads [34].

3.5 RQ5: Major Topics

The main topics of the papers in our dataset are shown in Table 3. As in the broader computing education literature, programming education is a popular topic. Upper-level courses, however, are investigated in almost as many papers as programming education is. Other frequent topics include infrastructure (hardware, software, internet connections, instructors, and/or technical support),

	Algeria	Angola	Botswana	Burundi	Cameroon	Egypt	Ethiopia	Ghana	Kenya	Lesotho	Liberia	Libya	Mauritius	Morocco	Mozambique	Namibia	Nigeria	Rwanda	Senegal	South Africa	Sudan	Tanzania	Uganda	Zambia
Total (Data)	1	1	4	0	2	8	3	11	10	1	1	1	1	5	2	12	33	2	2	33	2	19	7	2
Total (Authorship)	1	0	5	1	1	7	2	9	6	1	0	1	1	5	1	7	21	0	1	47	1	11	3	1

Table 1: African countries reporting data and authorship (per paper)

	Australia	Cambodia	Canada	China	Czech Republic	Denmark	Finland	France	Germany	Ireland	India	Italy	Japan	Malaysia	Nicaragua	Netherlands	Norway	Portugal	Qatar	Spain	Sri Lanka	Sweden	Taiwan	Thailand	UAE	UK	USA
Total (Data)	0	2	0	0	0	0	1	0	2	0	2	0	0	0	0	0	2	0	2	4	1	0	0	2	0	0	3
Total (Authorship)	1	1	3	3	1	1	43	1	3	3	2	1	2	2	1	2	2	1	0	4	1	11	1	1	3	11	17

Table 2: Non-African countries reporting data and authorship (per paper) in collaboration with authors from Africa and/or reporting on data from Africa

attitudes towards computing, using technology in teaching, collaboration/teamwork, and contextualisation (of curriculum, course materials, or teaching approaches).

4 Literature Review: Examples

Readers are invited to browse the following discussion of selected papers from our dataset.

4.1 Programming Education

The teaching and learning of programming has been a notable focus in the computing education research literature from Africa, as it is in the broader computing education literature. The programming-education papers in our dataset were published from 1996 [88] up to the present day [90]. Some address programming courses at the secondary level (for example [28, 66, 67]), many at the tertiary level (for example [85, 88, 89, 95]), and some in informal learning settings [26, 90].

These papers are based on data from countries spanning the continent from east to west and north to south: Botswana [29, 105, 177], Ethiopia [38], Ghana [17], Kenya [26, 90], Libya [131], Morocco [127], Namibia [112], Nigeria [13, 28, 97, 169, 175], Rwanda [105], Senegal [85], South Africa [86–89, 93, 110, 111, 116, 119, 120], and Tanzania [20, 22–25, 66, 67, 95, 104, 186].

Some examples of those papers follow, organised around questions that computing instructors or researchers might have.

4.1.1 What are some examples of computing course materials contextualised for African countries? The papers in our dataset reported several contextualised assignments for introductory programming courses – assignments that were designed to be relevant to students in a particular time, place, and culture. Duveskog et al. [66, 67], the first papers in our dataset to address contextualisation, reported on a course project at the Kidugala secondary school in the southern highlands of Tanzania, in which introductory programming students coded Java applets and built a website to provide information about HIV/AIDS. This course, offered in a rural secondary school in the southern highlands of Tanzania, was contextualised by the instructor's choice of project: creating a website for HIV/AIDS education. At that time, the death toll from HIV/AIDS in Sub-Saharan Africa was particularly high. Students were encouraged to use both Swahili and English in their website. Having something important and relevant to talk about and knowing that their website would be published on the Internet for the planet to see – even though neither they nor their neighbours had Internet access – was a strong motivation for the students to learn programming.

Anohah and Suhonen [17] is a study of the use of Akan symbols (from a specific culture within Ghana) as a source of examples for an introductory programming course. Students with no previous programming experience were first taught to write Snap programs to display various Akan symbols. After four weeks, they were then taught to write Java programs to display the same symbols. Both quantitative and qualitative results indicated that the use of the Akan symbols led to an increase in both the students' understanding of basic concepts and their motivation to study programming.

Vesisenaho et al. [186] took this strategy one step further, asking introductory programming students at the University of Iringa in Tanzania (then Tumaini University) to reflect on and suggest local applications for software. By the end of the semester, many of the students were able to make connections between programming and their own context.

Awaah et al. [28] reports taking a similar approach. In this project, students in two different Nigerian secondary schools were given a lesson in Python programming. One group's lesson was delivered as a standard lecture. The other lesson illustrated a theory called the "Culturo-Techno-Contextual Approach" that, in practice, involved using contextual examples and asking the students to suggest them as well. An analysis of pre-test and post-test data from both groups

Category	Key concept	Count	Total
Knowledge (*)	Programming	59	149
	AI/ML/robotics	28	
	Software Engineering	16	
	HCI	5	
	Computer ethics	4	
	Data structures	2	
	Foundational and Professional Knowledge		
	Collaboration/Teamwork	27	
	Computational Thinking	8	
Teaching Strategies	Using Tech in Teaching	29	83
	Contextualisation	24	
	Broadening Participation	10	
	Assessment	8	
	Distance Learning	6	
	Pair Programming	4	
	CS Unplugged	2	
Curriculum	Design	26	26
Other Factors	Infrastructure	40	92
	Learning computing - attitudes, perceptions, motivation	34	
	Gender and CS	14	
	Analytics - predictive factors	4	

Table 3: Key themes emphasised in the papers in our dataset – (*) denotes a category derived from CC2020

of students indicated that the Culturo-Techno-Contextual Approach was more effective.

Contextually relevant projects can be powerful motivators in upper-level project courses, as well as introductory programming. Laine and Sutinen [104] described a multi-national project course piloted as part of the contextualised Bachelors' in Information Technology program at the University of Iringa in Tanzania [104]. The course consisted of a single long assignment: developing "an Android-based pervasive learning game" for a museum in Tanzania, in collaboration with students from a Finnish university. The first contextual aspect was the choice of mobile phones, as those were both popular and more widely available than computers. In addition, a mobile app project would prepare students for the "vibrant mobile industry" in Tanzania.

The app itself, the Bagamoyo Caravan, was also deeply grounded in a local context. Designed for the Catholic Museum in Bagamoyo, Tanzania, which depicts the history of slavery in the area, the Bagamoyo Caravan tells the story of slaves' journey in a slave caravan from their home village to Bagamoyo. As a pervasive computing app, with access to the users' location, it was designed to relate the game's story and challenges to the museum exhibits near the users, as they moved around the museum. The language was contextualised by writing the game content in both Swahili and English. The course raised serious challenges, including organising and implementing a multi-national project involving technology that was new to the participants and travel (both between Finland and Tanzania, and between the Tanzanian university and the museum). Nevertheless, the students found the project very motivating, and the authors both recommended that the pilot course be added to the degree program and provided an analysis for those seeking to develop a similar course elsewhere.

Gotel et al. [85] extended WebWork, an open-work software tool for designing and grading practice questions, to computer programming. The resulting system was tested on students in a number of countries around the world, including Senegal. The open-source nature of the software was an advantage, but the difficulty of writing appropriate, clearly understandable questions in a variety of languages was a major obstacle. One lesson of this experiment might be that large-scale, top-down contextualisation is challenging.

For more about contextualisation, see Sections 6 to 13, where we present new contextualised assignments, developed by this Working Group, and discuss that project in more detail.

4.1.2 How many women are studying computing and how (if needed) can we increase that number? There has been a long-standing concern about the percentage of women in computing in Europe and North America [48, 130]. This is not the case in some other countries like Indonesia, Malaysia, and Thailand [184].

The evidence from our dataset is mixed – not surprising, given the diversity of African countries. One of the papers (though not specifically focused on introductory programming) found that "as the percentage of women studying CS has dropped in the U.S. and other [Anglo-Saxon, Scandinavian, and German-speaking] countries, the percentage of women studying computing in Mauritius has increased to levels proportional to the percentage of women in the general population" [5]. On the other hand, Ayalew et al. [29] reports a significant difference between the success of males and females in the introductory programming class at the University of

Botswana, and Liebenberg and Pieterse [111] report that in South Africa only 27% of the students taking programming in secondary schools are girls. Ochwa-Echel [140]'s PhD thesis reported that women made up only 14.8% of the computer science students at Makerere University in Uganda and provided cultural and historical background, along with an in-depth analysis of the factors that led to this disparity.

As a possible solution, Liebenberg and Pieterse [111] investigated the effect of pair programming in a secondary-school computing class in South Africa. It was received very enthusiastically. One of the girls in the class commented: [110, pp. 232-233]:

> But now that the pair programming is introduced I'm like: there's still hope because I'm learning new things, someone is helping me on my mistakes, I'm not actually alone. They're helping me along the way, holding my hand, you know, just introducing me to new ways of actually doing it.

Pair programming could help to create a critical mass of girls taking programming [110, p. 231]:

> I know it (pair programming) will attract more people. You can approach IT knowing you're not alone in it. I know when I came to IT in Grade 11 I thought: "look how little (few) people there are" . . . But if you're working as a team, and you're working with someone you can actually think: "It's not that bad. I shouldn't be so scared. This one knows just as much as I do, I can help him, he can help me, so why not?" I think girls would actually be a bit attracted to it. Yes, the more people that come, not just getting guys, getting girls as well.

One participant summed up the experience as follows [110, p. 232]:

> We all prefer pair programming because when we get in the class, sir gives us a program, the first thing we do: Pair programming we want!

4.1.3 How can mastery learning, supplemental coursework, and tutoring systems be used to support introductory programming? Three papers in our dataset, all from South Africa, discuss possible ways to support students in introductory programming. Greene et al. [88] described the situation in a South African engineering school in 1996 (shortly after the end of apartheid), when many entering students were failing the first-year programming course required for all engineers.

Their response had two parts: a mastery learning course in basic computing skills and a new alternative introductory programming course designed for less experienced students. The mastery learning course, Using Computers, was required for all students, but the least experienced students were assigned more lab time. All work and testing were hands-on in a computer lab. The first lab had the students type on the computer keyboard, which built confidence in the least experienced; the ultimate goal was for the students to learn word processing, spreadsheets, and the mathematics package used in the first-year mathematics course. After the first two weeks, students could request to be tested at any time on their current topic and repeat the test until they passed, so experienced students could complete the course quickly. The self-paced nature of the course was popular and effective; eventually almost all of the students demonstrated all of the desired skills. The new alternative programming course was optional and had only 15 students. It consisted of 1/3 MATLAB programming, 1/3 basic computer concepts (binary numbers, etc.), and 1/3 C programming [88]. It was a small sample, but preliminary results were encouraging.

Twenty years later in the second paper, Marais and Bradshaw [120, p. 339] reported fundamentally the same challenge: "South African universities need to accommodate learners from the flawed public schooling system in South Africa". Private schools provided the needed background, but graduates of the public schools were weak in both science and mathematics. Universities were facing larger and larger enrolments, with greater disparities in background. Marais and Bradshaw [120], arguing that the burden "falls on the universities to align [the ACM/IEEE curriculum] guidelines with their own institutional challenges", also proposed a new course. Unlike Greene et al. [88]'s solution, however, this course was to be offered in parallel with the programming course, and it provided training in what might be called computational thinking: graph reading, logic and inference, simplifying and decomposing problems, etc.

The third paper, Horner and Gouws [93], reported on an e-tutoring support system implemented at a university in South Africa to help introductory programming students in general [93]. The e-tutors were paid employees, required to have a bachelor's degree and to have taken at least the introductory programming course. Each e-tutor was assigned a specific group of students. They worked remotely and did not meet the students in person or grade them.

The e-tutors' primary task was to receive an assignment known as a Midweek Challenge from the lecturer each week, deliver it to the students, and help the students to solve it. Each Midweek Challenge was designed to help prepare the students for that week's assignment, which would be an important part of their grade. In addition to helping with the Midweek Challenges, the e-tutors also helped students with administrative questions, such as loading and configuring software and accessing various educational resources. The tutors received support and respect from the lecturers and were expected to help each other as well as the students.

The results included a reduced dropout rate. The lecturers and tutors felt that the Midweek Challenges "make it easy for students to begin with their programming exercises, and therefore not delay and end up failing or dropping out" [93, p. 35]. In addition, they concluded that the e-tutors' team spirit was a factor in the program's success.

4.1.4 Will a flipped classroom help my students? Apiola et al. [20] sought to understand how students work on their homework outside of class and the degree to which guided in-class problem-solving helps. Data included the results of instructor observations, a student survey, and a controlled experiment in which the class was split in half, with half receiving guidance and half not. The two groups were swapped half-way through the course, and student results were measured by the results of frequent quizzes. The results were contradictory: the students' learning outcomes were similar with and without guidance, but the qualitative results from students, instructors, and researcher-observers all supported guided

problem-solving. They suggested several possible reasons for this discrepancy, one of which was the fact that they were teaching by doing, and assessing by exams. The next step, they concluded, might be to change the assessment strategy.

4.1.5 How should I assess my programming students? Particularly for large classes, it would be helpful to automate the grading of programs. One paper from South Africa investigated the feasibility of automated program grading [111]. The researchers, Liebenberg et al., tested an automatic assessment tool and concluded that the tool could be useful for verifying functional correctness during lab sessions, but not for situations where functional, conceptual, and structural correctness all need to be evaluated. Moreover, the tool worked better for the better students. Still, it might be helpful as a first pass, to complete the initial phase of grading.

Another common issue is how to assess individual contributions to a pair or group project. This issue is addressed by Hahn et al. [89], also in a South African context. This paper investigated a strategy in which peer, individual, and facilitator evaluations were done at the end of each pair-programming session, using rubrics provided by the instructors. There were four pair-programming sessions, and pairs were re-assigned for each one. They concluded that the peer evaluations became more reliable as time went on and recommended the combination of all three types of evaluation.

4.1.6 What about upper-level courses? Papers about two contextualised upper-level courses are discussed in Section 4.1.1 (a project course [104]) and Section 4.2.3 (artificial intelligence [164]). A third paper about upper-level courses, Marshall et al. [122], examines the roles students take on within software teams. During a single project, they were required to work in four different instructor-assigned teams. Based on years of student data about their contributions and their peers', the paper concludes that student roles are not fixed. Students who do not contribute in one context can make strong contributions in another. This has intriguing implications for anyone teaching team project courses.

4.1.7 Can/should I use ChatGPT or similar tools? The latest technology to attract attention has been ChatGPT, along with similar artificial intelligence code generation and explanation tools. Introductory programming instructors from both Botswana and Rwanda participated in an interview study of instructors' attitudes towards these new tools, along with instructors from all five of the other inhabited continents [105]. The researchers captured a snapshot of instructors' opinions of these tools, after they had been around for about a year and before the field had reached a "consensus on best practices". In the short run, they reported, most instructors are taking measures to prevent their use for cheating. The Rwanda instructor, for example, suggested administering exams on paper, or alternatively oral exams.

In the long run, the instructors were divided between banning use of ChatGPT and incorporating it into their teaching. The Botswana instructor raised a concern about surface or fragile knowledge, suggesting that students might use solutions from these tools in the same way as they might copy from StackOverflow, without a deep understanding of the code they're using. Surface knowledge is a long-term issue, echoing the concern about students memorising code snippets in earlier papers in our dataset [23, 25].

4.1.8 Can teaching programming be a way of peacemaking? One paper is about, not programming, but a potential side effect of programming. Set in a refugee camp in Kenya, Arawjo et al. investigates "[t]he potential of computing education as a site for inter-cultural learning" [26, p. 52:2]. Computing attracts a diverse group of people, and collaboration can lead to empathy. Surprisingly, it was sometimes the software or hardware breakdowns that led to shared laughter; when everything went smoothly, students were less likely to collaborate. Still, instructors need to guide the process carefully. The researchers give the example of a young Muslim girl learning to shake hands despite parental prohibitions. "Learning a foreign gesture for greeting is undoubtedly intercultural learning, but some cultures may see specific forms of greeting as a transgression against modesty" [26, p. 52:16].

4.2 Artificial Intelligence, Machine Learning, and Robotics

Artificial intelligence (AI) and its components, including machine learning (ML) and robotics, have been a major focus in African computing education research from 2005 [62] to the present [96]. We found 28 papers that explored these topics at levels ranging from K-12 through university level. Papers investigated how AI, ML, and robotics can be promoted through robotic competition, science engagement for under-served communities, multi-national and industrial collaboration, informal education techniques in science centres, contextualisation/pedagogy and curriculum initiatives. We discuss these themes in the remainder of this section.

4.2.1 Robotics competitions. Weinert and Dirk Pensky [191, 192] introduced Robotino, an autonomous mobile mechatronic learning and research system, as a standardised platform for education across the fields of engineering and information technology. These papers highlight how different robotics competition across Africa, such as South Africa WorldSkills, South Africa Cyber Junkyard, and Kenya National Robot Contest, have been useful in addressing the skills-gap divide that is acutely prevalent between educational institutions and industrial manufacturing companies. The papers present the impact of these learning systems and competitions on the training of industrially-relevant engineers, equipping students with tangible skill and entrepreneurial passion to enter the workplace and drive the economies of the future.

Davrajh & Stopforth [58] reported on a project that introduced previously disadvantaged secondary school students to the field of engineering through robotics using LEGO Mindstorms. The authors developed a training session and designed a competition to stimulate industrial applications. Feedback from a survey of students and educators indicates that a pedagogical approach via the LEGO platform, integrated with professional development criteria, effectively engaged students whilst exposing them to professional requirements.

4.2.2 Science engagement in under-served communities. Dirsuweit et al. [64] explored whether girls encounter barriers to full participation in robotics competition teams, through a community engagement project that promotes science engagement in predominantly disadvantaged communities. Based on responses from the participants involved, the paper suggests that their community

needs to "build a substantive strategy around how their communities of practice can function as spaces that, not only attract girls but actively promote gender equity."

Dias et al. [62] presents the challenges and benefits of three higher education initiatives in Sri Lanka, Ghana, and the USA, examining the potential intersections of robotics and its component technologies with education and sustainable development. The authors identified some principles as important elements for success in higher education in developing communities. These include participatory research and design, empowerment rather than aid, shared resources, local partnerships, global partnerships, sustainability and evaluation metrics.

Dias et al. [61] also address the challenges and benefits of undergraduate robotics education in technologically under-served communities. The paper presented the successes, limitations, and lessons learned in two case studies in Qatar and Ghana. In both cases studies, the participants were reported to have learned about the intricacies, frustrations, and joys of working with hardware and software integration, and the importance of practical applications and challenges.

Building on previous work [61], Dias et al. [60] present their experiences in designing and teaching introductory robotics courses in Qatar and Ghana. They discussed the motivation, challenges, approach, impact, similarities and differences in teaching robotics in these two settings. The case studies illustrate two possible models for establishing such courses in technologically emerging regions. The paper emphasised that international partnerships can help to address the scarcity of the necessary experience to develop and teach robotics courses in technologically emerging regions.

4.2.3 Multi-national and industrial collaboration. Shipepe et al. [164] report their experiences of teaching artificial intelligence in a Namibian university through collaboration with a Finnish university and a few companies. Their approach includes creative applications of artificial intelligence to meet the contextual requirements by Africanising an artificial intelligence course. The findings from the study suggest that their effort in contextualising course material sparks interest in inventing and facilitates interactions and collaboration among universities and industry in and beyond Africa.

