Contents lists available at ScienceDirect

Scientific African

journal homepage: www.elsevier.com/locate/sciaf

A review on effective approach to teaching computer programming to undergraduates in developing countries[☆]

Idongesit Eteng[a,*], Sylvia Akpotuzor[b], Solomon O. Akinola[c], Iwinosa Agbonlahor[a]

[a] *Computer Science Department, University of Calabar, Cross River State, Nigeria*
[b] *Computer Science Department, Arthur Jarvis University, Cross River State, Nigeria*
[c] *Computer Science Department, University of Ibadan, Oyo State, Nigeria*

A R T I C L E I N F O

Article history:
Received 19 November 2021
Revised 30 March 2022
Accepted 3 June 2022

Editor DR B Gyampoh

Keywords:
Teaching Computer Programming
Programming Languages
Mobile compilers
Developing countries
National University Commission
Education

A B S T R A C T

Universities in developing countries and maybe a few in some developed countries are faced with the challenge of adopting an effective pedagogy for teaching Computer Programming Languages to students. This is due to lack of technological infrastructure and a number of peculiar limitations within these regions. The goal of this paper is two-fold, conducting a systematic review of related literature in Programming Languages and proposing a model for teaching Programming Languages effectively given sparse resources. On the systematic review, a search for literature from 4 databases using appropriate inclusion and exclusion techniques yielded 18 relevant research articles. Based on gaps found in literature, we propose an effective teaching approach that integrates teaching basic foundational aspects of programming, teaching students how to represent computing problems using these foundational concepts and using mobile compilers to compile their codes on the event that systems are not available. A few initial empirical results showing the effect of the adoption of a mobile tool for teaching Programming to undergraduate students are presented. This research recommends the adoption of an Online Console Compiler as the mobile Integration Development Environment (IDE). It also recommends some Programming Languages that should be adopted in teaching students from first year to fourth year using the National University Commission's (NUC) guidelines, course outlines from four Universities and data from Universities in developed countries. Care should be taken in the use of mobile phones as there are safety rules in some developed countries on the minimal set of acceptable systems for programmers. Recommendations for future research are discussed.

☆ Edited by: B Gyampoh
* Corresponding author.
E-mail address: ideteng@unical.edu.ng (I. Eteng).

https://doi.org/10.1016/j.sciaf.2022.e01240

Introduction

As technological advancement continues to rapidly change the way we live and do business, it is to be thought of that the foundation of any technological innovation is a computer programmer behind a system. It is no news that Information Technology has permeated through several disciplines today. Professionals within these disciplines are coming to terms with the need to possess substantive Information Communication Technology (ICT) knowledge. As stated at the world economic forum in Africa (2015) “It is hard to overstate the impact ICT could have on people’s lives in Africa” [1]. Computer programming skills therefore is at the heart of technological advancement across all disciplines and by extension a necessary skill for Economic Growth and National Development. The incorporation of ICT into the systems of developing countries will require the expertise of computer programmers. These computer programmers undergo several tutorials and intensive practical sessions, to be well equipped to create ICT solutions and manage the performance of ICT systems. The tutorial session is the womb that births this expertise. However, these tutorial sessions have not been efficient enough to inspire and educate apprentices. Reports have shown that this is caused by the inherent challenge with learning programming such that it requires a very different approach from what is obtainable in a traditional classroom learning session [2].

Computer programming as a course requires cognitive and meta-cognitive abilities. It requires the apprentice to understand the syntax and semantics of a selected Programming Language and apply their creativity, by using it to solve problems. It combines logical thinking with creativity. Generally, it is agreed that it takes roughly ten (10) years to turn a novice into an expert computer programmer; and such skills that require a lot of time to grab depends so much on internal motivation [3]. Challenges regarding the activity of programming were reported to be:- Learning a new language at the beginning, not knowing what and how to begin to code, debugging the silly mistakes, sitting for hours with no outputs, not understanding what the user wants from you, and many more [4]. However these challenges can be overcome or minimized with the right motivation and with practice over time. In addition to the inherent generic challenges in learning programming, developing nations are met with other peculiar challenges. In Tanzania (a developing country), there are peculiar challenges regarding learning programming, majorly socioeconomic and environmental challenges such as poor facilities, lack of up to date materials, inadequately trained personnel, inefficient pedagogical methods, poor educational background of the learners, limited and unstable electricity, cultural diversity, poor living conditions, low income, inability to acquire computers for home use and personal practices [3]. Other scholars have identified the generic challenges with education in developing countries as follows: Poor value for education, Opportunity cost on parents, Poor infrastructure, inadequate efficiency and quality, financial setbacks, equity and gender issues [5,6].