Soudi [168] presented Egyptian case studies in teaching robotics for school students from an early age. The paper discussed several initiatives on robotics competition and program to support robotics education. The author highlighted partnership among universities, schools and industry to promote robotics education

4.2.4 Informal education in a science centre. Elsayed [72] explored the effect of applying informal education techniques in science centres in the fields of artificial intelligence and robotics. The author highlighted the values of learning through science centres including motivating and enriching environments for learning and free-choice environment that strongly engages the attention of learners. The paper emphasised the immediate and long-term impact of science centres regardless of cultural and social variation between different communities. Elsayed [72] further suggested that informal learning space could be a model to complement the formal educational systems in the developing regions of Africa.

4.2.5 Contextualisation / pedagogy. Nannim et al. [136] investigated the effect of a project-based Arduino programming course on students' computational thinking skills. Through quasi-experimental research design, 73 undergraduate students in Nigeria took part in the study. The paper found that students who were taught using the robots outperformed those taught by the conventional lecture method.

Using co-design pedagogy, Sanusi et al. [159] collaboratively designed and prototyped machine learning applications with 43 eighth grade students (ages 11 to 14) in a Nigerian school to support their learning about artificial intelligence. The study found that the ideas generated by the students indicates that they began to identify the applicability of machine learning to their daily lives and as a solution to a plethora of societal challenges. The authors posit that design-oriented pedagogy approaches could be valuable for teaching ML in the middle school education in African settings.

Similarly, Sanusi et al. [161] engaged students with machine learning through extracurricular activities in a Nigerian middle school. Using a pre-post design with open-ended surveys and written assessments, the study found an increase in students' awareness about machine learning, including ethical and societal implication. Based on the relatable activities that were used to support students' machine-learning comprehension, the findings highlight the implication of using materials students can identify with in pedagogical design.

4.2.6 Curriculum. Pillay et al. [148] discusses how artificial intelligence can be incorporated into engineering and computer science education to prepare for the fourth industrial revolution in South Africa. The paper highlights the online courses and short courses with certifications for practitioners and for degrees in artificial intelligence or data science. Besides using intelligent tutoring systems and teaching assistants, the paper examines mechanisms and a case study for involving industry in engineering education at a South African university.

4.3 Infrastructure

The papers in our dataset frequently discuss the need for better computing-education infrastructure. Some schools lack computers, or if they have them, do not have enough to accommodate all students [68, 87, 112, 179]. The hardware and software may be outdated and slow [68]. There may be no internet connection [57, 66, 67]. Most rural areas do not have internet connectivity, and when it is available, it is often slow and expensive [68]. Even universities may not have reliable internet: "The ... Senegalese professors often use a computer to demonstrate concepts in the classroom, but do not require a connection to the Internet to do so" [85, p. 6]. Schools' limited funds is mostly allocated to more essential needs such as water supply, electricity, nutrition, and healthcare for students [36]. Particularly in rural areas, electricity may be unreliable or unavailable [33, 68, 180].

Obtaining the necessary software [42], specialised software and hardware for upper-level courses [135, 139], student textbooks [57], and funding for graduate students [91] can also be challenging.

In addition, there are shortages of teachers at the primary/secondary [179] and university [57] levels. Availability of teaching materials remains a challenge[179]. There is a need to improve

teacher professional development and empower instructors to use the available resources [179].

While availability and access to computers and the internet is still a challenge in Africa, this varies greatly from region to region and country to country. For example, many secondary schools in Botswana have access to computers and the internet, although some primary schools still struggle [179]. In Kenya, the government has been distributing laptops to some primary schools to achieve their "Vision 2030" [179].

Dasuki et al. [57] is a case study of the development of a computing degree program in Nigeria with generally excellent infrastructure. The program was located at Baze University, a private university in Nigeria that was, at the time of the article, only four years old and provided air-conditioned classrooms with internet connections and a specialised lab for teaching networking. Nevertheless, there were too few textbooks for the students, unreliable internet in the student computer labs, inadequate access to research journals for the faculty, and too few qualified faculty.

For computing-education programs that are facing infrastructure challenges, the papers in our dataset also report on solutions, many of which are applicable outside of Africa as well.

4.3.1 Computers. Several papers address the challenge of making sufficient computers available for students. In one project, Namibian secondary students learned to program in Python using Raspberry PIs [112]. The paper argues that the traditional platform for introductory programming should be replaced by Raspberry PIs (one for each student). Besides being affordable, the devices are much more motivating for students [112, p. 2]:

> [O]ne can now write code like `repeat_until_user_presses_button` or even `repeat_until_temperature_exceeds_100C`. The benefits of this approach will quickly become apparent. For example, the programming semantics of a for-loop can now translate into the flashing of an LED or the sounding of a buzzer and provide instantaneous and tangible consequences.

South African computing-education students responded positively to another physical device, the Arduino [87]. One, comparing the Arduino to their previous experience with Scratch, echoed the argument made about the Raspberry PI: "'Scratch together with Arduino is much more interesting, better than using the original Scratch because you see stuff happening" [87, p. 97].

Earlier, researchers in Tanzania reviewed several hardware platforms for teaching introductory programming, identified criteria, and recommended the Simputer, a basic hand-held environment designed to be affordable and usable in developing countries. They tested the Simputer at a secondary school in rural Tanzania. One student noted that the Simputer "is not difficult and complex as I thought before" [68, p. 3]. While technology has changed substantially since 2004, when this paper was published, the criteria and the analysis are still relevant.

It can be even more challenging to obtain the needed hardware and software for upper-level computing courses. For example, Dasuki et al. [57] discusses how to obtain funding for a computer networking lab in Nigeria, and Mwasaga and Joy [135] describe the design and testing of a hardware device for teaching high-performance computing in Tanzania [135]. As with the Simputer, the process is as interesting as the final result. The questions for the participants were very pragmatic: What do you want the device to do (functional requirements), and what other changes would you like (non-functional requirements)? The overall result was that the changes were feasible, and it would thus be possible to design a device that would be both usable and affordable.

4.3.2 Software. . In general, software solutions focused on free and open-source software. While experienced staff may be needed to maintain open-source software, its advantages in terms of quality and affordability are compelling [36]. For programming courses, Brown and Connan [42] describes a Linux environment distributed on a memory stick to university students in South Africa. Mavengere and Ruohonen [124] evaluates the use of Moodle in a database course in Mozambique. The failure rate went down, and students highlighted the availability of materials that can be downloaded, ease of discussing and addressing challenges from different perspectives, and effective communication between students and teachers as some of the advantages.

Thinyane et al. [174] reports on the successful use of Skype and GoogleDocs as a "light-weight virtual classroom system" for two upper-level honours classes that Rhodes University in South Africa was offering to remote students at the University of Namibia. Compared to the commercial software they had previously tried, the user interface was uncluttered and easy to learn and use. The remote students appreciated the easy access to teaching materials and lecture slides on GoogleDocs, and when low bandwidth led to network issues, the ability to switch easily from video to audio or from audio to chat messages was a key feature.

Oyelere et al. [146] reports on the design, development, and testing of an Android-based mobile learning system, MobileEdu. MobileEdu was motivated by overcrowded undergraduate computer science classes, the affordability and ubiquity of mobile phones, and the opportunity to reach a larger number of students without the constraints of the physical classroom. Features included enhanced class management, student participation, effective communication, quizzes, support for groupwork, interaction with course materials, and notifications. MobileEdu was implemented and tested in a third-year Software Analysis and Design course in a Nigerian university. The difference between pre- and post-test scores was significantly better for the students using MobileEdu than the control group, with a medium effect size, and student attitudes, based on both quantitative survey data and interviews, were also considerably better.

4.3.3 Teachers. Work is being done to increase the supply of teachers at the secondary, undergraduate, masters' and doctoral levels. In Nigeria, Scratch has been found to be helpful in teaching programming to pre-service teachers. Participants indicated that their foundation of programming concepts was "well grounded", and one participant mentioned, "Being taught something repeatedly, you will get more understanding about the topic" [175, p. 273]. The researchers believe that increased exercises, particularly those focusing on challenging concepts such as the use of variables, algorithms, and control structures, will deepen their understanding of these topics further.

In South Africa, pre-service teachers who had already experienced programming in Scratch were given the opportunity to program an Arduino [86]. As noted above, they found the feedback from a physical device made programming more interesting. In the background of this study, however, was a Department of Education proposal that a coding and robotics course be offered in primary school, which, if the proposal was successful, might be taught by the study's participants. The authors cautioned that, given the poor record of South African students in reading, science, technology, and engineering, such a course would need to be "a simple experience that induces motivation and engagement" [87, p. 99].

Apiola et al. [21], as noted above, reports on a new PhD program designed for students from African countries who want to focus on computing education research. One of its stated goals was to increase the capacity for computing education research in (and about) sub-Saharan Africa; it also had the potential to increase the number of university faculty. Harsh et al. [91] describes the development of computing research in Kenya and Uganda. The paper provides insights into the process of obtaining funding, the possible sources and the strings that are tied to them, with particular focus on the interplay of individual actors, government policy, and industry.

Finally, Daramola [56] is an auto-ethnographic study of the author's experience advising PhD students at universities in both Nigeria and South Africa. Some of his observations are particularly relevant to programs with budget constraints, such as those in developing countries. For example, he discusses helping his students to cope with the challenge of obtaining the papers and journal articles they need [56, p. 11]:

> A common problem for students particularly in developing countries is the inability to access papers that require a paid subscription. ... [W]hen students encounter papers that are relevant to their work but could not be accessed due to paywall restrictions, they are instructed on additional steps that could be taken to get the papers. These include reaching out to the author directly on ResearchGate or via their university email address, and also checking if free copies of such papers exist in open-access databases such as ArXiv, Base, CORE, Semantic Scholar etc.

Many of its perceptive observations are also relevant to computing graduate students and advisors generally.

5 Literature Review: Discussion

As noted above in Section 3.4, there were over 80 distinct venues represented in our dataset, and the large majority of these published only one or two of the papers in our dataset. Only 40 of the initial 462 papers retrieved were from the ACM DL, and only 31 of the 140 papers in our final dataset were published in what might be called "core" computing education venues. This suggests that our previous impression that African countries were under-represented in the computing education literature may be wrong: perhaps we were just looking in the wrong place. The good news is that a substantial and interesting literature can be found.

Our review suggests that international collaboration influences computing education research in Africa (as shown in Table 2). In particular, Finland is the pre-eminent non-African country reporting on African data and/or co-authoring with authors affiliated with African institutions. These co-authorships extend from 2003 [67] to the present [153]. The University of Eastern Finland has taken the lead in this effort. They co-established a program designed to train African PhD students and candidates in computing education research [21] and they have research hubs in Tanzania and Uganda [182].

There are surprisingly few papers about teaching specific computing concepts in our dataset, compared to the broader literature. One example is Jegede et al. [97], which provides a detailed analysis of the types of syntax errors made by students learning Java in Nigeria. Another paper, also from a university in Nigeria, describes a virtual reality game that might help students better understand object-oriented concepts such as method overloading [170]. We did find papers about student perceptions of computing [114, 143], computer security [33], and artificial intelligence [96, 153]. But a close examination of how students understand particular computing concepts might be an interesting direction for future work.

Our review of the literature found a substantial body of work about developing computing education materials for specific communities in Africa and their respective cultures. These efforts have been in both programming education and in upper-level computing subjects (for example, [136, 164]). In the following sections we extend this work, presenting a set of contextualised materials designed for six different African countries.

6 Contextual Materials: Overview

Students are more likely to engage with and retain information that is presented in a context familiar to them. Contextualising education is particularly important for addressing the academic challenges faced by historically marginalized groups, such as African American, Native American, and Latino students [103, 189]. Research demonstrates that teaching approaches that disregard students' cultural contexts risk perpetuating inequities by alienating learners whose lived experiences differ from those assumed by traditional pedagogy [103].

Traditional introductory computing courses often fail to engage a diverse student population, particularly in regions where cultural contexts are not reflected in the curriculum. In Africa, this issue is pronounced, with university-level courses often using generic materials [34] that do not resonate with students' cultural backgrounds. This disconnect may hinder student engagement and retention in computer science programs [103]. Addressing this disconnect, contextualised approaches have shown significant benefits. For example, using Akan symbols in Ghana increased understanding and interest in programming [17], while creating HIV/AIDS educational websites in Tanzania fostered motivation through a relevant and significant project [66, 67].

Faculty can make computing concepts more relatable and understandable by integrating culturally relevant content into the curriculum. This report presents contextualised materials for introductory programming courses at the university level (often referred

to as "CS1") intended to benefit lecturers who wish to adopt culturally relevant content.

- The objective of our work is to develop contextualised materials for faculty teaching introductory computing in African universities and evaluate the potential usefulness of these materials.

Section 7 discusses related work on contextualised materials outside and within Africa. In this section, we also present Bank's multi-level approach to integrating cultural content into education, beginning with the Contributions approach, which focuses on highlighting important national figures and events, followed by the second approach, Additive, which involves adding cultural content, themes and perspectives to the existing curriculum. The third approach, Transformation is more involved and centres the cultural groups' as opposed to just appending to existing curricula and the final level, the Social Action approach, ensures that students become engaged in the process and are empowered to address societal issues [35].

Section 8 describes our project's methods for developing contextualised materials based on applying Bank's Additive Approach and conducting a pilot evaluation with faculty from various African universities. Section 9 provides practical examples of our materials, illustrating how faculty can contextualise materials for teaching and learning computing in African countries. The results of our pilot evaluation are given in Section 11, a brief coverage of the threats and limitations of the study in Section 12, and a discussion of the contextual materials in Section 13. This report's overall conclusions and suggestions for future work are found in Section 14.

7 Contextual Materials: Related Work

In this section, we situate the contextual materials we have developed within the context of previous work. We begin by discussing computing education contextualisation papers from outside Africa, followed by a review of contextualisation in Africa, drawing from our exploratory literature review (Sections 2 through 5). Next, we address the pedagogy behind contextualised materials, and finally, we present the theoretical foundation guiding our project to develop these materials.

7.1 Outside of Africa

As we strive to make computing accessible to diverse groups, it is crucial to adopt approaches that would enhance their ability to learn the concepts. Contextualising computing education, therefore, is a global challenge. This section will briefly highlight approaches to contextualisation outside of Africa, including family-focused interventions, teacher-centered bilingual instruction, contextualised materials, and curriculum modifications.

7.1.1 Family-Focused Interventions. Studies such as Doyle et al. [65], have created culturally inclusive environments for underrepresented groups, especially Spanish-speaking students, through family-oriented interventions. By conducting non-formal education workshops in familiar community settings and providing bilingual materials, these interventions have successfully engaged families, built confidence, and fostered supportive environments for STEM education and careers.

7.1.2 Teacher-Centered Bilingual Instruction. Research has highlighted the impact of bilingual instruction in computer science education. Villegas et al. [187] investigated the use of bilingual instruction (Spanish and English) in high school programming courses, finding no significant difference in academic performance between bilingual and English-only groups. However, students in the bilingual group showed increased inquiry and higher-order thinking skills. Similar studies involving students from India, such as Pal and Iyer [147] and Soosai Raj et al. Soosai Raj et al. [167], reported mixed results on learning outcomes but noted improved engagement and classroom interaction in bilingual settings. These findings suggest that while bilingual instruction may not uniformly improve academic performance, it enhances student engagement and participation in learning activities.

7.1.3 Curriculum Modifications. Other studies have explored *curriculum modifications* to enhance inclusivity and cultural relevance in computer science education. Franklin et al. [79] developed a culturally relevant summer program integrating animal conservation and Mayan culture, effectively attracting underrepresented minorities to computer science. Tran et al. [176] conducted co-design sessions with K-12 teachers to harmonise Scratch Encore lessons, incorporating students' personal experiences and cultural themes. In New Zealand, Karetai et al. [100] introduced distinct curricula (Digital Technologies and Hangarau Matahiko) to decolonize computer science education, integrating Māori language, customs, and computational thinking to foster cultural inclusivity.

7.1.4 Contextualised Materials. Research integrating *cultural narratives* into computer science education has shown significant benefits. Lusa et al. [117] utilised hip-hop music and the Sonic Pi platform in a three-week online camp to engage Black and Latino middle school students, resulting in improved computing enjoyment, confidence, sense of belonging, and persistence. Chipps et al. [49] employed Skokomish storytelling to teach event-driven programming to rural American Indian middle school students, demonstrating positive outcomes in student engagement and interest in computer science. The ANCESTOR program in British Columbia [39] also integrated digital storytelling to introduce Aboriginal youth to computer science careers, emphasising indigenous knowledge and holistic learning practices to enhance inclusivity and relevance in educational settings. In the realm of *educational tools*, researchers are integrating culturally relevant content in educational tools to enrich learning experiences and outcomes. In the USA, Eglash et al. [70] explored fractal simulations of African designs with 10th-grade students, using culturally specific examples like scaling patterns in cornrow hairstyles. Their study in a high school computing class showed that integrating these tools significantly boosted both academic performance and attitudes towards computing, highlighting the potential of culturally relevant content to enhance engagement and learning in diverse educational settings.

7.2 In African Countries

In this section, we extend the discussion of contextualisation beyond the selected papers discussed in Section 4.1.1 to provide a background for the materials we have developed. Contextualisation of computing education – developing and using teaching materials

and approaches that are culturally relevant to the students being taught – has been an important theme in the African computing education literature from at least 2003 [67] to the present [90]. Our review of the literature reveals that studies have been conducted in a variety of contexts: secondary schools, universities, graduate programs, and informal learning settings, in Ghana [17, 18], Kenya [90], Namibia [76, 164], Nigeria [28], Sudan [55], and Tanzania [22, 24, 66–68, 104, 135, 186]. Much of the work involves adapting western materials to a local context, but one interesting paper describes the adaptation of a masters' program from one African country for use in another [55].

Research on contextualisation, along with various frameworks, contributes to a broader effort to adapt computing education to the diverse cultural and educational needs of underserved communities, such as African students in the field of computer science. Banks' Additive Approach focuses on incorporating cultural content into existing curricula, and Anohah and Suhonen's work [17, 18] extends this by incorporating indigenous knowledge into computing concepts, and Awaah's Culturo-Techno-Approach [28] emphasises student-generated examples from their own cultural contexts. This framing allows for a better understanding of how contextualised teaching materials and pedagogy can evolve to meet the specific needs of African learners. In the following subsections, we explore how contextualised teaching materials, pedagogy, and curriculum have been adapted across various contexts in Africa.

7.2.1 Contextualised materials. The most closely related papers to the project presented here are those that describe contextualised teaching materials (the Level 2 Additive Approach, according to Banks [35]). The Additive Approach involves teachers appending cultural content, themes and perspectives to the existing curriculum. For example, assignments have incorporated culturally and socially relevant topics, such as HIV/AIDS education in Tanzania [66, 67] and slavery [104]. Additionally, the use of local languages alongside English has been implemented to enhance understanding and engagement for both the students themselves and potential users of the software they develop [67, 68, 104].

Cultural artefacts, including symbols, fabrics, and traditional games, have been employed to explain computing concepts effectively. Anohah and Suhonen [17, 18] present a general framework for this type of contextualisation and then describe a concrete investigation into the use of symbols, fabrics, and traditional games from Akan culture to illustrate computing concepts such as loops and recursion. In this study, they surveyed 41 computing educators who were knowledgeable about Akan symbols and then did follow-up interviews with 22 of the participants.

The quotations express the views of African computing educators on the subject of contextualisation, offering valuable insights into the unique cultural and educational needs of the region [17, p. 57]:

> I know Africa has rich cultural heritage to contextualise computing education. What we need is revised national teaching strategies to include local ideas in computing lessons or lectures. The cultural products are already known to our learners and remain strong previous knowledge for computing education.…

After investigating the educators' perspective, Anohah and Suhonen implemented this approach in a secondary school in Ghana [18].

7.2.2 Contextualised pedagogy. Contextualisation can extend beyond teaching materials to the way we teach. Apiola and Tedre [22] argued for the contextualisation of pedagogical techniques in introductory programming. Changes included reducing the amount of lecture time, incorporating live coding into the lectures, and converting the remaining class time to an "open learning environment", in which students spend much of their class time completing a series of programming exercises with feedback and guidance from the instructors.

The transformation of an upper-level artificial intelligence course at a Namibian university illustrates changes to both teaching materials and pedagogy [164]. The authors, Shipepe et al., describe how the module's contents were redesigned [164, p. 1]:

> The topics were explicitly related to various opportunities opened by the Fourth Industrial Revolution (like smart countryside, or self-driving cars for disabled students). Students were learning by applying existing AI tools to design solutions for real-life problems on the ground, with users (like IoT-controlled gardening or enhancing education by robotics). Indigenous knowledge systems and representation by the multitude of languages and cultures were related to AI-based knowledge representation.

The revised pedagogy included active learning (as suggested by [22]) and more [164, p. 1]:

> Pedagogical principles for the transformation included flipped classroom for increasing student engagement and responsibility, diversifying assessment methods, rethinking the course prerequisites, differentiation of students' individual learning goals, and enlarging the learning community by inviting stakeholders and experts from outside the university.

In addition, the paper suggests that to make teamwork effective, "Teamwork within the AI course requires continuous and transparent elaboration of every team member's contribution to the project" [164, p. 8]. They concluded that "Africanizing an AI course can spark the interest to invent AI-based solutions to key African challenges" [164, p. 7], and argued that "The results indicate that Africanization gives a novel impetus to reform Computing education. This applies, first, in Africa, but secondly and equally importantly, beyond Africa" [164, p. 8].

In a paper about software engineering, Fendler and Winschiers-Theophilus [76] argue that we need to change, not just the examples, the hardware, or the pedagogy, but *what* we teach: the software engineering process itself needs to be contextualised. Like Shipepe et al. [164], they argued that effective student teamwork is different in a Namibian context. In addition, prototypes or questionnaires are ineffective strategies for eliciting software requirements when respondents tend to give what they believe to be the expected answer, a "behaviour ... deeply rooted in a culture of conflict avoidance oriented towards listener satisfaction" [76, p. 602]. These are factors that would affect software development on the job, as well as in an

educational setting. The authors propose a framework for developing a contextualised software engineering process, and illustrate how that framework might be applied to the Namibian context.

Some papers cross the border into what Banks [35] might call the Transformation Approach to contextualisation, by obtaining input from students about their culture. As noted above in Section 4, Vesisenaho et al. [186] reported on an introductory programming class in which students were taught to think about ways in which programming could be applied in their local context, and Awaah et al. [28]'s Culturo-Techno-Approach to teaching introductory programming, as well as incorporating contextually relevant materials into lectures, asked students for their own examples.

7.2.3 Contextualised curriculum. Contextualisation is not limited to individual assignments or modules (courses). Several of the papers we looked at reported changes to an entire degree program, either undergraduate or graduate. Dasuki et al. [57] illustrates a minimal approach to contextualisation. This paper is a case study of the challenges involved in implementing a British computing degree program at Baze University, a private university in Nigeria that was, at the time of the article, only four years old. An important part of the context is the Nigerian combination of public and private universities, with public universities receiving most of the public education funding and private universities chartered by the government, but largely funded by their founders. The private universities argue that since they're performing a public service – education – they should receive some of the public funding, but this is controversial. In addition, the competition for funding is surely intensified by the extremely rapid growth in the number of universities – 12 new federal universities, 3 new state universities, and 9 new private universities – in the previous four years. The one instance of adapting the curriculum to the Nigerian context is the addition of a three-month "industrial training", which is required by the Nigerian government for all science students. This change seems to be regarded with regret, however, as it prevents the students from completing their degree in three years, as is standard in the United Kingdom.

By contrast, the computing curriculum of Tumaini University in Tanzania had contextualisation as one of its primary goals. Contextualisation works on three levels: curriculum, topic, and pedagogy. Students were encouraged to identify societal expectations and potential local applications of software starting early in their program. Other core design principles included an emphasis on project work and a continuous cycle of research to develop and improve the program.[2]

At the masters' degree level, Cronjé reported on the adaptation of a program developed at the University of Pretoria in South Africa for use at the Sudan University of Science and Technology in Khartoum, Sudan [55]. While the program is focused on training computing teachers, rather than computing, the paper is valuable for its rich, concrete description of specific contextual differences, and particularly because it illustrates differences between two African countries. Differences ranged from small ones, like the timing of the lunch break, to larger ones, like the national holidays that affected the timing of classes and assignments, to the society's fundamental educational goals. The author gives a memorable description of the profound differences in those fundamental goals [55, p. 284]:

> In my country, I said, the purpose of education is ... [t]o help people get jobs and make money. I added that it was probably the aim of all education. A representative from the Ministry of education interrupted me. "In this country," he said, "The purpose of education is to teach people to live together in peace and harmony." This was greeted with much enthusiasm from the audience. I had forgotten that in their war-torn country, they had more important things than money on their minds.

At the Ph.D. level Apiola et al. [21] report on a computer science program designed to build research capacity in sub-Saharan Africa. The program, a joint project of the University of Eastern Finland and the College of Business Education in Dar es Salaam, Tanzania, built on an established program in which students could obtain a Ph.D. from the university at distance, visiting Finland on several occasions during their studies, but not relocating there. By the time the partnership with Tanzania began, 14 Ph.D. students had already graduated from the program, including at least one based in Africa.

Thus, the first contextualisation was physical: students were able to study while still living and working in Africa. Several faculty members from the College of Business Education were selected to participate in the program. Second, the program was designed to fit the engineering tradition in computer science, as opposed to more theoretical work in, for example, algorithms, as engineering had more connection with the practical development goals of many countries in sub-Saharan Africa. Third, the students' research projects were contextualised. Some projects focused on information technology solutions for local businesses; one interesting example was the street vendors. Education-oriented projects focused on new ways of teaching introductory programming, investigating ways to teach high-performance computing at university level, and the use of educational technology to improve education in local schools [21].

As demonstrated by previous work on contextualisation, it is not merely about modifying content; it also involves reshaping pedagogical practices to better align with local learning styles and experiences. This lays the foundation for a deeper exploration of the additional perspectives that support the development of contextualised materials, which we address in the following section. in the following section.

7.3 Perspectives on Contextualising Materials

Contextualising materials should not be done in an ad-hoc approach but should instead consider the learning outcomes, students' existing knowledge, cultural context, instructors' positionality, resources and other considerations. We believe frameworks and pedagogy are needed to ensure contextualised materials are developed with the students at the centre to benefit their learning. In some cases, research in contextualised computing education is motivated by a concrete need to find the most effective way to teach the particular students in a particular place (See, e.g., [55, 67, 90, 104]). In others, it is motivated by a philosophical commitment to Africanising, or

[2]This is similar to the Scholarship of Teaching and Learning. For a good brief description of the Scholarship of Teaching and Learning in a computing education context, see Tenenberg and McCartney [173].

de-colonialising, a technology with its roots in the West [183]. In this section, we present frameworks, pedagogy and guidelines that have been used to inform the contextualisation process; following this, we conclude with the framework we adopted for our work.

7.3.1 Ethnocomputing. Ethnocomputing is the study of the interaction between culture and computing. From a pedagogy lens, it is a broad umbrella for discussing how computing impacts teaching within cultural contexts. Within the African context, there are a number of papers that have used ethnocomputing as a justification for contextualisation computing materials. Duveskog discusses this in [66, p. 2]:

> Technical concepts of Computer Science ... have been developed within a culture, and are learned in a community. Thus, they are specific to [the] Western scientific community „ Instead of taking our categorisations and ideas to the developing countries, we would like to see CSE originating from the cultures, using the cultures' concepts, artefacts, and knowledge to contextualise the science.

This understanding led to the conclusion that more needed to be changed than just one course [66, p. 6]:

> We cannot just translate a Western curriculum, for instance, to Tanzania. Some courses should be replaced by new ones. The whole curriculum has to be designed from the Tanzanian point of view. This does not imply a lower quality, but probably quite a different perspective.

Van der Poll et al. discuss previous work in ethnocomputing and lay out an underlying philosophy of decolonising computing education [183]. The paper offers no simple answers, but provides an extensive discussion of related work and critiques of others' work and its underlying assumptions.

7.3.2 Culturo-Techno-Contextual Approach. Awaah et al. describe a quasi-experimental evaluation of the Culturo-Techno-Contextual Approach. Two separate classes were each given one Python session, one taught by lecture and one using the Culturo-Techno-Contextual Approach. In practice, this meant using culturally relevant examples and encouraging the students to come up with others. The "Culturo-Techno-Contextual Approach" is defined as "a teaching method based on culture, technology, and the context or the environment in which teaching and learning occur ... anchored on Kwame Nkrumah's ethnophilosophy for culture, Martin Heidegger's techno-philosophy for technology, and Michael Williams' contextualism for the contextual element" [28, p. 2]. Ethno-philosophy, in turn, is "rooted in the idea of the uniqueness and specific culture of the African people as pioneered by Kwame Nkrumah, who opined that African culture is unique from European ways of life but in no sense inferior to it." Techno-philosophy is "deeply rooted in the "Heideggerian" definition of technology as "a technique of disclosing the universe, a revelation in which people seize authority over reality" and Contextualism "claims that our acts, utterances, expressions, and learning can only be understood in the context in which they occur" [28, p. 2].

7.3.3 Culturally Responsive Pedagogy. Culturally Responsive Pedagogy refers to a student-centered approach to teaching which recognises the importance of the students' cultural backgrounds and experiences in all aspects of learning [82]. It involves incorporating references to students' culture in learning and bringing it to the fore of discussion. It aims to promote engagement and achievement by creating inclusive and equitable learning environments where students can feel valued for who they are and connected to their cultural identities. Waite et al. [189] proposed ten reflective prompts for K–12 computing educators in England to enhance cultural responsiveness in their teaching. These prompts cover understanding learners, reflecting on cultural perspectives, incorporating relevant content, using familiar contexts, ensuring accessibility, offering open-ended activities, fostering collaboration, promoting student choice, reviewing materials and environment, and examining school policies. The study suggests these reflective prompts could benefit educators globally, encouraging further research and discussion on cultural integration in computer science education.

7.3.4 Culturally Relevant Pedagogy. The term "culturally relevant pedagogy" is defined as "a commitment to recognising the importance of learners' cultural backgrounds and encouraging teachers to create relevant learning experiences." [90, p. 29]. It was coined in the mid-1990s by Dr. Gloria Ladson-Billings as an approach to teaching that emphasises students' cultural context [103]. It seeks to use the student's background and lived experiences as a foundation to enhance their learning. It focuses on academic success, cultural competence, and sociopolitical consciousness. It differs from culturally responsive pedagogy in that cultural responsiveness focuses on dialectic engagement with students and constantly responding to their needs and culture. Hall et al. [90] describe how a culturally relevant approach was used in an introductory programming course taught to 16-to-25 year-olds in a refugee camp in Kenya.

This overview from existing research in Africa and globally underscores the critical importance of culturally responsive approaches in computer science education to enhance engagement, learning outcomes, and inclusivity, particularly for under-represented groups. Our research extends this foundation by concentrating on developing culturally contextualised materials for CS1 courses at the university level in Africa through a systematic approach.

In order to better ground our work in the context of African computing education, we will now turn to Banks' Approach to Multicultural Education Reform. As highlighted in the previous sections, a variety of frameworks, including culturally responsive and relevant pedagogies, offer useful strategies for integrating cultural context into computing education. However, Banks' comprehensive framework provides an explicit and structured way of understanding how multicultural approaches can be systematised into education reform. This is particularly useful as we move toward our own contextualisation framework, which is grounded in the African context. The next section presents the Banks' approach.

7.4 Banks' Approaches to Multicultural Education Reform

Banks advocates for multicultural education, but also highlights that it is a complex and multi-dimensional process [35]. One particular challenge highlighted is that, "many school and university practitioners have a limited conception of multicultural education, viewing it primarily as curriculum reform that involves only changing or restructuring the curriculum to include content about ethnic groups, women, and other cultural groups". Banks posits that this does not constitute multicultural education, arguing that the field is not understood and requires more formal conceptualisation.

Banks conceptualises multicultural education as a field with five dimensions:

- **Content Integration:** Focusing on integrating diverse perspectives, cultures, and histories into the curriculum.
- **Knowledge Construction:** Examining how knowledge is produced, validated, and transmitted within different cultural contexts.
- **Prejudice Reduction:** Addressing biases, stereotypes, and discriminatory attitudes.
- **Equity Pedagogy:** Implementing teaching methods that promote fairness and equal opportunities for all students.
- **Empowering School Culture:** Creating an environment that empowers students and challenges existing power dynamics

Banks describes four levels of approach to integrating cultural content into education:

Level 1: *The Contributions Approach* focuses on heroes, holidays and other cultural elements.

Level 2: *The Additive Approach* involves teachers appending cultural content, themes and perspectives to the existing curriculum.

Level 3: *The Transformation Approach* is when the curriculum is changed to incorporate concepts, issues, events and themes from the perspective of cultural groups, rather than simply appending additional content.

Level 4: *The Social Action Approach* is when the transformation approach is extended to give students agency to decide which social issues are important and equips them to take action to help solve them.

Building on Banks' framework, we can now examine how these levels can be applied within the context of developing contextualised materials for computing education in Africa. The Additive Approach, which we identified as most applicable for our work, provides a structured pathway for integrating cultural content into the curriculum. The Additive Approach will allow for the integration of cultural content into the existing curriculum without fundamentally altering it. The next section will detail the methods used to design and implement these materials within the framework of the Design Science Research (DSR) approach, outlining our process from problem explication to evaluation.

8 Contextual Materials: Methods

This study employed a design science research (DSR) approach. The DSR framework supports researchers in producing both relevant and rigorous research. According to Johannesson and Perjons [98], DSR involves five different phases, which include problem explication, requirements definition, design and development, demonstration, and evaluation.

In the problem explication phase, we built on the results of the exploratory literature review described earlier, particularly focusing on region-specific approaches to contextualisation (Section 7.2). Then (Section 7.1), we extended the scope by highlighting global examples of contextualisation, drawing on various theories and frameworks from outside Africa, thereby broadening the research base for developing contextualised materials. The findings from these reviews provided a solid foundation and strengthened the motivation for our research.

For the requirement definition phase, we investigated the literature, including anecdotal evidence and the authors' personal reflections, to examine the requirements for developing contextualised materials for learning programming. We reviewed the CS2023 report, drawing from both literature and personal teaching experiences to identify the most relevant knowledge units for contextualisation in CS1 (Section 8.1). Additionally we incorporated both literature and personal reflections to identify cultural themes (Section 8.2). Lastly, we joined the cultural themes and the topics from CS1 using Banks's Additive Approach (Section 8.3). Using this approach, we merged these cultural themes with the identified CS1 topics, enhancing the curriculum by adding culturally relevant content without altering its foundational structure.

The factors identified in the requirement-gathering stage provided ideas for the design and development of the contextualised learning materials for CS1. We designed the contextualised materials, each inspired by African cultural elements and aligned with key learning outcomes and software development fundamentals (Section 8.4). The next step was to implement and pilot them, which is the rationale for the demonstration and evaluation phase. We performed an initial evaluation by sharing the materials with a few faculty members in computing programs to understand their perspectives on the relevance of the materials for their own teaching and their students' learning (Section 8.5).

8.1 CS1 Topics and Learning Outcomes

In this work, we use *Computer Science Curricula 2023 - The Final Report* (CS2023) [47], the latest version of computer science curricular guidelines produced by a joint task force of the ACM, IEEE Computer Society and AAAI, as a basis to define common content in the foundational course in the Computer Science undergraduate curriculum, commonly described as "CS1". We chose this curricular framework due to its comprehensive and adaptable structure and the fact that it is the de-facto standard for Computer Science curricula. We then utilise Bank's Additive Approach to multicultural education reform [35], in which teachers append cultural content, themes, and perspectives to existing curriculum.

The CS2023 final report identifies the knowledge model of a Computer Science curriculum as consisting of 17 Knowledge Areas (KAs), each of which is made up of a collection of Knowledge Units (KUs). Each knowledge unit is a set of topics (some core, some elective) and learning outcomes for these topics. Although the report does not prescribe a specific packaging of these knowledge areas and units into courses in the curriculum, it does state that the

Software Development Fundamentals (SDF) KA "will generally be covered in introductory courses, often called CS1 and CS2". In our work, we chose to focus on the first six of the twelve knowledge units in the Software Development Fundamentals knowledge area, listed below:

- SDF-KU1: Basic concepts such as variables, primitive data types, expressions and their evaluation.
- SDF-KU2: How imperative programs work: state and state transitions on execution of statements, flow of control.
- SDF-KU3: Basic constructs such as assignment statements, conditional and iterative statements, basic I/O.
- SDF-KU4: Key modularity constructs such as functions (and methods and classes, if supported in the language) and related concepts like parameter passing, scope, abstraction, data encapsulation.
- SDF-KU5: Input and output using files and APIs.
- SDF-KU6: Structured data types available in the chosen programming language like sequences (e.g., arrays, lists), associative containers (e.g., dictionaries, maps), others (e.g., sets, tuples) and when and how to use them.

The CS2023 report identifies 14 illustrative learning outcomes for the knowledge units in the SDF knowledge area. Nine of these learning outcomes, which are listed below, correspond to the knowledge units we focus on in this work.

- SDF-LO1: Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.
- SDF-LO2: Develop programs using functions with parameter passing.
- SDF-LO3: Develop programs that effectively use the different structured data types provided in the language like arrays/lists, dictionaries, and sets.
- SDF-LO4: Develop programs that use file I/O to provide data persistence across multiple executions.
- SDF-LO7: Develop programs that create simple classes and instantiate objects of those classes (if supported by the language).
- SDF-LO11: Read a given program and explain what it does.
- SDF-LO12: Write comments for a program or a module specifying what it does.
- SDF-LO13: Trace the flow of control during the execution of a program.
- SDF-LO14: Use appropriate terminology to identify elements of a program (e.g., identifier, operator, operand)

8.2 Cultural Theme Identification

A key step in the process of developing contextually relevant computing education materials is identifying elements of culture and context that can be reflected in the materials. While programming activities often require addressing multiple concepts, as noted in [118], our approach goes beyond this by embedding these concepts within scenarios that reflect African cultural contexts.

To ensure cultural relevance, we define culture broadly, encompassing not only traditional African cultural practices and norms but also modern and popular culture and ways of life. Drawing from personal experience of life in Africa as well as reference books regarding African culture [7, 27, 59, 74, 151, 172], the team identified the following non-exclusive list of cultural elements to consider in developing materials: architecture, art, cuisine, dance, dress, family, gender roles, games, governance, housing, lifestyles, literature, marriage, media, music, religion, social customs, sport, and technology.

These cultural elements are intentionally integrated into the design of programming activities. For example, a task might involve simulating traditional African board games, analysing data related to local music trends, or designing algorithms to generate clothing patterns in specific regions. This deliberate integration connects compound programming concepts to relatable and meaningful cultural contexts, ensuring that students engage with material that resonates with their identities and lived experiences.

By situating programming tasks within African cultural contexts, we aim to enhance the relevance and accessibility of computing education. This approach differs from typical assessments, where multiple concepts may be present but lack contextual grounding, as highlighted in [118]. Instead, the cultural framing becomes a pedagogical tool to motivate and inspire students, fostering both technical and cultural engagement.