Much pedagogy have been suggested and experimented, in a bid to successfully aid scholars adopt programming skills. Most of which have recorded successes in spurring interest and improved learning as it is appraised by the average class grades in institutions where it has been implemented [7–10]. In all these advances and successes, it could not be sufficiently agreed that students in developing countries have had considerable progress in programming. In a study, It was reported that this is partly because there have not been much changes in the teaching methods employed. The study projected that this limitation is not exclusively a function of the instructor’s inability to adapt to new methods, but because there’s limited infrastructure at the disposal of the tertiary institutions in developing countries [3]. We conducted a systematic review of relevant literature to ascertain the state of arts of technology in developing countries. Considering the technology available at the disposal of institutions in developing communities, we propose an efficient approach to teaching programming to undergraduates in developing countries.

The main contributions of this work include the following:

- A systematic review of literature on teaching programming Languages in developing countries.
- An efficient approach for teaching programming to undergraduates in developing countries. The paper proposes a blended approach that involves the use of special Integrated Development Environments – IDE, theoretical soundness and the use of mobile phones. The approach seeks to achieve cost-effectiveness and applicability in limited state-of-the-art facilities.
- The paper suggests an array of Programming Languages to be taught to undergraduates – these combinations are methodologically chosen using the national guidelines, course descriptions from 4 universities and lessons learned from developed countries.
- Finally, we show initial results to validate our proposed approach.

The present study

The goal of this paper is two-fold: (1) to compile and review illuminating literature in the subject of teaching Programming Languages in developing countries and (2) to propose a two-fold solution that makes suggestions on Programming Languages to be taught and a blended approach for teaching these languages in the known reality of sparse state-of-the-arts facilities and financial resources.

In an attempt to proffer a solution on the best approach to teaching programming in developing countries, this article answers the following questions:

Research question 1: What approaches/Models have been identified in previous research for teaching Programming Languages in developing countries showing limitations and successes?

Research question 2: What were the problems/challenges identified in previous research with current approaches to teaching programming Languages and suggested solutions?

Research question 3: What model/approach can be proposed that is less costly and can be adopted by any educational institution in developing countries especially after the COVID 19 pandemic?

Research question 4: What improvement can be identified from our proposed approach in particular for:

Improving the performance of students' in learning Programming Languages?

The use of cost-effective technology in teaching Programming Languages to students?

Challenges as regards teaching Programming Languages led to the development of various models or approaches to teaching programming to undergraduate University students. Major general challenges have been highlighted in the introductory part of this paper. In the subsequent sections, the methods adopted for both searching for literature and proposing our approach will be described.

Materials and methods

Search and selection criteria

The literature search was conducted in April 2019. The following scientific databases were searched: Springer Link, Association for Computing Machinery (ACM) digital library, SCOPUS and Institute of Electrical and Electronics Engineers (IEEE). In the search query, two (2) sets of search terms covering the main concepts of the study's focus (identifying effective approaches to Teaching Computer Programming to Undergraduates) were combined:

[Approaches to teaching Programming Languages] and [tertiary institutions].

[Approaches to teaching Programming Languages]

The set of search terms were broadly defined and included the existing approaches to teaching Programming Languages especially in developing countries. The search was limited to peer-reviewed articles written in English and published between the year 2011–2022. The literature search did not focus exclusively on empirical studies, but also included theoretical and philosophical publications that discuss the approaches needed for teaching Programming Languages especially in developing countries. In total, 55 articles were found which covered the two sets of search terms that met the aforementioned inclusion criteria as shown in Fig. 1.

The abstracts of the fifty five (55) articles were screened to assess their relevance. Articles that did not focus on approaches to teaching Programming Languages were excluded. The fifty five (55) articles were screened by the first author to identify papers for inclusion and exclusion. The second author screened all relevant papers that should be included while other authors evaluated the relevant papers.

Based on the screening of the abstracts, forty one (41) out of the fifty five (55) articles were excluded from further analysis. Most of the excluded articles were based on approaches adopted in developed countries, repeated papers across databases selected, blog information and approaches applied in Secondary and Primary Schools. After the screening of the abstracts, 14 articles turned out to be relevant for further analysis. Since the researchers were not satisfied with the number of relevant articles, through snowballing, (scanning references from the full-text articles and selecting relevant publications), twenty two (22) additional titles that combined all sets of search terms were included in the next stage of the review, regardless of the year of publication. Although 4 articles were found not relevant because they only described the forms of learning. Thus a total of eighteen (18) articles were selected.

Summarizing articles and analysis

A spreadsheet was used to summarize the articles. This summary included bibliographical and methodological characteristics of the studies, theoretical frameworks, locations and remarks on the papers.

The summaries were made by the first author of this paper. The second, third and fourth authors served as a review board that critically discussed the spreadsheet displaying the data that were extracted from the articles by the first author. As a result, ambiguities regarding the quality and methodology of the studies and the exact meaning of concepts and definitions used in the articles were clarified by the research team.

Existing approaches/models of teaching computer programming

Several factors/ challenges motivated researchers to come up with more effective approaches to teaching Programming Languages especially to undergraduate students. These include: increased rate of assessment failures in programming courses, difficulty in managing large classes, inactive participation of students in learning, poor foundation and background knowledge from students' previous education level especially in the attainment of mathematical and abstraction skills.

A study was conducted to find out why students failed computer programming courses using a University in Nigeria as a case study. In their findings they identified lack of intrinsic motivation, lack of future expectation, anxiety, peer influences, and poor lecturer skills and behavior, as the challenges resulting in a high failure rate in introductory programming courses [11]. In addition to these challenges, a study in Ghana also revealed that even though research has been conducted about failures in programming courses which focused on the inability or the weaknesses of the student to understand the concepts, they found out that teaching methods were also factors that contributed to high rate of failure of computer programming