8.3 Applying Banks' Additive Approach

In Banks' Additive Approach, teachers append cultural content, themes, and perspectives to the existing curriculum. In our work, this involved creating a matrix that joined cultural themes with topics from the CS1 knowledge units. A brainstorming process then generated multiple examples, each one of which illustrated a particular CS1 topic using a given cultural element. Here are a few of such examples:

- The problem of determining which year the African Cup of Nations tournament is held (cultural element "Sports") can be used to illustrate the topic of boolean expressions and conditional statements in SDF-KU1 and SDF-KU2.
- In Egypt, a list of different types of traditional clothing (cultural element "Dress") can be used to illustrate the topic of iteration and built-in data structures in SDF-KU3 and SDF-KU6.
- In Nigeria, call-and-response storytelling (cultural element "Literature") can be used to illustrate the topic of functions in SDF-KU4.
- In Ghana, the Oware game (cultural element "Games") can be used to illustrate the topic of classes and objects in SDF-KU4.

It should be noted that although we have adopted Banks' Additive Approach to multicultural education reform as a theoretical framework for this work, this process of joining cultural themes with topics from CS1 fits naturally into other frameworks for contextualising education. For example, it is in harmony with the Culturo-Techno-Contextual Approach [28], in which students are asked to research, reflect on, and share indigenous knowledge or cultural practices and beliefs associated with a topic or concept. The instructor also shares his/her indigenous knowledge and cultural practices associated with the topic, and the instructor also draws examples from the immediate surroundings or local context.

8.4 Development of Contextualised Materials

Of all the educational materials that could be developed, we focused on developing programming activities that could be used as in-class examples, case studies or programming assignments in the context of a CS1 class. We developed an initial set of 25 activities[3]. We acknowledge that certain activities may be more easily adapted to other cultural contexts, while others,require deeper cultural knowledge for effective implementation.

Each activity developed from an example from the matrix described in the previous section: that is, it was inspired by a cultural element from one or more African countries, and addresses a topic from the six knowledge units identified from the Software Development Fundamentals (SDF) knowledge area of CS2023. The activities are designed to be relevant and engaging for students, providing practical examples of how cultural contexts can be integrated into programming education. Each activity comprises the following information:

- The CS1 Learning Outcomes (LOs) covered (as described in the ACM computing curriculum guidelines)
- The primary CS concepts being taught (e.g., iteration, functions with parameters, file I/O, etc.)
- The type of problem (e.g., debugging, code tracing, problem-solving, etc.)
- The cultural elements involved (e.g., dress, cuisine, lifestyle, etc.)
- The problem description
- Where appropriate and convenient, a model solution to the problem.

Section 9 describes the initial activities developed in greater detail.

8.5 Pilot Testing

8.5.1 Data Collection. In order to gather feedback on the designed contextualised materials, we developed a survey instrument using Google Docs and obtained an Institutional Review Board (IRB) exemption from the ethical review for the study, as no personally identifiable information was collected from participants. The full survey can be found in Figure 3.

Purposive sampling was employed across six (6) countries to select at least two (2) academics per country who have experience teaching the foundational programming course at the university level. We distributed the survey and the initial catalogue of developed programming activities via email.

8.5.2 Data Analysis. A reflexive thematic analysis approach, as outlined by Braun and Clarke [41], was employed to analyse and interpret themes within the survey data, which were imported into Excel for detailed coding. Reflexive thematic analysis emphasises the researcher's active role in theme development and is well-suited for qualitative research requiring flexibility and depth. The coding process followed a rigorous, iterative approach in three phases. Initially, two authors independently familiarised themselves with the data and coded it based on the research questions, ensuring a nuanced understanding of the dataset. They then engaged in collaborative discussions to resolve discrepancies and refine initial codes, in line with best practices for reflexive thematic analysis.

[3]The reader is welcome to contact one of the authors to gain access to these activities

(1) Do you think there is a need to develop computing education materials that are contextually relevant to Africa? Why or why not?
(2) Having reviewed the sample materials that have been created by our team so far, do you think they will be useful for your class? Why or why not?
(3) What are some of the strengths of the materials that have been created so far?
(4) What are some of the shortcomings or weaknesses of the materials created so far? How can they be improved?
(5) What programming language is used in the "CS1" course that you teach?
(6) Which of the following learning outcomes (as described in the ACM computing curriculum guidelines) are addressed by your course? (Note that it is not expected that all learning outcomes below are addressed). [Options: "Not Addressed", "Partially Addressed" or "Fully Addressed".]
  (a) Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.
  (b) Develop programs using functions with parameter passing.
  (c) Develop programs that effectively use the different structured data types provided in the language like arrays/lists, dictionaries, and sets.
  (d) Develop programs that use file I/O to provide data persistence across multiple executions.
  (e) Develop programs that create simple classes and instantiate objects of those classes (if supported by the language).
  (f) Read a given program and explain what it does.
  (g) Write comments for a program or a module specifying what it does.
  (h) Trace the flow of control during the execution of a program.
  (i) Use appropriate terminology to identify elements of a program (e.g., identifier, operator, operand)
  (j) Write programs that work with text by using string processing capabilities provided by the language.
  (k) Develop tests for modules, and apply a variety of strategies to design test cases.
  (l) Explain some limitations of testing programs.
  (m) Build, execute and debug programs using a modern IDE and associated tools such as visual debuggers.
  (n) Apply basic programming style guidelines to aid readability of programs such as comments, indentation, proper naming of variables, etc.
  (o) Write specifications of a module as module comment describing its functionality.
(7) In what country is your university located?
(8) Is your institution public or private?
(9) Do you have students from African countries other than the one in which your university is located? (i.e. international students from other African countries)
(10) If you have international students in your course, what are some of the countries represented?
(11) In what degree programmes are the students who take your class enrolled? (select all that apply) [Options: "Computer Science", "Information Systems", "Information Technology", "Computer Engineering" and "Other"]
(12) In what year of university are the majority of students who take your class? [Options: "Year 1", "Year 2" or "Other"]
(13) Additional general comments about the materials, the project, this survey, or your class.

Figure 3: Survey questions

Emerging themes were iteratively refined to ensure they captured the core issues, with sub-themes supported by illustrative quotes selected to represent key patterns in the data. The final step involved synthesizing the codes and themes, avoiding overlap between research questions, and identifying overarching themes, which were critically examined and linked to the paper's conclusion for a cohesive discussion. This approach aligns with the quality practices for reflexive thematic analysis, as detailed in Clarke and Braun's guidelines [51].

9 Contextual Materials: Example Activities

The material development process, introduced in Section 8.4, was designed to capture specific learning outcomes, cover various computing concepts, highlight cultural elements and reflect the author's nationalities. In this section, we include six example activities and follow this in Section 10 with a discussion of the set of activities created.

The following example activities were developed by the authors, one example per country. We have aimed to give example activities which provide broad coverage of the various CS1 concepts covered, exercise types, and cultural components included. The examples presented here do not necessarily provide a complete representation of the work conducted, so readers should refer to the initial set of 25 activities for a better understanding, which can be found here: https://doi.org/10.5281/zenodo.14238903

9.1 Example 1: Deconstructing Poetry (Botswana)

CS1 LOs:

- Develop programs that effectively use the different structured data types provided in the language like arrays/lists, dictionaries, and sets.
- Read a given program and explain what it does.

CS Concepts: Iteration, built-in data structures (list)
Problem Type: Code comprehension
Cultural Elements: Media (Poems)
Description:

Explain what each line of code does in the program below. What is the output of the overall program? How can you modify it to print only the first words: `Naledi e le`?

```
leboko = [
    "Naledi e le,",
    "Ya mariberibe, ribela ka pele,",
    "Re ye go nwa metsi,",
    "Metsi ga a yo,",
    "A nolwe ke Kgaupe,",
    "Kgaupe ga ke morate,",
    "Ke rata Masilonyana."
]

for line in leboko:
    print(line)
```

9.2 Example 2: Trace the output of Koshari ingredients (Egypt)

CS1 LOs:

- Trace the flow of control during the execution of a program.

CS Concepts: Iteration, built-in data structures (list)
Problem Type: Code tracing
Cultural Elements: Cuisine
Description:

Koshari is a traditional Egyptian dish combining rice, lentils, pasta and other ingredients. You can find the recipe and a photograph at this link: https://www.taste.com.au/recipes/koshari-egyptian-rice-lentils-pasta-recipe/ygqlkvwa.

Trace the following code and write down the output:

```
# List of ingredients in Koshari
koshari_ingredients = ["Rice", "Lentils", "Chickpeas", "
    Pasta", "Onions", "Garlic", "Tomato Sauce", "Vinegar
    ", "Spices (cumin, coriander)", "Fried Onions (
    optional topping)"]


# Iterating through the list of Koshari ingredients
print("Ingredients for making Koshari:")
for ingredient in koshari_ingredients:
    print("- " + ingredient)
```

9.3 Example 3: Calculating Tin Cans for a Pyramid in *Three Toti* (South Africa)

CS1 LOs:

- Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.

CS Concepts: Iteration, built-in data structures (list)
Problem Type: Programming problem-solving
Cultural Elements: Game
Description:

You and your friends would like to play a game of *three toti*. They are struggling to determine how many tin cans they require to build a pyramid of tin cans (a representation of the pyramid can be found in Figure 4). Write an algorithm, in the language of your choice, to determine how many tin cans are needed to stack on top of one another when the base number of tin cans are provided. The following provides on illustration of how the tin cans are stacked.
Model Solution (Python):

```
base = 5
line = base
sum = base

while line > 1:
    line = line - 1
    sum = sum + line
print(sum)
```

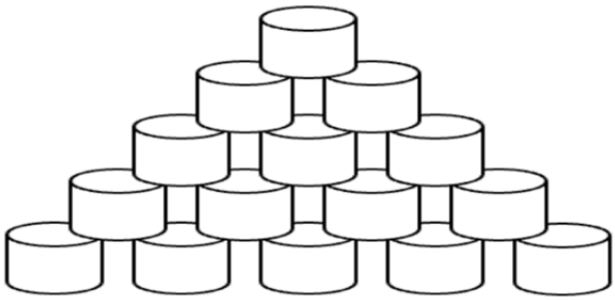

Figure 4: A *three toti* pyramid representation

9.4 Example 4: Is AFCON this year? (Ghana)

CS1 LOs:

- Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.
- Develop programs using functions with parameter passing.

CS Concepts: Functions with parameters, conditionals
Problem Type: Programming problem-solving
Cultural Elements: Game
Description:

The African Cup of Nations (AFCON) is the main international men's association football competition in Africa[4]. It was first held in 1957, and was subsequently held in 1959, 1962, 1963, 1965, and 1968. Since 1968, it has been held every two years, switching to odd-numbered years in 2013. (So, the tournament was held in both 2012 and 2013.)

Write a function that takes the year as a parameter and returns a boolean indicating whether or not the African Cup of Nations was/will be held in that year.

Model Solution (Java):

```java
public static boolean isAFCONYear(int year){
    if (year >= 2013 && year%2 == 1)
        return true;
    else if (year >= 1968 && year%2 == 0)
        return true;
    else if (year == 1957 || year == 1959 || year == 1962
      || year == 1963 || year == 1965)
        return true;
    else
        return false;
}
```

9.5 Example 5: Royal Greeting Merge (Nigeria)

CS1 LOs:

- Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.
- Develop programs using functions with parameter passing.
- Develop programs that use file I/O to provide data persistence across multiple executions.

CS Concepts: Iteration, conditionals, file I/O, error handling
Problem Type: Programming problem-solving
Cultural Elements: Government, Social Customs, Royalty Poetry
Description:

An Oriki is a form of praise poetry in Yoruba culture, often used to honour individuals, deities, or towns. These poems celebrate the subject's achievements, qualities, and lineage. They are traditionally recited during various ceremonies and significant events, using a call-and-response structure to engage the audience and enhance the communal experience.

You are tasked with creating a program that reads a call-and-response Oriki from two separate files, merges them into a single file, and adds the labels "Call" and "Response" before each line as necessary. There should be an equal number of calls and responses.

call.txt

```
The king with authority, second only to the gods!
The king who lives in a palace with an indigo roof!
The glorious king who resides in his city!
```

response.txt

```
Authority, second only to the gods!
Palace with an indigo roof!
The king who resides in his city!
```

Model Solution (Python):

```python
def read_lines(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        return [line.strip() for line in file.readlines()
    ]


def save_oriki(file_path, oriki_lines):
    with open(file_path, 'w', encoding='utf-8') as file:
        for line in oriki_lines:
            file.write(line + '\n')


def merge_call_response(calls, responses):
    if len(calls) != len(responses):
        raise ValueError("The number of calls and
     responses must be equal.")

    merged_oriki = []
    for call, response in zip(calls, responses):
        merged_oriki.append(f"Call: {call}")
        merged_oriki.append(f"Response: {response}")
    return merged_oriki


def main():
    # Step 1: Read calls and responses from files
    calls = read_lines('call.txt')
    responses = read_lines('response.txt')


    # Step 2: Merge calls and responses with labels
    merged_oriki = merge_call_response(calls, responses)


    # Step 3: Save the merged Oriki to a file
    save_oriki('oriki_output.txt', merged_oriki)


if __name__ == "__main__":
    main()
```

[4] en.wikipedia.org/wiki/Africa_Cup_of_Nations

9.6 Example 6: Design a *citenge* with Turtle graphics (Zambia)

CS1 LOs:

- Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements.
- Develop programs using functions with parameter passing.

CS Concepts: Iteration, graphics module/library
Problem Type: Programming problem-solving
Cultural Elements: Dress
Description:

A *citenge* is a garment in the form of a wide cloth with bright, repetitive patterns. Using the turtle graphics module, write a program to design a *citenge*. Your program should:

- Colour the graphics background with an appropriate colour
- Include a design of reasonable complexity
- Repeat the design several times across the *citenge* with a loop

You should consider writing a function to draw the design, given the coordinates to place it. Then you should consider how to loop your program so that it can draw the design in several "rows" and "columns" of the *citenge*.

Figure 5: A sample of a *citenge*. (Photo of a citenge belonging to the family of one of the authors.)

10 Activities Analysis

Once the activities were developed, they were compiled into a catalogue and were reviewed for their coverage of the CS1 learning outcomes, computing concepts and cultural elements. One of our primary goals was to explore the rich diversity of cultural concepts that could be infused into the teaching of fundamental programming concepts students in CS1. While the catalogue developed is not necessarily comprehensive, it shows some of the many possibilities for incorporating contextually relevant concepts when teaching programming to African students.

Figure 6 is an alluvial diagram which depicts the programming concepts, cultural elements, and learning outcomes focused on by the activities. While there were 25 activities, some assessed multiple learning outcomes and involved various concepts. Therefore, each link in the alluvial diagram represents one relationship; it does not depict 25 distinct activities but rather the variety of relationships among them. For instance, there was a question focused on the cultural aspect of dance from Botswana, which aligned with the first learning outcome. This question included concepts of iteration, conditionals, and built-in data structures. As shown, it has three links entering the dance node.

A number of activities used multiple concepts such as built-in data structures (e.g., lists, arrays, dictionaries), iteration, and conditionals. From a pedagogical standpoint, this is worth discussing. When students learn how to program, they typically learn these concepts in isolation, so having more straightforward or more basic questions could be used at the early stages of learning. An alternative to looking at these examples is that since they include multiple concepts, they may be used in their current configuration when students have already been exposed to all of the relevant concepts for a more summative activity. This raises the question of when it is best to use culturally relevant materials: when students are first exposed to concepts, or after mastering those concepts.

From the perspective of cultural elements, four out of the six authors included activities relating to games. Games are pervasive in any culture, but from a pedagogical standpoint, two things come to mind. The first is that instructors may tend to use games because they have set rules that are suitable for programming that are typically fixed. Second, games benefit students because they are fun and can serve as a form of intrinsic motivation and engagement to create a game and then share it with their peers.

The activities were mainly developed independently by each author, so we identified areas for improvement when they were aggregated. As depicted in Figure 6, there tends to be a preference for focusing on the first learning outcome. In addition, most of the activities tended to involve programming problem-solving activities. These activities typically include describing a scenario and a problem, with the student expected to write a programmatic solution. Future work should include more ways for students to engage with code, for example, code tracing, code reading, testing, and debugging exercises.

It is worth noting that though a given author developed an activity designed for their country, that activity may be applicable across countries due to shared cultural elements and values common across many African nations. Example 4 (Ghana, 9.4) is an example of this, where the activity was originally designed for the Ghanaian context, but could be applicable to any African context. Another example would be Example 6 (Zambia, 9.6) which describes an airtime credit system which is not unique to Zambia and can be found in many African countries. Additionally, there are several instances of similar but not identical scenarios in the activities which could be adapted to a different context by only changing a few words. For example, one of the exercises from Ghana in the activity catalogue involves modelling a "tro-tro" system, which

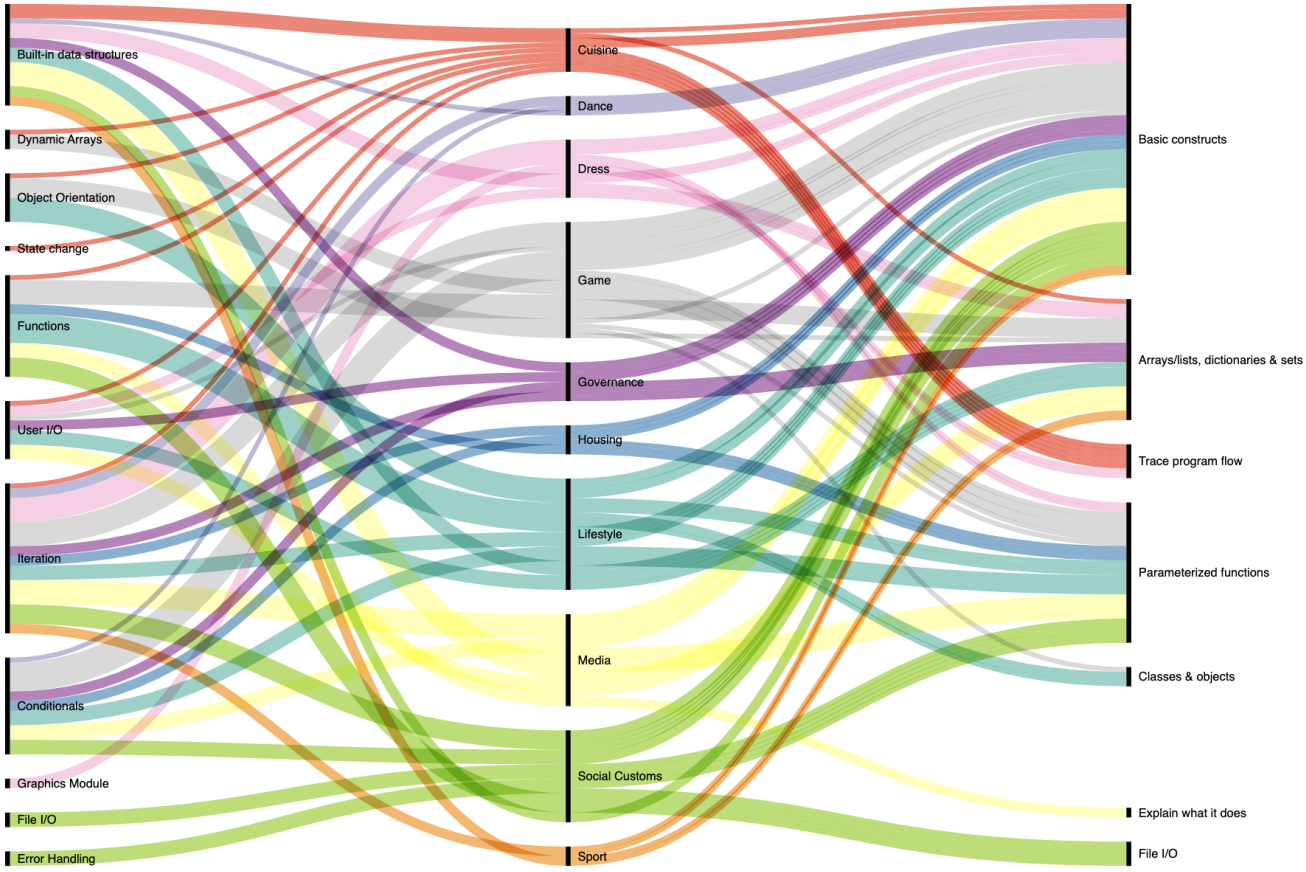

Figure 6: Alluvial diagram depicting the categorisation of the development materials with CS concepts covered on the left, cultural elements in the centre and the CS1 learning outcomes on the right

is a term for a minibus in Ghana; other African countries have different names for a very similar minibus system (e.g., "DalaDala" in Tanzania or "matatus" in Kenya, or simply "minibus" in Zambia) so the activity could easily be modified for different contexts.