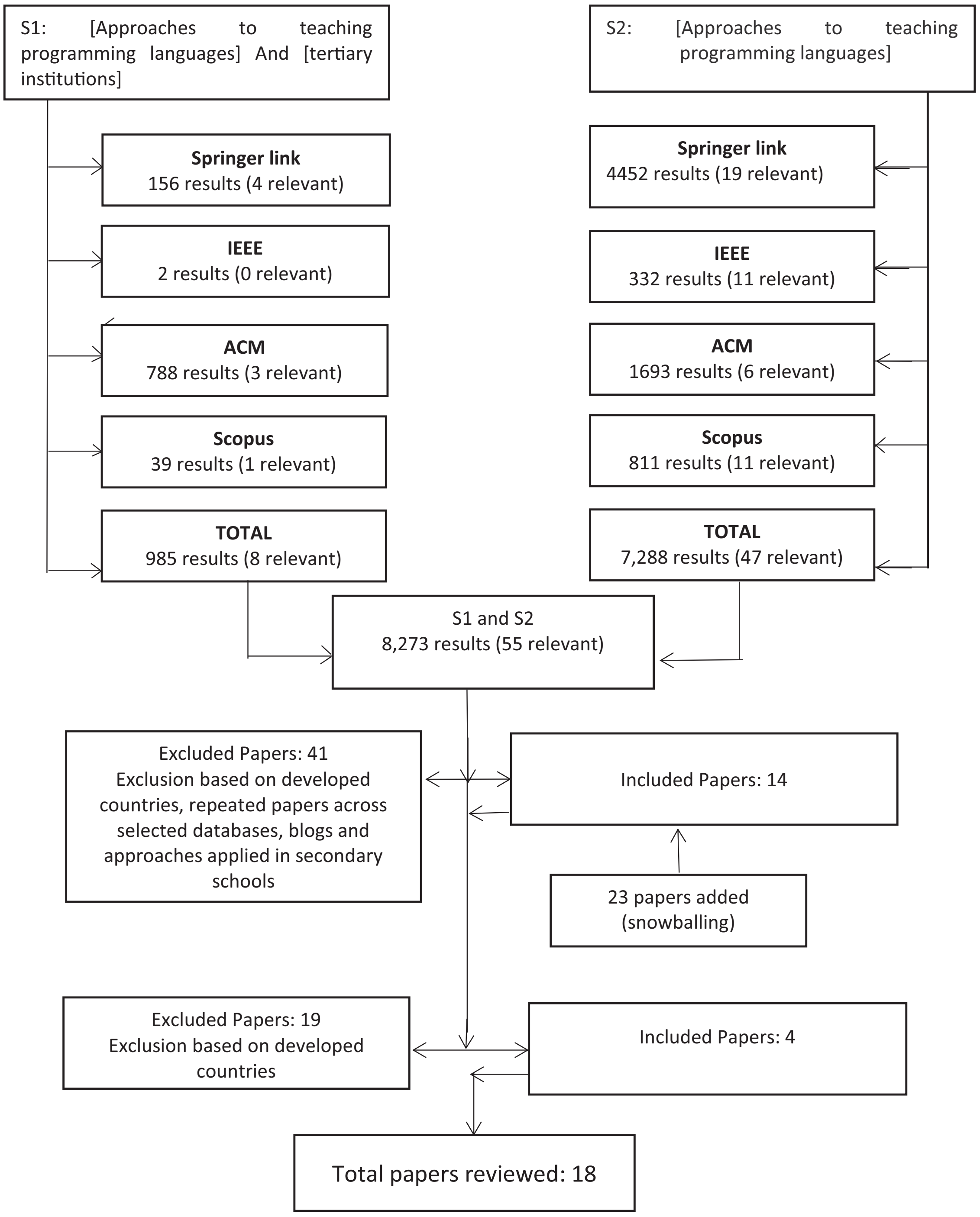
S1: [Approaches to teaching programming languages] And [tertiary institutions]
S2: [Approaches to teaching programming languages]
Springer link 156 results (4 relevant)
IEEE 2 results (0 relevant)
ACM 788 results (3 relevant)
Scopus 39 results (1 relevant)
TOTAL 985 results (8 relevant)
Springer link 4452 results (19 relevant)
IEEE 332 results (11 relevant)
ACM 1693 results (6 relevant)
Scopus 811 results (11 relevant)
TOTAL 7,288 results (47 relevant)
S1 and S2 8,273 results (55 relevant)
Excluded Papers: 41 Exclusion based on developed countries, repeated papers across selected databases, blogs and approaches applied in secondary schools
Included Papers: 14
23 papers added (snowballing)
Excluded Papers: 19 Exclusion based on developed countries
Included Papers: 4
Total papers reviewed: 18

Fig. 1. Article search and selection process.

courses [7]. Based on these challenges, a research was conducted to identify different teaching methods used by teachers to teach computer Programming Languages. These methods were rated by students who took programming courses, this is shown in Table 1. From their findings, out of the eight strategies mentioned, they recommended that teachers should adapt peer tutoring, pair/group programming and problem solving teaching strategies to give learners better opportunities to interact with their peers and teachers, hoping that this would reduce the failure rate of students in programming courses [7]. To address the issue of lack of intrinsic motivation, a review showed that the adoption of educational games could be used to teach computer programming as this method would reinforce students' intrinsic motivation through the sense

Table 1
Teaching methods and strategies [7].

Methods	Strategies
Lectures	Explicit teaching
Laboratory practice	Command style teaching
Projects	Teaching by task
e-learning	Problem solving teaching
Seminars and tutorials	Pre-recorded lectures
Field trips	Puzzled-based learning
Continuous assessment and examinations	Pair/group programming
Problem based teaching	Peer tutoring

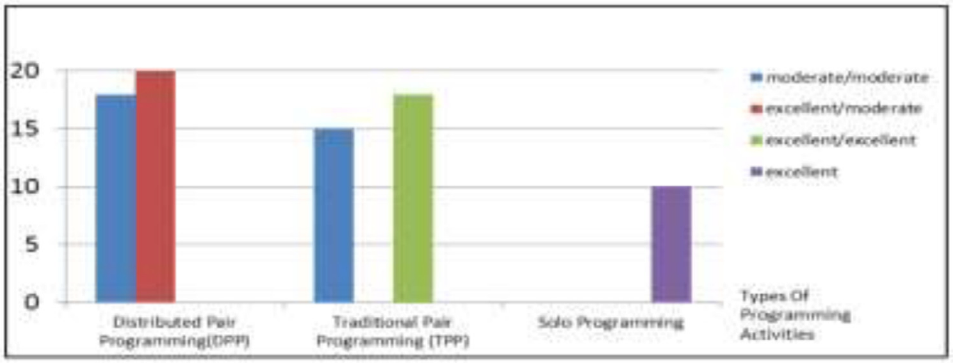

Fig. 2. The result of adopting pair programming [16].