11 Contextual Materials: Survey Results

We sent the activity catalogue and survey to 26 faculty teaching at 15 institutions across 6 countries. Thirteen (13) faculty members reviewed our activities. We removed one faculty member who did not meet the criteria for how we defined CS1 in Section 8, leaving us with 12 faculty members who taught CS1 in Africa. The countries the faculty members come from were Botswana (2), Egypt (1), Ethiopia (1), Ghana (4), Nigeria (2), South Africa (1), and Zambia (1). The respondents represent eight institutions, of which three were private, and five were public. Finally, ten out of the twelve participants informed us about having international students (primarily from other African countries) in their CS1 classes.

11.1 Qualitative Analysis

Recall the open-ended survey questions sent out to the faculty at institutions in Africa:

- Having reviewed the sample materials created by our team so far, do you think they will be useful for your class? Why or why not?
- What are some strengths of the materials created so far?
- Are some shortcomings or weaknesses of the materials created so far? How can they be improved?

This section explores the responses to these questions through thematic analysis, the process for which is described in Section 8.4.

The respondents are referred to as Faculty *n*, where *n* is a random number used to distinguish them. Since there were originally 13 responses and one was removed, these numbers range from 1 to 13.

11.1.1 Perceived need for materials. Most faculty saw the usefulness of computing education materials that resonate with students' backgrounds and experiences. Our survey responses emphasised the importance of student relatability and familiarity in understanding computer programming. Faculty members who responded to

the survey stressed the necessity of connecting learning to daily life through local examples, stories, and challenges specific to African industries and cultures. As Faculty 2 stated:

> *"Students learn better with materials they can relate to... what they interact with on a daily basis."*

Faculty 9 added that incorporating

> *"...such materials can engage students by addressing local challenges and using familiar examples, which can improve understanding and make learning more relatable and effective."*

We note that the benefits of our contextualised materials extend beyond engagement. By incorporating culturally relevant elements into materials faculty use to teach students, there is the chance to increase student interest in innovation in fixing problems within their 'home' communities or neighbourhoods. Faculty 9 echoed this sentiment:

> *"Such materials can engage students by addressing local challenges and using familiar examples, which can improve understanding and make learning more relatable and effective. It could also inspire students to use programming to solve community problems they may not have previously considered."*

This shift empowers students to see themselves as change-makers, equipped with the skills and confidence to contribute innovative solutions to their communities' specific challenges.

Choosing the correct context for these contextually relevant materials is important, particularly because our faculty expressed unique computational challenges to African classrooms, such as the '*numerous languages*' and the '*lack of computational tools for learners.*' By ensuring the chosen analogies and scenarios truly resonate with students, the faculty believes that contextually relevant materials can bridge the digital divide, promote inclusivity in education, and encourage students to tackle Africa's unique challenges with a passion for innovation and problem-solving.

11.1.2 Perceived usefulness of materials. Our survey, given to faculty members, indicated that using contextually relevant materials in CS1 courses can potentially spark enthusiasm, active learning, student motivation, and engagement. For example, Faculty 1 stated that *"relatable examples will generate excitement and push students to seek solutions, despite the solution being difficult to solve."* Similarly, Faculty 4 shared *"As someone who lives in Ghana, seeing an activity titled 'how to model a tro-tro?' automatically sparks [my] curiosity. [The specific activity local to our context] will be a lot of fun for students taking an introductory course in object-oriented programming."*

However, a concern by the faculty was that the use of 'overly specific' examples tied to a single context (e.g., country) might limit students' interest and engagement when context-specific examples are used in situations in which the student is unfamiliar, for example, in multinational classrooms. As Faculty 12, Faculty 13, and Faculty 3 stated:

> *"I think they will be useful, at least or especially to some of the students familiar with those contextual elements."*
>
> *"the modern average student is a global citizen... if we over-contextualise, it may again work against the purpose of the contextualisation."*

Participants proposed valuable solutions to this challenge, suggesting including diverse global challenges or interdisciplinary current problems in course materials. Faculty 12 and Faculty 13 noted:

> *"it would be beneficial to also include global challenges. Technology keeps evolving, and our learners need to understand various levels of problem-solving to stay up-to-date and competitive."*
>
> *"but it may also be important to infuse [the contextualised materials] with geographic context samples from different other science/art domains to truly expose students to the importance of computing".*

Focusing on broadly applicable scenarios could help students develop transferable problem-solving skills, preparing them for the ever-changing world of technology, where solutions often need adaptation to new contexts. For instance, a student who masters a program to optimise tro-tro routes can apply those core programming concepts to design algorithms for optimising delivery routes in any city worldwide.

11.1.3 Our identified strengths of materials. Our study identified a few strengths of the educational materials we showed to our faculty participants. One strength was the use of contextually relevant materials that leverage students' existing knowledge of local entertainment (including music and games). Faculty members informed us that providing contextually relevant and specific examples would set a strong foundation for learning, enhance student engagement, and also reduce teacher preparation time.

For example, Faculty 2 noted:

> *"The strength is that the examples are relevant to our local context. They use entertainment like music and games, which helps make the technical concepts I teach more relatable for students."*

Faculty 8 also highlighted the variety and practical applicability of the questions:

> *"The mix of questions effectively tests different aspects of the material. These questions can be used for both practical and written assessments. Additionally, the diverse examples from across the continent ensure all students learn about different African cultures."*

Another strength is the inclusion of a clear and concise preamble that defines the key concept being taught, as Faculty 5 observed:

> *"The preamble to the question helps students understand what they are learning by explicitly defining the key concept."*

Finally, focusing on readily available and low-cost activities (e.g., chess [Faculty 2]) can promote inclusivity by removing financial barriers for both faculty and students. These culturally relevant activities can ignite students' curiosity about their heritage, deepen their appreciation for their own and other cultures, and foster a pan-African perspective that celebrates the continent's cultural richness.

11.1.4 Our identified shortcomings of materials. The feedback from our faculty revealed some limitations in the current approach of our contextually relevant materials. One issue raised was that the

materials raise the risk of alienating the "new generation" of students with activities grounded in practices that are fading due to *culture erosion* [188]. This calls for a nuanced approach, as Faculty 2 emphasised the importance of:

> *"...creating a balance in using traditional and cultural activities by also looking into not only local activities but modern activities and examples so the modern students don't still struggle to understand the cultural activity as well as the concept being taught."*

In summary, key themes that can be taken from the qualitative analysis are given in Table 4. These include the ones highlighted in Sections 11.1.1 to 11.1.4.

12 Contextual Materials: Threats to Validity and Limitations

The development of the contextual materials reported in the paper is by no means exhaustive. It is meant as a starting point for conversation and further development of materials for use in teaching CS1 courses.

This initial catalogue of contextual material activities is skewed towards the preferences of the researchers who developed them and does not cover all topics in a typical CS1 course. The catalogue is limited in the number of African counties represented, yet more extensive than many other reports developing materials for CS1 courses. This opens the opportunity for collaborations within focus groups working towards expanding both the content to fully represent the topics in a CS1 course and the diversity of activities for different countries on the African continent and beyond.

The sample of faculty chosen to evaluate the materials was purposefully chosen to reflect the African countries that are represented by the researchers who created the contextual materials. The sample is therefore limited. As the materials include additional countries and content, similar studies may be conducted to gauge the perceptions of the expanded catalogue of contextualised materials by faculty representing CS1 courses across more countries.

The current work is a preliminary exploration and does not include an evaluation of the developed materials in a classroom context and thus an investigation of the effect of the use of the materials on learning outcomes or student interest in computing. Prior work [115] has raised the question of under what circumstances contextualisation reduces or increases the cognitive load of students. This work does not attempt to answer this question.

13 Contextual Materials: Discussion

The process for developing contextually relevant materials for the African context, using Banks' Additive Approach , was as follows:

- The authors agreed on a subset of computing to focus on, in this case introductory programming
- We identified a curriculum to base the set of materials around, in this case the CC2023 curriculum
- We identified a broad range of cultural elements which could be incorporated into computing examples, or around which computing examples could be built
- We individually developed some example programming exercises which were categorised by the CS2023 learning outcomes and computing concepts they covered, as well as the cultural elements we included (from our own backgrounds, experiences and knowledge of our relevant cultures)
- We compiled these into a catalogue and established which learning outcomes, computing concepts and cultural elements had been covered
- We circulated the catalogue among an extended community of faculty based in institutions in Africa and collected feedback on the examples

We summarise this process here, with details of each step in previous sections, for any reader interested in developing their own contextually relevant materials to follow along. The collaborative nature of this project, involving faculty from various African countries, allowed the exploration of a wide range of cultural contexts. This diversity allowed for a rich tapestry of cultural elements to be integrated into the CS1 curriculum, making it more relevant and engaging for students. One challenge faced was ensuring that the culturally contextualised materials could be standardised and adopted across different educational institutions. Balancing the need for cultural specificity with the requirements of a standardised curriculum required careful consideration and iterative refinement. Additionally, the process of aligning cultural themes with programming concepts highlighted the need for flexibility and creativity in curriculum development. The iterative approach to developing and refining these materials based on educator feedback ensured that they remained relevant and effective over time. An ongoing evaluation and improvement process is essential for maintaining the impact and applicability of the materials.

Furthermore, this project emphasises the importance of including local faculty in the development process, which fosters a sense of ownership and empowerment among African faculty. This collaborative model positions them as leaders and innovators in computer science education. By promoting culturally relevant education, we support broader goals of inclusion and diversity in computer science. This initiative aligns with the global movement towards decolonising education and ensuring that academic content reflects and responds to learners' diverse cultures and experiences, both in computing education [183] and worldwide [40, 83, 100].

In an era where the digital divide continues to perpetuate inequalities, providing culturally relevant education is a crucial step toward ensuring that all students have the opportunity to succeed in the digital era. In particular, with the rise of LLMs and companies appropriating our data and labour [52], it is crucial for both students and faculty in developing nations to be more aware of how computing affects them and to develop proactive steps to address the encroachment on our liberties.

To further the success of integrating African cultural elements into CS1 courses, it is recommended that future initiatives continue to emphasise collaboration among faculty from diverse cultural backgrounds. Establishing networks and forums for sharing best practices and resources can enhance the quality and relevance of educational materials. Additionally, ongoing professional development for faculty in culturally responsive pedagogy is essential to implement and sustain these initiatives effectively. Institutions

Table 4: Key themes and quotes extracted from survey responses from instructors in Africa

Key Themes	Quotes
Cultural Relevance	"Aids in student learning. Contextually relevant analogies essential."
	"Students learn better if they are given the materials that they can relate to or what they interact with on a daily basis."
Engagement and Excitement	"Having these relatable examples will generate more excitement."
	"It will be a lot of fun for students taking an introductory course in OOP to think about the various instance variables and methods involved in designing the tro-tro class."
Strengths of Materials	"The examples are country-specific and use examples that the majority of local students would be able to relate with as local content."
	"The activities are very relevant to African settings by including local traditions, languages, and cultural practices in computing education."
Weaknesses of Materials	"Will prefer a more systematic approach - list out the topics in CS1 and associating problems."
	"With culture erosion we must be careful that some of the activities are not relatable to the new generation."
Suggestions for Improvement	"To improve, it would be good to add activities that cover both local and global issues in computing."
	"Providing different versions of activities or suggestions for places with fewer resources would make sure all students take part in learning."
Positive sentiments about the overall work	"This is a good study and with the common struggles that our students seem to experience, contextualisation may help address some of the ambiguity associated with the course materials and content."
	"The project is progressing positively and deserves full support because it is evidence-based and has the potential to bring about positive changes in the curriculum."

should consider adopting flexible curricular frameworks that allow for incorporating culturally relevant content while maintaining standardisation. This flexibility will enable faculty members to adapt materials to their specific cultural contexts without compromising the integrity of the curriculum.

In this work, we used Bank's multi-level approach to describe the ways of contextualising curriculum. While beneficial, we encourage others to explore how other frameworks can be used to support contextualisation. Within Banks, we have focused on creating materials at the second level, which is additive. This was because we intended to highlight the diversity across the content in a constrained time. A more robust approach should explore the Transformation and Social Action approach, which would require collaborating with faculty and students in their local context. We encourage faculty to consider which level is best suited for their context. To conduct work at the Transformational and Social Action levels, it is necessary to fully explore the local context and institutional factors, as well as have student participation and faculty support. While this may sound arduous, work at this level will better equip students to address the socio-technical challenges of our age.

Finally, continuous research and evaluation should be conducted to assess the impact of these contextualised materials on student engagement, retention, and learning outcomes. This data will be used to refine and improve the approach, ensuring that it remains effective and responsive to the needs of students. In conclusion, the integration of African cultural elements into CS1 courses represents a significant step towards making computer science education more inclusive and accessible. By leveraging the rich cultural heritage of African countries, faculty can create a more engaging and relevant learning experience for students, thereby broadening participation in the field of computer science.

14 Conclusions and Future Work

In this report, we share the first comprehensive literature review of computing education in Africa, with papers spanning the period from 1978 to March 2024. The work reviewed illustrates a rich conversation about computing education in several countries across Africa, encompassing topics such as programming education, the need for and approaches to contextualisation, AI and Robotics education, and the impact of and challenges with infrastructure. This body of work has been brought together for the first time in this report, serving as a baseline for future studies of computing education in African countries.

Recognising the importance of making education relevant to learners in their context, this report also presents a process for developing contextualised materials, using the foundational programming course, commonly referred to as CS1, as a case study. The result is an initial activity catalogue of programming exercises grounded in various elements of culture from six African countries, namely Botswana, Egypt, Ghana, Nigeria, South Africa, and Zambia. Initial feedback from a dozen instructors in these countries shows the value of the process that has been initiated and of the example materials that have been developed.

This report opens up many directions for future work. A review of the growing number of PhD theses on the topic of computing education in Africa would be of interest, for example [8, 43, 134, 140, 144, 152, 185]. The large number of venues represented in our dataset, plus the very small number of papers per venue, also raises

interesting questions. What form might computing education research communities take in this context? Another direction for future work is to look into each of the categories of papers discovered (see Table 3) and make an in-depth discussion and analysis. For example papers which discussed tools, using technology, teaching strategy, or curriculum for computing education can be further analysed to elucidate the current state of what is being done, how it is shaping computing education in Africa and what impact it can have on computing education globally. Investigating computing education research in other languages, particularly local African languages, would be another interesting area of future work.

Related to contextual materials, there is a significant amount of work that can be done. An in-depth study of the level of use of contextualised materials in African classrooms would be interesting. Materials could be developed for upper-level courses. More extensive contextualisations at higher Banks levels, involving whole curricula or pedagogies, and involving the students themselves in creating materials, have been begun in related work, as discussed in Sections 4 and 7, but could be expanded.

In the short run, further learning materials can be developed, building on this initial catalogue. Recognising that there are many cultural similarities across groups of African countries, the catalogue can also be enhanced with a guide to cross-cultural translation of the materials, helping instructors understand, for example, how to quickly convert a programming activity about food preparation in Nigeria to one that would work in the Ghanaian context. An important next step is a systematic study of the use of these contextualised materials in real classrooms. African instructors, particularly those who have not yet participated in computing education research, would be welcome to join in this effort. Finally, the hope is to build an online e-textbook for CS1 materials that would automatically adapt its examples to the country or culture selected by the instructor.

Acknowledgments

This material is partially based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2241144. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

We give our thanks to Gabriel Parriaux for his very helpful assistance in identifying French-language databases and to Robert McCartney for valuable feedback and comments throughout the project. Finally, we dedicate this report to the memory of our wonderful colleague and friend, Brett Becker, who worked with us on the project through the first draft and the start of the revisions.

References

[1] Yohannis Abate. 1978. African Population Growth and Politics. *Issue* 8, 4 (1978), 14–19. https://doi.org/10.2307/1166318

[2] Khadija Mansour Abuzagia. 2017. Cloud Computing Techniques: Strategies and Applications for Education. In *Joint International Conference on Information and Communication Technologies for Education and Training and International Conference on Computing in Arabic (ICCA-TICET)*. IEEE, Khartoum, Sudan, 1–14. https://doi.org/10.1109/icca-ticet.2017.8095301

[3] ACM. 2024. ACM Transactions on Computing Education: Scope. https://dl.acm.org/journal/toce/scope.

[4] ACM, IEEE Computer Society, and AAAI. 2023. CS2023 – the Final Report with Feedback. https://csed.acm.org/cs2023-report-with-feedback/.

[5] *Joel C. Adams, Vimala Bauer, and Shakuntala Baichoo. 2003. An Expanding Pipeline: Gender in Mauritius. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (Reno, NV, USA) *(SIGCSE '03,)*. ACM, New York, NY, USA, 59–63. https://doi.org/10.1145/611892.611932

[6] *Joseph Adigun, John Onihunwa, Eric Irunokhai, Yusuf Sada, and Olubunmi Adesina. 2015. Effect of Gender on Students' Academic Performance in Computer Studies in Secondary Schools in New Bussa, Borgu Local Government of Niger State. *Journal of Education and Practice* 6, 33 (2015), 1–7.

[7] Funso S. Afolayan. 2004. *Culture and Customs of South Africa*. Holtzbrinck, Westport, Conn.

[8] Friday Joseph Agbo. 2022. *Co-Designing a Smart Learning Environment to Facilitate Computational Thinking Education in the Nigerian Context*. Ph. D. Dissertation. Itä-Suomen yliopisto.

[9] Friday J. Agbo, Maria Ntinda, Sonsoles López-Pernas, Mohammed Saqr, and Mikko Apiola. 2023. Computing Education Research in the Global South. In *Past, Present and Future of Computing Education Research: A Global Perspective*. Springer, New York City, NY, USA, 311–333. https://doi.org/10.1007/978-3-031-25336-2_15

[10] *Friday J. Agbo, Linda O. Okpanachi, Patrick Ocheja, Solomon S. Oyelere, and Godwin Sani. 2024. How Can Unplugged Approach Facilitate Novice Students' Understanding of Computational Thinking? An Exploratory Study from a Nigerian University. *Thinking Skills and Creativity* 51 (2024), 16. https://doi.org/10.1016/j.tsc.2023.101458

[11] *Friday J. Agbo, Olayemi Olawumi, Oluwafemi S. Balogun, Ismaila T. Sanusi, Sunday A. Olaleye, Kissinger Sunday, Emmanuel A. Kolog, Donald D. Atsa'am, Frank Adusei-Mensah, Ayobami Adegbite, and Funmilola W. Ipeayeda. 2021. Investigating Students' Perception towards the Use of Social Media for Computing Education in Nigeria. *Journal of Information Systems Education* 32, 3 (2021), 212–228.

[12] *Friday J. Agbo, Olayemi Olawumi, Solomon S. Oyelere, Emmanuel A. Kolog, Sunday A. Olaleye, Richard O. Agjei, Dandison C. Ukpabi, Abdullahi A. Yunusa, Saheed A. Gbadegeshin, Luqman Awoniyi, Kehinde O. Erinle, Emmanuel Mogaji, Aziaka D. Silas, Chijioke E. Nwachukwu, and Adedayo Olawuni. 2020. Social Media Usage for Computing Education: The Effect of Tie Strength and Group Communication on Perceived Learning Outcome. *International Journal of Education and Development using Information and Communication Technology* 16, 1 (2020), 5–26.

[13] *Friday J. Agbo and Solomon S. Oyelere. 2019. Smart Mobile Learning Environment for Programming Education in Nigeria: Adaptivity and Context-Aware Features. *Advances in Intelligent Systems and Computing* 998 (2019), 1061–1077. https://doi.org/10.1007/978-3-030-22868-2_71

[14] *Friday J. Agbo, Solomon S. Oyelere, Jarkko Suhonen, and Markku Tukiainen. 2019. Identifying Potential Design Features of a Smart Learning Environment for Programming Education in Nigeria. *International Journal of Learning Technology* 14, 4 (Jan. 2019), 331–354. https://doi.org/10.1504/ijlt.2019.106551

[15] *Friday J. Agbo, Solomon S. Oyelere, Jarkko Suhonen, and Markku Tukiainen. 2023. Design, Development, and Evaluation of a Virtual Reality Game-Based Application to Support Computational Thinking. *Educational Technology Research and Development* 71, 2 (2023), 505–537. https://doi.org/10.1007/s11423-022-10161-5

[16] *Samia Ait-Adda, Nabila Bousbia, and Amar Balla. 2023. Enriching the Learner's Model through the Semantic Analysis of Learning Traces. *E-Learning and Digital Media* 20, 1 (2023), 1–24. https://doi.org/10.1177/20427530221102993

[17] *Ebenezer Anohah and Jarkko Suhonen. 2018. Conceptual Model of Generic Learning Design to Teach Cultural Artifacts in Computing Education: An Analysis Based on Akan Culture in Ghana. *International Journal of Online Pedagogy and Course Design* 8, 4 (Oct. 2018), 50–64. https://doi.org/10.4018/ijopcd.2018100104

[18] *Ebenezer Anohah and Jarkko Suhonen. 2019. Perceived Effectiveness of Students Programming Indigenous Symbols in Ghanaian Context. *International Journal of Learning Technology* 14, 3 (2019), 214–235. https://doi.org/10.1504/IJLT.2019.105708

[19] *J. A. Anyanwu. 1978. Computer Science Education in a Developing Nation. In *Papers of the 1978 SIGCSE/CSA Technical Symposium on Computer Science Education* (Detroit, MI, USA) *(SIGCSE '78)*. ACM, New York, NY, USA, 37–40. https://doi.org/10.1145/990555.990573

[20] *Mikko Apiola, Nella Moisseinen, and Matti Tedre. 2012. Results from an Action Research Approach for Designing CS1 Learning Environments in Tanzania. In *Proceedings of the 2012 Frontiers in Education Conference* (Seattle, WA, USA) *(FIE '12)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/fie.2012.6462321

[21] *Mikko Apiola, Jarkko Suhonen, Abbi Nangawe, and Erkki Sutinen. 2015. Building CS Research Capacity in Sub-Saharan Africa by Implementing a Doctoral Training Program. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (Kansas City, MO, USA) *(SIGCSE '15)*. ACM, New York, NY, USA, 633–638. https://doi.org/10.1145/2676723.2677242

[22] *Mikko Apiola and Matti Tedre. 2011. Towards a Contextualized Pedagogy for Programming Education in Tanzania. In *Proceedings of the 2011 AFRICON*

Conference (Victoria Falls, Zambia) *(AFRICON '11)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/afrcon.2011.6072010

[23] *Mikko Apiola and Matti Tedre. 2012. New Perspectives on the Pedagogy of Programming in a Developing Country Context. *Computer Science Education* 22, 3 (2012), 285–313. https://doi.org/10.1080/08993408.2012.726871

[24] *Mikko Apiola, Matti Tedre, Matti Lattu, and Tomi A. Pasanen. 2012. Towards a Framework for Designing and Analyzing CS Learning Environments. In *Proceedings of the 2012 Frontiers in Education Conference* (Seattle, WA, USA) *(FIE '12)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/fie.2012.6462430

[25] *Mikko Apiola, Matti Tedre, and Josephat O. Oroma. 2011. Improving Programming Education in Tanzania: Teachers' and Students' Perceptions. In *Proceedings of the 2011 Frontiers in Education Conference* (Rapid City, SD, USA) *(FIE '11)*. IEEE, New York, NY, USA, 1–7. https://doi.org/10.1109/fie.2011.6142787

[26] *Ian Arawjo, Ariam Mogos, Steven J. Jackson, Tapan Parikh, and Kentaro Toyama. 2019. Computing Education for Intercultural Learning: Lessons from the Nairobi Play Project. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (Nov. 2019), 1–24. https://doi.org/10.1145/3359154

[27] Molefi K. Asante. 2002. *Culture and Customs of Egypt*. Holtzbrinck, Westport, Conn.

[28] *Fred Awaah, Peter Okebukola, Juma Shabani, Daniel Solarin, and Ekwam E. Okyere. 2022. "I Am a Cultural Teaching Method – I Was Successful in the ICT Class in the Global South". *Cogent Education* 9, 1 (2022), 15. https://doi.org/10.1080/2331186X.2022.2134704

[29] *Yirsaw Ayalew, Ethel Tshukudu, and Moemedi Lefoane. 2018. Factors Affecting Programming Performance of First Year Students at a University in Botswana. *African Journal of Research in Mathematics, Science and Technology Education* 22, 3 (2018), 363–373. https://doi.org/10.1080/18117295.2018.1540169

[30] *Musa A. Ayanwale, Emmanuel K. Frimpong, Oluwaseyi A. G. Opesemowo, and Ismaila T. Sanusi. 2024. Exploring Factors That Support Pre-service Teachers' Engagement in Learning Artificial Intelligence. *Journal for STEM Education Research* 7, 2 (2024), 31. https://doi.org/10.1007/s41979-024-00121-4

[31] *Musa A. Ayanwale and Ismaila T. Sanusi. 2022. Perceptions of STEM vs. Non-STEM Teachers Toward Teaching Artificial Intelligence. In *Proceedings of the 2022 AFRICON Conference* (Nairobi, Kenya) *(AFRICON '22)*. IEEE, New York, NY, USA, 1–5. https://doi.org/10.1109/AFRICON55910.2023.10293455

[32] *Musa A. Ayanwale, Ismaila T. Sanusi, Owolabi P. Adelana, Kehinde D. Aruleba, and Solomon S. Oyelere. 2022. Teachers' Readiness and Intention to Teach Artificial Intelligence in Schools. *Computers and Education: Artificial Intelligence* 3 (2022), 11. https://doi.org/10.1016/j.caeai.2022.100099

[33] *Musa A. Ayanwale, Ismaila T. Sanusi, Rethabile R. Molefi, and Adekunle O. Otunla. 2023. A Structural Equation Approach and Modelling of Pre-Service Teachers' Perspectives of Cybersecurity Education. *Education and Information Technologies* 29, 3 (2023), 3699–3727. https://doi.org/10.1007/s10639-023-11973-5

[34] *Engineer Bainomugisha, Karen Bradshaw, Martin M. Ujakpa, Joyce Nakatumba-Nabende, Lawrence Nderu, Neema Mduma, Patrick Kihoza, and Annette Irungu. 2024. Computer Science Education in Selected Countries from Sub-Saharan Africa. *ACM Inroads* 15, 1 (2024), 65–82. https://doi.org/10.1145/3643037

[35] James A. Banks. 1993. Multicultural Education: Historical Development, Dimensions, and Practice. *Review of Research in Education* 19 (1993), 3. https://doi.org/10.2307/1167339

[36] *Bazara I. A. Barry. 2009. Using Open Source Software in Education in Developing Countries: The Sudan as an Example. In *Proceedings of the 2009 International Conference on Computational Intelligence and Software Engineering* (Wuhan, China). IEEE, New York, NY, USA, 1–4. https://doi.org/10.1109/cise.2009.5364872

[37] *Ilsa Basson. 2021. Twenty Years into the New Millennium: How Integrated Is Mathematics, Physics and Computer Science at Secondary School Level? *Perspectives in Education* 39, 4 (2021), 3–26. https://doi.org/10.18820/2519593X/pie.v39.i4.2

[38] *Tesfaye Bayu Bati, Helene Gelderblom, and Judy van Biljon. 2014. A Blended Learning Approach for Teaching Computer Programming: Design for Large Classes in Sub-Saharan Africa. *Computer Science Education* 24, 1 (2014), 71–99. https://doi.org/10.1080/08993408.2014.897850

[39] Dianne Biin and Marla L Weston. 2015. An Indigenous Learning Approach to Computer Science Education: 21st Century Skills for Middle and High School Aboriginal Children on British Columbia's West Coast. In *Emerging Technologies for STEAM Education*, Xun Ge, Dirk Ifenthaler, and J. Michael Spector (Eds.). Springer International Publishing, Cham, 95–112. https://doi.org/10.1007/978-3-319-02573-5_6

[40] Abeba Birhane and Olivia Guest. 2020. Towards Decolonising Computational Sciences. https://doi.org/10.48550/arXiv.2009.14258 arXiv:2009.14258 [cs]

[41] Virginia Braun and Victoria Clarke. 2021. One Size Fits All? What Counts as Quality Practice in (Reflexive) Thematic Analysis? *Qualitative Research in Psychology* 18, 3 (2021), 328–352.

[42] *Dane Brown and James Connan. 2021. A Robust Portable Environment for First-Year Computer Science Students. *Communications in Computer and Information Science* 1518 (2021), 88–103. https://doi.org/10.1007/978-3-030-92858-2_6

[43] Oladele Campbell. 2023. *Impact of Scratch on the Achievements of First-Year Computer Science Students in Programming in Some Nigerian Polytechnics*. Ph. D. Dissertation. University of South Africa.

[44] *Oladele O. Campbell, Oluwatoyin Adelakun-Adeyemo, Fatimah Y. Akinrinola, Patience C. Akih, Ethel Tshukudu, and Brett A. Becker. 2023. The Impacts of a Constructionist Scratch Programming Pedagogy on Student Achievement with a Focus on Gender. In *Proceedings of the 2023 Global Computing Education Conference* (Hyderabad, India) *(CompEd '23)*. ACM, New York, NY, USA, 29–35. https://doi.org/10.1145/3576882.3617911

[45] *Geraldo Cangondo, Nuno Pombo, Leonice Souza-Pereira, Sofia Ouhbi, and Bruno Silva. 2022. Computer Science Education in Angola: The Key Challenges. In *Proceedings of the 2022 Global Engineering Education Conference* (Tunis, Tunisia) *(EDUCON '22)*. IEEE, New York, NY, USA, 139–147. https://doi.org/10.1109/educon52537.2022.9766392

[46] *Juan M. Carrillo De Gea, Joaquín Nicolás, José L. Fernández Alemán, Ambrosio Toval, Sofia Ouhbi, and Ali Idri. 2016. Co-Located and Distributed Natural-Language Requirements Specification: Traditional versus Reuse-Based Techniques. *Journal of Software: Evolution and Process* 28, 3 (2016), 205–227. https://doi.org/10.1002/smr.1772

[47] CC2020 Task Force. 2020. *Computing Curricula 2020: Paradigms for Global Computing Education*. ACM, New York, NY, USA. https://doi.org/10.1145/3467967

[48] Jinjushang Chen, Lara Perez-Felkner, Chantra Nhien, Shouping Hu, Kristen Erichsen, and Yang Li. 2023. Gender Differences in Motivational and Curricular Pathways Towards Postsecondary Computing Majors. *Research in Higher Education* (hybrid) (Aug. 2023), 24. https://doi.org/10.1007/s11162-023-09751-w

[49] Joseph Chipps, Brittany Terese Fasy, Stacey A. Hancock, and Bradley McCoy. 2023. Ant and Bear Dance for Dokweebah: Using a Skokomish Story to Engage Middle School Students in Event-Driven Programming. In *Proceedings of the ACM Conference on Global Computing Education Vol 1 (CompEd '23)*. ACM, Hyderabad, India, 43–49. https://doi.org/10.1145/3576882.3617920

[50] *Aaron Ciaghi, Pietro Molini, Adolfo Villafiorita, and Komminist S. Weldemariam. 2013. Maputo Living Lab Summer School of ICTs: An Experience Report. In *Proceedings of the 2013 IST-Africa Conference & Exhibition* (Nairobi, Kenya) *(IST-ACE '13)*. IEEE, New York, NY, USA, 1–8.

[51] Victoria Clarke and Virginia Braun. 2019. Guidelines for Reviewers and Editors Evaluating Thematic Analysis Manuscripts.

[52] Tony Clear. 2024. THINKING ISSUES: Large Language Models, the 'Doctrine of Discovery' and 'Terra Nullius' Declared Again? *ACM Inroads* 15, 2 (June 2024), 6–9. https://doi.org/10.1145/3638564

[53] *Ricardo Colomo-Palacios, Nesrine Ben Yahia, and Xabier Larrucea. 2019. Gender Diversity among Computing Students: Reflections from Norway, Spain and Tunisia. In *Proceedings of the 7th International Conference on Technological Ecosystems for Enhancing Multiculturality* (León, Spain) *(TEEM '19)*. ACM, New York, NY, USA, 196–200. https://doi.org/10.1145/3362789.3362796

[54] *Ricardo Colomo-Palacios, Nesrine Ben Yahia, Xabier Larrucea, and Cristina Casado-Lumbreras. 2020. Is the Gender Gap Narrowing in Higher Education Computing Studies? The Case of Norway, Spain, and Tunisia. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje* 15, 4 (Nov. 2020), 336–343. https://doi.org/10.1109/rita.2020.3033211

[55] *Johannes C. Cronjé. 2006. Pretoria to Khartoum–How We Taught an Internet-Supported Masters' Programme across National, Religious, Cultural and Linguistic Barriers. *Educational Technology & Society* 9, 1 (2006), 276–288.

[56] *Olawande Daramola. 2021. Lessons from Postgraduate Supervision in Two African Universities: An Autoethnographic Account. *Education Sciences* 11 (2021), 21.

[57] *Salihu Ibrahim Dasuki, Peter Ogedebe, Rislana A. Kanya, Hauwa Ndume, and Julius Makinde. 2015. Evaluating the Implementation of International Computing Curricular in African Universities: A Design-Reality Gap Approach. *International Journal of Education and Development using Information and Communication Technology* 11, 1 (2015), 17–35.

[58] *Shaniel Davrajh and Riaan Stopforth. 2016. Integrating Lego Mindstorms and Developmental Outcomes to Address Engineering Education in Previously Disadvantaged Secondary Schools. In *Proceedings of the 2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference* (Stellenbosch, South Africa) *(PRASA-RobMech '16)*. IEEE, New York, NY, USA, 1–8. https://doi.org/10.1109/RoboMech.2016.7813182

[59] James R. Denbow and Phenyo C. Thebe. 2006. *Culture and Customs of Botswana*. Greenwood Press, Westport, Conn.

[60] *M. Bernardine Dias, Brett Browning, G. Ayorkor Mills-Tettey, and Nathan Amanquah. 2007. Robotics Education in Emerging Technology Regions. In *Proceedings of the 22nd AAAI Spring Symposium: Semantic Scientific Knowledge Integration* (Washington, DC, USA). AAAI, Palo Alto, CA, USA, 6.

[61] *M. Bernardine Dias, Brett Browning, G. Ayorkor Mills-Tettey, Nathan Amanquah, and Noura El-Moughny. 2007. Undergraduate Robotics Education in Technologically Underserved Communities. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation* (Rome, Italy) *(ICRA '07)*. IEEE, New York, NY, USA, 1387–1392. https://doi.org/10.1109/robot.2007.363178

[62] *M. Bernardine Dias, G. Ayorkor Mills-Tettey, and T. Nanayakkara. 2005. Robotics, Education, and Sustainable Development. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (Barcelona, Spain) *(ICRA '05)*. IEEE, New York, NY, USA, 4248–4253. https://doi.org/10.1109/robot.2005.1570773

[63] *M. Bernardine Dias, G. Ayorkor Tettey-Mills, and Joseph Mertz. 2005. The TechBridgeWorld Initiative: Broadening Perspectives in Computing Technology Education and Research. In *Proceedings of the 2005 International Symposium on Women and ICT (CWIT '05)*. ACM, New York, NY, USA, 9. https://doi.org/10.1145/1117417.1117434

[64] *Teresa Dirsuweit and Patricia Gouws. 2023. Barriers to the Full Participation of Girls in Robotics: A Case Study of a South African Community of Practice. *Interchange: A Quarterly Review of Education* 54, 4 (2023), 439–457. https://doi.org/10.1007/s10780-023-09504-9

[65] Maureen Doyle, Kevin G Kirby, and Gary Newell. 2008. Engaging Constructions: Family-Based Computing Experiences for Immigrant Middle School Students. *ACM SIGCSE Bulletin* 40, 1 (2008), 58–62.

[66] *Marcus Duveskog, Erkki Sutinen, Matti Tedre, and Mikko Vesisenaho. 2003. In Search of Contextual Teaching of Programming in a Tanzanian Secondary School. In *Proceedings of the 33rd Frontiers in Education Conference* (Westminster, CO, USA) *(FIE '03)*. IEEE, New York, NY, USA, 6. https://doi.org/10.1109/FIE.2003.1264728

[67] *Marcus Duveskog, Erkki Sutinen, Mieo Vesisenaho, and Chaltu Gasso. 2003. HIV/AIDS Education in Tanzania Blended with a Programming Course. In *Proceedings of the International Conference on Information Technology: Research and Education* (Newark, NJ, USA) *(ITRE '03)*. IEEE, New York, NY, USA, 179–183. https://doi.org/10.1109/itre.2003.1270598

[68] *Marcus Duveskog, Erkki Sutinen, Mikko Vesisenaho, and Evgeny Yusha. 2004. Simputer as a Platform for ICT Education in Tanzania. In *Proceedings of the 2004 IEEE International Conference on Advanced Learning Technologies* (Joensuu, Finland) *(ICALT '04)*. IEEE, New York, NY, USA, 1018–1023. https://doi.org/10.1109/icalt.2004.1357741

[69] *Barry Dwolatzky. 2004. Software Engineering Education in South Africa. *Transactions of the South African Institute of Electrical Engineers* 95, 4 (Dec. 2004), 226–235.

[70] Ron Eglash, Mukkai Krishnamoorthy, Jason Sanchez, and Andrew Woodbridge. 2011. Fractal Simulations of African Design in Pre-College Computing Education. *ACM Transactions on Computing Education (TOCE)* 11, 3 (2011), 1–14.

[71] *Passent Elkafrawy, Ahmed Kamal, Walaa Medhat, Mohamed El-Hadidi, and Ahmed Hassan Yousef. 2024. Statistical Analysis for Evaluation and Improvement of Computer Science Education. In *Proceedings of the 21st Learning and Technology Conference* (Jeddah, Saudi Arabia) *(L&T '24)*. IEEE, New York, NY, USA, 79–85. https://doi.org/10.1109/lt60077.2024.10468818

[72] *Ayman Elsayed. 2021. Science Centers as an Essential Tool for AI Pre-College Education in Developing Countries. In *Proceedings of the 2021 IEEE Global Conference on Artificial Intelligence and Internet of Things* (Dubai, United Arab Emirates) *(GCAIoT '21)*. IEEE, New York, NY, USA, 94–99. https://doi.org/10.1109/GCAIoT53516.2021.9692949

[73] Absalom El-Shamir Ezugwu, Olaide N. Oyelade, Abiodun M. Ikotun, Jeffery O. Agushaka, and Y. S. Ho. 2023. Machine Learning Research Trends in Africa: A 30 Years Overview with Bibliometric Analysis Review. *Archives of Computational Methods in Engineering* 30 (2023), 4177–4207. https://doi.org/10.1007/s11831-023-09930-z

[74] Toyin Falola. 2001. *Culture and Customs of Nigeria.* Greenwood Press, Westport, Conn.