of challenges [8]. In the United States, a study was conducted to implement a combined teaching method to assess the motivation of students towards learning Programming Languages. The researcher used active learning strategies, combined with their ability to think logically and service-learning, to help students increase their active participation and broaden their educational experience, respectively. In all, the activities promoted outside-classroom projects which helped to boost their self-confidence [10].

The management of large classes when it comes to teaching computer programming has posed several problems. This was confirmed by Kerr [12], where he stated that in such an environment, students may feel isolated and anonymous, leading them to disengage and dissociate from attendance. To tackle this issue, [9] presented a design of a blended learning solution for large class teaching of programming grounded in constructivist learning theory and use of free and open source technologies, the implementation of this approach improved the performance of more than 200 students in a semester. Contrary to this, [13] proposed a different approach to solving the challenge of managing large classes. The researcher proposed the adoption of agile principles and practices of pair programming in teaching computer programming to a large class. The study highlighted that since Agile software development has gained widespread popularity and acceptance in the software industry, the same approach could be integrated into teaching computer programming to undergraduates. Furthermore, the researcher highlighted limitations of adopting this approach. He stated that there would be need for the presence of teachers next to the students throughout the classes.

Previous studies have proven that when students work together in pairs as regards programming, the students feel more engaged and they achieve a good learning outcome. A study reported benefits of practicing pair programming in educational setting. Some of the reported benefits is the ability of students to be more confident in their work, Knowledge is consistently being shared between partners etc., [14,15] conducted a study that was implemented, monitored and regulated by instructors at a given time, their study proved that pair programming enhanced students' problem solving skills, improved quality of their works and increased teamwork. Furthermore, many studies have emphasized the need for collaborative tools for practicing pair programming among students. Fig. 2 displays the results obtained from a Malaysian University where an experiment was conducted to evaluate the effectiveness of the collaboration tool that was developed, to teach computer programming to first year students using the pair programming feature. From their findings, it was clear that students' performance improved in terms of collaborative learning, interactive participation among peers and improved communication skills. Furthermore, it was reported that the tool assisted students in practicing the techniques of pair programming with less supervision from their lecturers [16]. This was an improvement to the limitation stated by Isong [13] in adopting Agile practices of pair programming.

A few studies have been reported to show concern about the gender divide in programming courses. [17,18,32] showed that gender difference may not come into play at all when it comes to computer programming. Other factors like fear, lack of interest or attitude may however be responsible for the low representation of females in computing.

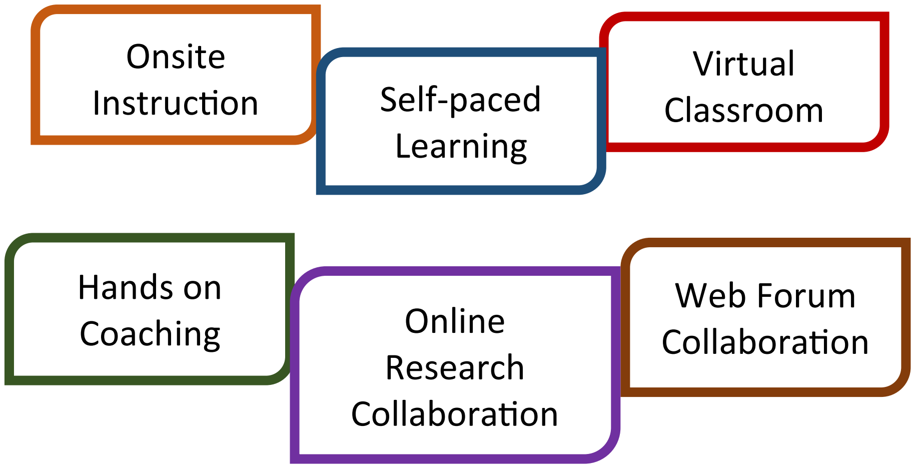

Fig. 3. An overview of Blended Learning Approach.