[75] *Yousef Farhaoui. 2017. Teaching Computer Sciences in Morocco: An Overview. *IT Professional* 19, 4 (2017), 12–15. https://doi.org/10.1109/mitp.2017.3051325

[76] Jens Fendler and Heike Winschiers-Theophilus. 2010. Towards Contextualised Software Engineering Education: An African Perspective. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*. ACM, Cape Town South Africa, 599–607. https://doi.org/10.1145/1806799.1806888

[77] *Jens Fendler and Heike Winschiers-Theophilus. 2010. Towards Contextualised Software Engineering Education: An African Perspective. In *Proceedings of the 32nd International Conference on Software Engineering* (Cape Town, South Africa) *(ICSE '10, Vol. 1)*. ACM, New York, NY, USA, 599–607. https://doi.org/10.1145/1806799.1806888

[78] *José L. Fernández-Alemán, Juan M. Carrillo-de-Gea, Joaquín V. Meca, Joaquín N. Ros, Ambrosio Toval, and Ali Idri. 2016. Effects of Using Requirements Catalogs on Effectiveness and Productivity of Requirements Specification in a Software Project Management Course. *IEEE Transactions on Education* 59, 2 (2016), 105–118. https://doi.org/10.1109/TE.2015.2454472

[79] Diana Franklin, Phillip Conrad, Gerardo Aldana, and Sarah Hough. 2011. Animal Tlatoque: Attracting Middle School Students to Computing through Culturally-Relevant Themes. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*. ACM, Dallas TX USA, 453–458. https://doi.org/10.1145/1953163.1953295

[80] *Maxwell Fundi, Ismaila T. Sanusi, Solomon S. Oyelere, and Mildred Ayere. 2024. Advancing AI Education: Assessing Kenyan in-Service Teachers' Preparedness for Integrating Artificial Intelligence in Competence-Based Curriculum. *Computers in Human Behavior Reports* 14 (2024), 1–10. https://doi.org/10.1016/j.chbr.2024.100412

[81] *Vashti Galpin, Ian Sanders, Heather Turner, and Bernadine Venter. 2003. Computer Self-Efficacy, Gender, and Educational Background in South Africa. *IEEE Technology and Society Magazine* 22, 3 (2003), 43–48. https://doi.org/10.1109/mtas.2003.1237471

[82] Geneva Gay. 2002. Preparing for Culturally Responsive Teaching. *Journal of Teacher Education* 53, 2 (March 2002), 106–116. https://doi.org/10.1177/0022487102053002003

[83] Hanli Geyser. 2024. Decoloniality, Digital-coloniality and Computer Programming Education. *ACM Transactions on Computing Education* [Just Accepted] (Nov. 2024), 36 pages. https://doi.org/10.1145/3702332

[84] *Olly Gotel, Christelle Scharff, and Vidya Kulkarni. 2012. Mixing Continents, Competences and Roles: Five Years of Lessons for Software Engineering Education. *IET Software* 6, 3 (2012), 199–213. https://doi.org/10.1049/iet-sen.2011.0078

[85] *Olly Gotel, Christelle Scharff, Andrew Wildenberg, Mamadou Bousso, Chim Bunthoeurn, Phal Des, Vidya Kulkarni, Srisupa P. N. Ayudhya, Cheikh Sarr, and Thanwadee Sunetnanta. 2008. Global Perceptions on the Use of WeBWorK as an Online Tutor for Computer Science. In *Proceedings of the 38th Frontiers in Education Conference* (Saratoga Springs, NY, USA). IEEE, New York, NY, USA, 5–10. https://doi.org/10.1109/fie.2008.4720331

[86] *Irene Govender and Desmond W. Govender. 2012. A Constructivist Approach to a Programming Course: Students' Responses to the Use of a Learning Management System. *African Journal of Research in Mathematics, Science and Technology Education* 16, 2 (2012), 238–252. https://doi.org/10.1080/10288457.2012.10740742

[87] *Reginald G. Govender and Desmond W. Govender. 2021. A Physical Computing Approach to the Introduction of Computer Programming among a Group of Pre-Service Teachers. *African Journal of Research in Mathematics, Science and Technology Education* 25, 1 (2021), 91–102. https://doi.org/10.1080/18117295.2021.1924440

[88] *F.B. Greene, J.P. Jawitz, and N.J. Marais. 1996. Establishing Basic Computing Skills amongst First-Year Engineering Students from Diverse Educational Backgrounds. In *Proceedings of the 1996 AFRICON Conference* (Stellenbosch, South Africa) *(AFRICON '96, Vol. vol. 2)*. IEEE, New York, NY, USA, 923–927. https://doi.org/10.1109/afrcon.1996.563018

[89] *Jan H. Hahn, Elsa Mentz, and Lukas Meyer. 2009. Assessment Strategies for Pair Programming. *Journal of Information Technology Education* 8 (2009), 273–284.

[90] *Ben Hall, Sway Grantham, and Robert Whyte. 2024. Embedding Culturally Relevant Pedagogy in Practice: Considerations for Training and Resource Development. In *Proceedings of the 8th Conference on Computing Education Practice* (Durham, UK) *(CEP '24)*. ACM, New York, NY, USA, 29–32. https://doi.org/10.1145/3633053.3633058

[91] *Matthew Harsh, Ravtosh Bal, Jameson Wetmore, G. Pascal Zachary, and Kerry Holden. 2018. The Rise of Computing Research in East Africa: The Relationship between Funding, Capacity and Research Community in a Nascent Field. *Minerva: A Review of Science, Learning and Policy* 56, 1 (2018), 35–58. https://doi.org/10.1007/s11024-017-9341-1

[92] *Marietjie Havenga. 2018. Problem-Based Projects in Computer Programming: Students' Cooperation, Responsibilities and Dependencies. *African Journal of Research in Mathematics, Science and Technology Education* 22, 2 (2018), 254–264. https://doi.org/10.1080/18117295.2018.1483596

[93] *V. Horner and P. Gouws. 2016. E-Tutoring Support for Undergraduate Students Learning Computer Programming at the University of South Africa. In *Proceedings of the 2016 Computer Science Education Research Conference* (Pretoria, South Africa) *(CSERC '16)*. ACM, New York, NY, USA, 29–36. https://doi.org/10.1145/2998551.2998557

[94] *Willy A. Innocent and Visent T. Kipene. 2022. Assessment of Female Students' Perception and Integration of ICT Courses in Tanzania's Higher Education Institutions. *International Journal of Education and Development using Information and Communication Technology* 18, 2 (2022), 223–230.

[95] *Kimmo Järvinen, Tuukka Pienimäki, Tommi Teräsvirta, John J. Kyaruzi, and Erkki Sutinen. 1999. Between Tanzania and Finland: Learning Java over the Web. *Sigcse Bulletin : A Quarterly Publication of The Special Interest Group On Computer Science Education* 31, 1 (March 1999), 217–221. https://doi.org/10.1145/384266.299761

[96] *Cloneria N. Jatileni, Ismaila T. Sanusi, Sunday A. Olaleye, Musa A. Ayanwale, Friday J. Agbo, and Peter B. Oyelere. 2024. Artificial Intelligence in Compulsory Level of Education: Perspectives from Namibian in-Service Teachers. *Education and Information Technologies* 29 (2024), 12569–12596.

[97] *Philip O. Jegede, Emmanuel A. Olajubu, Adekunle O. Ejidokun, and Isaac O. Elesemoyo. 2019. Concept-Based Analysis of Java Programming Errors among Low, Average and High Achieving Novice Programmers. *Journal of Information Technology Education: Innovations in Practice* 18 (2019), 49–59. https://doi.org/10.28945/4322

[98] Paul Johannesson and Erik Perjons. 2014. *An Introduction to Design Science.* Vol. 10. Springer, New York City, NY, USA.

[99] *Evelyn K. Kahiigi, Mikko Vesisenaho, Henrik Hansson, Mats Danielson, and F. F. Tusubira. 2012. Modelling a Peer Assignment Review Process for Collaborative E-Learning. *Journal of Interactive Online Learning* 11, 2 (2012), 67–79.

[100] Mawera Karetai, Samuel Mann, Dhammika D. Guruge, Sherlock Licorish, and Alison Clear. 2023. Decolonising Computer Science Education - A Global Perspective. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto, ON, Canada) *(SIGCSE '23)*. ACM, New York, NY, USA, 1097–1102. https://doi.org/10.1145/3545945.3569870

[101] *Ermias A. Kassa and Enguday A. Mekonnen. 2022. Computational Thinking in the Ethiopian Secondary School ICT Curriculum. *Computer Science Education* 32, 4 (2022), 502–531. https://doi.org/10.1080/08993408.2022.2095594

[102] Paula Kotzé and Alta van der Merwe. 2009. The Research Foci of Computing Research in South Africa as Reflected by Publications in the South African Computer Journal. *South African Computer Journal* 44 (Dec. 2009), 67–84.

[103] Gloria Ladson-Billings. 1995. Toward a Theory of Culturally Relevant Pedagogy. *American Educational Research Journal* 32, 3 (Sept. 1995), 465–491. https://doi.org/10.3102/00028312032003465

[104] *Teemu H. Laine and Erkki Sutinen. 2011. Refreshing Contextualised IT Curriculum with a Pervasive Game Project in Tanzania. In *Proceedings of the 11th Koli Calling International Conference on Computing Education Research* (Koli, Finland) *(Koli Calling '11)*. ACM, New York, NY, USA, 66–75. https://doi.org/10.1145/2094131.2094145

[105] *Sam Lau and Philip Guo. 2023. From "Ban It Till We Understand It" to "Resistance Is Futile": How University Programming Instructors Plan to Adapt as More Students Use AI Code Generation and Explanation Tools Such as ChatGPT and GitHub Copilot. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 1* (Chicago, IL, USA) *(ICER '23)*. ACM, New York, NY, USA, 106–121. https://doi.org/10.1145/3568813.3600138

[106] *Shaimaa Lazem. 2016. A Case Study for Sensitising Egyptian Engineering Students to User-Experience in Technology Design. In *Proceedings of the 7th Annual Symposium on Computing for Development* (Nairobi, Kenya) *(ACM DEV '16)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/3001913.3001916

[107] *Shaimaa Lazem. 2019. Championing HCI Education to CS Undergraduates at a Grassroots Level: A Case Study in Egypt. *J. Usability Studies* 15, 1 (Nov. 2019), 8–22.

[108] *Janet Liebenberg, Magda Huisman, and Elsa Mentz. 2015. The Relevance of Software Development Education for Students. *IEEE Transactions on Education* 58, 4 (2015), 242–248. https://doi.org/10.1109/TE.2014.2381599

[109] *Janet Liebenberg, Magda Huisman, and Elsa Mentz. 2015. Software: University Courses versus Workplace Practice. *Industry and Higher Education* 29, 3 (2015), 221–235. https://doi.org/10.5367/ihe.2015.0254a

[110] *Janet Liebenberg, Elsa Mentz, and Betty Breed. 2012. Pair Programming and Secondary School Girls' Enjoyment of Programming and the Subject Information Technology (IT). *Computer Science Education* 22, 3 (2012), 219–236. https://doi.org/10.1080/08993408.2012.713180

[111] *Janet Liebenberg and Vreda Pieterse. 2018. Investigating the Feasibility of Automatic Assessment of Programming Tasks. *Journal of Information Technology Education: Innovations in Practice* 17 (2018), 201–223. https://doi.org/10.28945/4150

[112] *Anton Limbo, Gabriel T. Nhinda, and William Sverdlik. 2017. Rethinking Tertiary Computer Science Education: Let's Have Pi. In *Proceedings of the 2017 IST-Africa Week Conference* (Windhoek, Namibia) *(IST-Africa '17)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.23919/istafrica.2017.8102331

[113] S. Lopez-Pernas, M. Apiola, M. Saqr, A. Pears, and M. Tedre. 2023. A Scientometric Perspective on the Evolution of the SIGCSE Technical Symposium: 1970-2021. In *Past, Present, and Future of Computing Education Research: A Global Perspective*, M. Apiola, S. López-Pernas, and M. Saqr (Eds.). Springer, New York City, NY, USA, 213–243. https://doi.org/10.1007/978-3-031-25336-2_10

[114] *Hugo Lotriet, Machdel Matthee, and Patricia Alexander. 2011. Internet Access as a Structural Factor in Career Choice: A Comparison between Computing and Non-Computing Major Students. *African Journal of Research in Mathematics, Science and Technology Education* 15, 2 (2011), 138–153. https://doi.org/10.1080/10288457.2011.10740708

[115] Ellie Lovellette, Dennis J. Bouvier, and John Matta. 2024. Contextualization, Authenticity, and the Problem Description Effect. *ACM Transactions on Computing Education* 24, 2 (June 2024), 1–32. https://doi.org/10.1145/3643864

[116] *Jose Lukose and Kuttickattu J. Mammen. 2018. Enhancing Academic Achievement in an Introductory Computer Programming Course through the Implementation of Guided Inquiry-Based Learning and Teaching. *Asia-Pacific Forum on Science Learning and Teaching* 19, 2 (2018), 36.

[117] Douglas Lusa Krug, Edtwuan Bowman, Taylor Barnett, Lori Pollock, and David Shepherd. 2021. Code Beats: A Virtual Camp for Middle Schoolers Coding Hip Hop. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. ACM, Virtual Event USA, 397–403. https://doi.org/10.1145/3408877.3432424

[118] Andrew Luxton-Reilly and Andrew Petersen. 2017. The Compound Nature of Novice Programming Assessments. In *Proceedings of the 19th Australasian Computing Education Conference* (Geelong, Australia) *(ACE '17)*. ACM, New York, NY, USA, 26–35. https://doi.org/10.1145/3013499.3013500

[119] *N. Madhav and Meera K. Joseph. 2016. Cloud-Based Virtual Computing Labs for HEIs. In *Proceedings of the 2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies* (Balaclava, Mauritius) *(EmergiTech '17)*. IEEE, New York, NY, USA, 373–377. https://doi.org/10.1109/EmergiTech.2016.7737369

[120] *Craig Marais and Karen Bradshaw. 2016. Towards a Technical Skills Curriculum to Supplement Traditional Computer Science Teaching. In *Proceedings of the 2016 Innovation and Technology in Computer Science Education Conference (ITiCSE '16)*. ACM, New York, NY, USA, 338–343. https://doi.org/10.1145/2899415.2899446

[121] *Mudaray Marimuthu, Deepak Kumar, and Mishaan Chhagan. 2020. The Difference between Attitudinal Factors Influencing Success of Commerce and Computer Science Students at a South African University. *African Journal of Research in Mathematics, Science and Technology Education* 24, 3 (2020), 321–332. https://doi.org/10.1080/18117295.2020.1833544

[122] Linda Marshall, Vreda Pieterse, Lisa Thompson, and Dina M. Venter. 2016. Exploration of Participation in Student Software Engineering Teams. *ACM Transactions on Computing Education* 16, 2, Article 5 (Feb. 2016), 38 pages. https://doi.org/10.1145/2791396

[123] *Linda Marshall, Vreda Pieterse, Lisa Thompson, and Dina M. Venter. 2016. Exploration of Participation in Student Software Engineering Teams. *ACM Transactions on Computing Education* 16, 2 (2016), 1–38. https://doi.org/10.1145/2791396

[124] *Nicholas B. Mavengere and Mikko J. Ruohonen. 2011. Using Open Source Software for Improving Dialog in Computer Science Education - Case Mozambique University. *IFIP Advances in Information and Communication Technology* 348 Aict (2011), 52–61. https://doi.org/10.1007/978-3-642-19715-4_6

[125] *Chao Mbogo. 2019. A Structured Mentorship Model for Computer Science University Students in Kenya. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. ACM, New York, NY, USA, 1109–1115. https://doi.org/10.1145/3287324.3287447

[126] R. McCartney and Kate Sanders. 2023. ITiCSE Working Groups as an Engine for Community-Building. In *Past, Present and Future of Computing Education Research: A Global Perspective*, Mikko Apiola, Sonsoles López-Pernas, and Mohammed Saqr (Eds.). Springer, New York City, NY, USA, 213–243.

[127] *Loubna Mekouar. 2022. The Art of Teaching Programming Languages: Challenges and Accomplishments. In *Proceedings of the 2022 IEEE World Engineering Education Conference* (Santos, Brazil) *(EDUNINE '22)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/edunine53672.2022.9782372

[128] *Elsa Mentz, Roxanne Bailey, Marietjie Havenga, Betty Breed, Desmond Govender, Irene Govender, Frank Dignum, and Virginia Dignum. 2012. The Diverse Educational Needs and Challenges of Information Technology Teachers in Two Black Rural Schools. *Perspectives in Education* 30, 1 (2012), 70–78.

[129] *Elsa Mentz, J. L. van der Walt, and L. Goosen. 2008. The Effect of Incorporating Cooperative Learning Principles in Pair Programming for Student Teachers. *Computer Science Education* 18, 4 (2008), 247–260. https://doi.org/10.1080/08993400802461396

[130] Noemi Merayo and Alba Ayuso. 2023. Analysis of Barriers, Supports and Gender Gap in the Choice of STEM Studies in Secondary Education. *International Journal of Technology and Design Education* 33, 4 (2023), 1471–1498.

[131] *Salma A. O. Mohammed, Mariam E. Elhaddad, and Alfaroq O. M. Mohammed. 2017. Proposing a Solution for the Problem of Teaching Programming to Novice Students Using Soft Systems Methodology. In *Proceedings of the 2017 International Conference on Engineering & MIS* (Monastir, Tunisia) *(ICEMIS '17)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/icemis.2017.8273095

[132] *Tulimevava K. Mufeti. 2023. The Role of Reflective Practice during Emergency Online Teaching: Experiences from a Computer Science Course. *International Journal of Education and Development using Information and Communication Technology* 19, 2 (2023), 93–106.

[133] *Leah Mutanu and Philip Machoka. 2019. Enhancing Computer Students' Academic Performance through Predictive Modelling - A Proactive Approach. In *Proceedings of the 14th International Conference on Computer Science & Education* (Toronto, ON, Canada) *(ICCSE '19)*. IEEE, New York, NY, USA, 97–102. https://doi.org/10.1109/iccse.2019.8845452

[134] Nkundwe M. Mwasaga. 2022. *Design and Evaluation of Micro High-Performance Computing as an Educational Tool to Improve High-Performance Computing Usability*. Ph. D. Dissertation. Itä-Suomen yliopisto.

[135] *Nkundwe M. Mwasaga and Mike Joy. 2021. Contextualizing Micro High Performance Computing Artifacts in Higher Education. In *Proceedings of the 2021 IEEE Frontiers in Education Conference* (Lincoln, NE, USA) *(FIE '21)*. IEEE, New York, NY, USA, 1–9. https://doi.org/10.1109/fie49875.2021.9637400

[136] *Fadip A. Nannim, Nnenna E. Ibezim, Basil C. E. Oguguo, and Emmanuel C. Nwangwu. 2023. Effect of Project-Based Arduino Robot Application on Trainee Teachers Computational Thinking in Robotics Programming Course. *Education and Information Technologies* 29 (2023), 13155–13170. https://doi.org/10.1007/s10639-023-12380-6

[137] *Maria Ntinda, Mikko Apiola, and Erkki Sutinen. 2021. Mind the Gap: Aligning Software Engineering Education and Industry in Namibia. In *Proceedings of the*

2021 IST-Africa Virtual Conference (IST-Africa '21). IEEE, New York, NY, USA, 1–8.

[138] *Ngatchu D. Nyinkeu and Henry Ngatchu. 2017. Work and Play in Software Engineering Training: Experiences from the Silicon Mountain. In *Proceedings of the 30th IEEE Conference on Software Engineering Education and Training* (Savannah, GA, USA) *(CSEE&T '17)*. IEEE, New York, NY, USA, 112–116. https://doi.org/10.1109/cseet.2017.26

[139] *Anthony N. Nzeako. 1989. Teaching Computer Graphics in a Constrained Environment: The Top-down Approach. In *Proceedings 1989 Frontiers in Education Conference* (Binghamton, NY, USA) *(FIE '89)*. IEEE, New York, NY, USA, 46–54. https://doi.org/10.1109/fie.1989.69369

[140] *James R. Ochwa-Echel. 2005. *Gender Gap in Computer Science Education: Experiences of Women in Uganda*. Ph. D. Dissertation. Ohio University, USA.

[141] *Chijioke J. Olelewe, Chunli Dong, Mohammed Abdullahim, and Chinweike E. Nwangwu. 2023. Effects of Using a Video-Clip Instructional Strategy on Students' Performance in a Computer Networking Course. *Technology, Pedagogy and Education* 32, 3 (2023), 351–365. https://doi.org/10.1080/1475939x.2023.2201931

[142] *Ibrahim Ouahbi, Hassane Darhmaoui, Fatiha Kaddari, and Abdellah Bemmouna. 2019. Computer Science Program in Moroccan Secondary Schools: Curricula Analysis. *International Journal of Modern Education and Computer Science* 11, 3 (2019), 10–15. https://doi.org/10.5815/ijmecs.2019.03.02

[143] *Mayowa Oyedoyin, Ismaila T. Sanusi, and Musa A. Ayanwale. 2022. Young Children's Conceptions of Computing in an African Setting. *Computer Science Education* 34, 2 (2022), 1–36. https://doi.org/10.1080/08993408.2024.2314397

[144] Solomon S. Oyelere. 2018. *Design and Development of a Mobile Learning System for Computer Science Education in Nigerian Higher Education Context*. Ph. D. Dissertation. Itä-Suomen yliopisto.

[145] *Solomon S. Oyelere, Friday J. Agbo, Ismaila T. Sanusi, Abdullahi A. Yunusa, and Kissinger Sunday. 2019. Impact of Puzzle-based Learning Technique for Programming Education in Nigeria Context. In *Proceedings of the 19th IEEE International Conference on Advanced Learning Technologies* (Maceio, Brazil) *(ICALT '19)*. IEEE, New York, NY, USA, 239–241. https://doi.org/10.1109/ICALT.2019.00072

[146] *Solomon S. Oyelere, Jarkko Suhonen, Greg M. Wajiga, and Erkki Sutinen. 2018. Design, Development, and Evaluation of a Mobile Learning Application for Computing Education. *Education and Information Technologies* 23, 1 (2018), 467–495. https://doi.org/10.1007/s10639-017-9613-2

[147] Yogendra Pal and Sridhar Iyer. 2015. Effect of Medium of Instruction on Programming Ability Acquired through Screencast. In *2015 International Conference on Learning and Teaching in Computing and Engineering*. IEEE, Taipei, Taiwan, 17–21. https://doi.org/10.1109/LaTiCE.2015.38

[148] *Nelishia Pillay, B.T. Maharaj, and Gerdus van Eeden. 2018. AI in Engineering and Computer Science Education in Preparation for the 4th Industrial Revolution: A South African Perspective. In *Proceedings of the 2018 World Engineering Education Forum - Global Engineering Deans Council* (Albuquerque, NM, USA) *(WEEF-GEDC)*. IEEE, New York, NY, USA, 1–5. https://doi.org/10.1109/weef-gedc.2018.8629703

[149] *Sumarie Roodt and Yusuf Ryklief. 2019. Using Digital Game-Based Learning to Improve the Academic Efficiency of Vocational Education Students. *International Journal of Game-Based Learning* 9, 4 (2019), 45–69. https://doi.org/10.4018/IJGBL.2019100104

[150] *Diana Rwegasira. 2017. Agile Software Development Methods Practise in Computer Science Education: Adoption and Recommendations in Tanzania. In *Proceedings of the 2017 IST-Africa Week Conference* (Windhoek, Namibia) *(IST-Africa '17)*. IEEE, New York, NY, USA, 1–9. https://doi.org/10.23919/istafrica.2017.8102324

[151] Steven J. Salm and Toyin Falola. 2002. *Culture and Customs of Ghana*. Greenwood Press, Westport, Conn. London.

[152] Ismaila T. Sanusi. 2023. *Machine Learning Education in the K–12 Context*. Ph. D. Dissertation. University of Eastern Finland.

[153] *Ismaila T. Sanusi, Musa A. Ayanwale, and Thomas K.F. Chiu. 2024. Investigating the Moderating Effects of Social Good and Confidence on Teachers' Intention to Prepare School Students for Artificial Intelligence Education. *Education and Information Technologies* 29 (2024), 273–295. https://doi.org/10.1007/s10639-023-12250-1

[154] *Ismaila T. Sanusi, Musa A. Ayanwale, and Adebayo E. Tolorunleke. 2024. Investigating Pre-Service Teachers' Artificial Intelligence Perception from the Perspective of Planned Behavior Theory. *Computers and Education: Artificial Intelligence* 6 (2024), 1–15. https://doi.org/10.1016/j.caeai.2024.100202

[155] Ismaila Temitayo Sanusi and Fitsum Gizachew Deriba. 2024. What Do We Know about Computing Education in Africa? A Systematic Review of Computing Education Research Literature. *arXiv preprint arXiv:2406.11849* (preprint) (2024), 9. https://doi.org/10.48550/arXiv.2406.11849 arXiv:2406.11849

[156] *Ismaila T. Sanusi and Sunday A. Olaleye. 2022. An Insight into Cultural Competence and Ethics in K-12 Artificial Intelligence Education. In *Proceedings of the 2022 IEEE Global Engineering Education Conference* (Tunis, Tunisia) *(EDUCON '22)*. IEEE, New York, NY, USA, 790–794. https://doi.org/10.1109/EDUCON52537.2022.9766818

[157] *Ismaila T. Sanusi, Sunday A. Olaleye, Friday J. Agbo, and Thomas K.F. Chiu. 2022. The Role of Learners' Competencies in Artificial Intelligence Education. *Computers and Education: Artificial Intelligence* 3 (2022), 1–10. https://doi.org/10.1016/j.caeai.2022.100098

[158] *Ismaila T. Sanusi, Sunday A. Olaleye, Solomon S. Oyelere, and Raymond A. Dixon. 2022. Investigating Learners' Competencies for Artificial Intelligence Education in an African K-12 Setting. *Computers and Education Open* 3 (2022), 1–12. https://doi.org/10.1016/j.caeo.2022.100083

[159] *Ismaila T. Sanusi, Joseph O. Omidiora, Solomon S. Oyelere, Henriikka Vartiainen, Jarkko Suhonen, and Markku Tukiainen. 2023. Preparing Middle Schoolers for a Machine Learning-Enabled Future Through Design-Oriented Pedagogy. *IEEE access : practical innovations, open solutions* 11 (2023), 39776–39791. https://doi.org/10.1109/access.2023.3269025

[160] *Ismaila T. Sanusi, Solomon S. Oyelere, and Joseph O. Omidiora. 2022. Exploring Teachers' Preconceptions of Teaching Machine Learning in High School: A Preliminary Insight from Africa. *Computers and Education Open* 3 (2022), 1–10. https://doi.org/10.1016/j.caeo.2021.100072

[161] *Ismaila T. Sanusi, Solomon S. Oyelere, Henriikka Vartiainen, Jarkko Suhonen, and Markku Tukiainen. 2023. Developing Middle School Students' Understanding of Machine Learning in an African School. *Computers and Education: Artificial Intelligence* 5 (2023), 11. https://doi.org/10.1016/j.caeai.2023.100155

[162] *Ismaila T. Sanusi, Kissinger Sunday, Solomon S. Oyelere, Jarkko Suhonen, Henriikka Vartiainen, and Markku Tukiainen. 2024. Learning Machine Learning with Young Children: Exploring Informal Settings in an African Context. *Computer Science Education* 34, 2 (2024), 161–192. https://doi.org/10.1080/08993408.2023.2175559

[163] *Anastasia Shipepe, Lannie Uwu-Khaeb, Carmen de Villiers, Ilkka Jormanainen, and Erkki Sutinen. 2022. Co-Learning Computational and Design Thinking Using Educational Robotics: A Case of Primary School Learners in Namibia. *Sensors* 22 (2022), 1–20. https://doi.org/10.3390/s22218169

[164] *Annastasia Shipepe, Lannie Uwu-Khaeb, Emmanuel A. Kolog, Mikko Apiola, Kauna Mufeti, and Erkki Sutinen. 2021. Towards the Fourth Industrial Revolution in Namibia: An Undergraduate AI Course Africanized. In *Proceedings of the 2021 IEEE Frontiers in Education Conference* (Lincoln, NE, USA) *(FIE '21)*. IEEE, New York, NY, USA, 1–8. https://doi.org/10.1109/fie49875.2021.9637356

[165] *Anastasia Shipepe, Lannie Uwu-Khaeb, David V. Ruwodo, Ilkka Jormanainen, and Erkki Sutinen. 2023. Integrating Secondary School and Primary School Learners to Grasp Robotics in Namibia Through Collaborative Learning. *Robotics in Education, Lecture Notes in Networks and Systems* 747 (2023), 65–77. https://doi.org/10.1007/978-3-031-38454-7_7

[166] *Leslie Simulwi and Evaristo Musonda. 2020. The Impact of Compulsory Computer Studies on ICT Literacy at Junior Secondary Schools in Livingstone District. *International Journal of Information and Communication Technology Education* 16, 4 (2020), 20–34. https://doi.org/10.4018/IJICTE.2020100102

[167] Adalbert G. Soosai Raj, Eda Zhang, Saswati Mukherjee, Jim Williams, Richard Halverson, and Jignesh M. Patel. 2019. Effect of Native Language on Student Learning and Classroom Interaction in an Operating Systems Course. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 499–505. https://doi.org/10.1145/3304221.3319787

[168] *Marwa Soudi. 2014. Robotics Education in Africa: Africa Compete. In *Impacts of the Knowledge Society on Economic and Social Growth in Africa*, Lloyd G. Adu Amoah (Ed.). IGI Global, Hershey, Pennsylvania, USA, 215–230. https://doi.org/10.4018/978-1-4666-5844-8.ch012

[169] *Kissinger Sunday, Patrick Ocheja, Sadiq Hussain, Solomon S. Oyelere, Oluwafemi S. Balogun, and Friday J. Agbo. 2020. Analyzing Student Performance in Programming Education Using Classification Techniques. *International Journal of Emerging Technologies in Learning* 15, 2 (2020), 127–144. https://doi.org/10.3991/ijet.v15i02.11527

[170] *Kissinger Sunday, Seng Yue Wong, Balogun O. Samson, and Ismaila T. Sanusi. 2022. Investigating the Effect of Imikode Virtual Reality Game in Enhancing Object Oriented Programming Concepts among University Students in Nigeria. *Education and Information Technologies* 27, 5 (2022), 6819–6845. https://doi.org/10.1007/s10639-022-10886-z

[171] Florent Tasso and Monique O. Kouaro. 2018. Intégration de l'enseignement de l'informatique Dans Les Établissements d'enseignement Secondaire Du Bénin. In *Didapro 7 – DidaSTIC. De 0 à 1 Ou l'heure de l'informatique à l'école*. HAL, Lausanne, Switzerland, 14.

[172] Scott D. Taylor. 2006. *Culture and Customs of Zambia*. Greenwood Press, Westport, Conn.

[173] Josh Tenenberg and Robert McCartney. 2008. Grounding the Scholarship of Teaching and Learning in Practice. *Journal on Educational Resources in Computing* 8, 2, Article 4 (May 2008), 3 pages. https://doi.org/10.1145/1362787.1362788

[174] *Hannah Thinyane, Tulimevava K. Mufeti, Alfredo Terzoli, and Madeleine Wright. 2010. Google Docs and Skype for a Low Bandwidth Virtual Classroom for Developing Countries. In *Proceedings of the 2010 IST-Africa Conference* (Durban, South Africa) *(IST-Africa '10)*. IEEE, New York, NY, USA, 1–7.

[175] *Fatimah Tijani, Ronel Callaghan, and Rian de Villers. 2020. An Investigation into Pre-Service Teachers' Experiences While Transitioning from Scratch Programming to Procedural Programming. *African Journal of Research in Mathematics, Science and Technology Education* 24, 2 (2020), 266–278. https://doi.org/10.1080/18117295.2020.1820798

[176] Minh Tran, Heather Killen, Jen Palmer, David Weintrop, and Diana Franklin. 2024. Harmonizing Scratch Encore: Scaffolding K-8 Teachers in Customizing Culturally Responsive Computing Materials. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE '24)*. ACM, Portland OR USA, 1335–1341. https://doi.org/10.1145/3626252.3630756

[177] *Ethel Tshukudu, Maria Kallia, Katharine Childs, and William Darragh. 2023. Broadening Participation in Computing: Experiences of the Hour of Code in an African Country. In *Proceedings of the 18th Conference on Primary and Secondary Computing Education Research (WiPSCE '23)*. ACM, New York, NY, USA, 1–4. https://doi.org/10.1145/3605468.3605502

[178] *Ethel Tshukudu, Sofiat Olaosebikan, Kenechi Omeke, Alexandrina Pancheva, Stephen McQuistin, and Lydia J. Jilantikiri. 2022. Broadening Participation in Computing: Experiences of an Online Programming Workshop for African Students. In *Proceedings of the 2022 Conference on Innovation and Technology in Computer Science Education* (Dublin, Ireland) *(ITiCSE '22)*. ACM, New York, NY, USA, 393–399. https://doi.org/10.1145/3502718.3524773

[179] *Ethel Tshukudu, Sue Sentance, Oluwatoyin Adelakun-Adeyemo, Brenda Nyaringita, Keith Quille, and Ziling Zhong. 2023. Investigating K-12 Computing Education in Four African Countries (Botswana, Kenya, Nigeria, and Uganda). *ACM Transactions on Computing Education* 23, 1 (Jan. 2023), 9:1–9:29. https://doi.org/10.1145/3554924

[180] Ethel Tshukudu, Sue Sentance, Oluwatoyin Adelakun-Adeyemo, Brenda Nyaringita, Keith Quille, and Ziling Zhong. 2023. Investigating K-12 Computing Education in Four African Countries (Botswana, Kenya, Nigeria, and Uganda). *ACM TOCE* 23, 1, Article 9 (Jan. 2023), 29 pages. https://doi.org/10.1145/3554924

[181] *Peace B. Tumuheki, Jacques Zeelen, and George L. Openjuru. 2016. Motivations for Participation in Higher Education: Narratives of Non-Traditional Students at Makerere University in Uganda. *International Journal of Lifelong Education* 35, 1 (2016), 102–117. https://doi.org/10.1080/02601370.2016.1165745

[182] University of Eastern Finland. 2024. IMPDET Hubs in Tanzania and Uganda. https://sites.uef.fi/impdet/impdet-community/impdet-hubs/.

[183] *Arthur van der Poll, Izak van Zyl, and Jan H. Kroeze. 2020. Towards Decolonizing and Africanizing Computing Education in South Africa. *Communications of the Association for Information Systems* 47 (2020), 140–163. https://doi.org/10.17705/1cais.04707

[184] Roli Varma, John H. Falk, and Lynn D. Dierking. 2023. Challenges and Opportunities: Asian Women in Science, Technology, Engineering, and Mathematics. *American Behavioral Scientist* 67, 9 (Aug. 2023), 1063–1073. https://doi.org/10.1177/00027642221078509

[185] Mikko Vesisenaho. 2007. *Developing University-Level Introductory ICT Education in Tanzania:A Contextualized Approach*. Ph. D. Dissertation. Department of Computer Science and Statistics, University of Joensuu, Finland.

[186] *Mikko Vesisenaho, Erkki Sutinen, and H.H. Lund. 2006. Contextual Analysis of Students' Learning during an Introductory ICT Course in Tanzania. In *Proceedings of the 4th IEEE International Workshop on Technology for Education in Developing Countries* (Iringa, Tanzania) *(TEDC '06)*. IEEE, New York, NY, USA, 9–13. https://doi.org/10.1109/tedc.2006.6

[187] Ismael Villegas Molina, Adrian Salguero, Shera Zhong, and Adalbert G. Soosai Raj. 2023. The Effects of Spanish-English Bilingual Instruction in a CS0 Course for High School Students. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1*. ACM, Turku Finland, 75–81. https://doi.org/10.1145/3587102.3588845

[188] E. O. Wahab, S. O. Odunsi, and O. E. Ajiboye. 2012. Causes and Consequences of Rapid Erosion of Cultural Values in a Traditional African Society. *Journal of Anthropology* 2012, 1 (2012), 327061. https://doi.org/10.1155/2012/327061 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1155/2012/327061

[189] Jane Waite, Anjali Das, Alyson Hwang, and Sue Sentance. 2023. Culturally Relevant Areas of Opportunity for K-12 Computing Lessons. In *2023 IEEE Frontiers in Education Conference (FIE)*. IEEE, College Station, TX, USA, 1–5. https://doi.org/10.1109/FIE58773.2023.10343308

[190] *Ian Wayman and Michael Kyobe. 2012. Incorporating Knowledge of Legal and Ethical Aspects into Computing Curricula of South African Universities. *Journal of Information Technology Education: Innovations in Practice* 11 (2012), 139–157.

[191] *Horst Weinert and Dirk Pensky. 2011. Mobile Robotics in Education and Student Engineering Competitions. In *Proceedings of the 2011 AFRICON Conference* (Victoria Falls, Zambia) *(AFRICON '11)*. IEEE, New York, NY, USA, 1–5. https://doi.org/10.1109/afrcon.2011.6072186

[192] *Horst Weinert and Dirk Pensky. 2012. Mobile Robotics in Education – South African and International Competitions. In *Proceedings of the 5th Robotics and Mechatronics Conference of South Africa* (Gauteng, South Africa) *(RobMech '12)*. IEEE, New York, NY, USA, 1–6. https://doi.org/10.1109/robomech.2012.6558455

[193] *Ellen W. Zegura and Rebecca E. Grinter. 2013. Community Building for Capacity Building: Case Study of Liberia's iLab. In *Proceedings of the 6th International Conference on Information and Communications Technologies and Development* (Cape Town, South Africa) *(ICTD '13)*. ACM, New York, NY, USA, 171–174. https://doi.org/10.1145/2517899.2517936

[194] *Amir Zeid. 2007. Lessons Learned from Establishing a Software Engineering Academic Programme in Developing Countries. In *Proceedings of the 20th Conference on Software Engineering Education & Training* (Dublin, Ireland) *(CSEET '07)*. IEEE, New York, NY, USA, 11–18. https://doi.org/10.1109/cseet.2007.34

[195] *Amir Zeid and Moemen Elswidi. 2005. A Peer-Review Based Approach to Teaching Object-Oriented Framework Development. In *Proceedings of the 18th Conference on Software Engineering Education & Training* (Ottawa, ON, Canada) *(CSEET '05)*. IEEE, New York, NY, USA, 51–58. https://doi.org/10.1109/cseet.2005.3