The need for a more effective approach in developing countries

Categorically, all models pointed in Section 3.2.1 can be categorized as either collaborative learning approaches or blended learning approaches. All models discussed in Section 3.2.1 come with their respective limitations. These limitations are highlighted and listed in Table 2 . Table 2 also contains additional approaches/ strategies to teaching Programming Languages. From Table 2, we can see that there is a need to consider factors identified in an educational system when adopting a blended approach or a collaborative approach, the selected approach must be in line with the institutions' goal or need. Looking at Universities in the developing countries, it is essential that one would have to come up with an effective approach of teaching Programming Languages that is less costly. This approach can be adopted in an institution with insufficient state-of-the-art facilities, and yet offer substantially good quality of learning.

From the literature gathered, we could identify five (5) approaches for teaching Programming Languages as presented in Table 3: These approaches includes: Collaborative learning, blended learning, experimental learning, integrated learning and constructivist approaches. So, inspite of the fact that there are several approaches and techniques, there is still a need for techniques that are cost-effective and can be applied where facilities are scarce.

Summary of literature

The proposed teaching approach

The concept of blended learning cannot be defined precisely, as different scholars put different content into the term, though all the researchers agree that the blended learning is an integrated learning experience that is controlled and guided by the instructor whether in the form of face-to-face communication or by virtual presence of the instructor [33–36] and many others investigated the effectiveness of using blended learning to tackle learning difficulties and to enhance students' motivation. It has been found that the blended learning is the most suitable model for learning programming because it facilitates and advances the learning process better than the traditional face-to-face learning approach. Additionally, students' academic performances and success rates were improved, and their satisfaction was increased [33,34,37]. Blended learning has several components, some of which are displayed in Fig. 3.

- On-site Instruction – This is when the tutor is available on site to give instructions to the learner on what to do and how to do it.
- Virtual classroom – In this situation, the tutor and the learner communicate through a digital device that is connected to a network.
- Self-paced Learning – This is when the learner is able to control the amount of material they consume as well as the duration of time they need to learn the new information.
- Hands-on Coaching – Hands-on coaching is a form of learning in which the learner learns by doing. Instead of simply listening to a lecture about a given subject, the student engages with the subject matter to solve a problem or create a software product.
- Web forum collaborative learning- This type of learning is a learning method based on the premise that knowledge can be created within a population where members actively interact by sharing experiences and taking on asymmetric roles through an internet of intranet [38].
- Online Research collaboration: This type of learning involves the collaboration of the academia to promote research among students with lectures serving as online supervisors to enable research activities to be shared among team members [39].

Table 2
Limitations of Identified Models/Approaches.

Category: Collaborative Learning Approach			
Author/ Title	**Approach/ Strategy**	**Successes**	**Limitation**
Owolabi, J., Adedayo, O.A., Amao-Kehinde, A.O, Olayanju, T.A Effects of Solo and Pair Programming Instructional Strategies on Students' Academic Achievement in Visual-Basic.Net Computer Programming Language [19]	Solo and Pair Programming Instructional Strategies. The sample comprised 68 subjects distributed over the three treatment groups from three hundred computer science students.	The study assesses the relative effectiveness of solo and pair programming instructional strategies on students' academic achievement in Visual-Basic.Net Computer Programming Language. Their findings shows that both solo and pair programming approaches are more effective approaches to learning computer programming when compared to the lecture method that has been in use for decades [15–17].	Solo and pair programming approaches will work better in tertiary institutions that would only provide sufficient computers in computer laboratories.
Manuel B. Garcia Cooperative learning in computer programming: A quasi-experimental evaluation of Jigsaw teaching strategy with novice programmers [20]	Cooperative learning in computer programming using Jigsaw teaching Strategy. A quasi-experimental research using a nonequivalent control group pretest-posttest design was adopted.	The study evaluated the impact of cooperative learning using jigsaw technique when teaching computer programming to novice programmers. The findings of the paper showed that cooperative learning using Jigsaw technique is a valid and effective teaching strategy when handling novice programmers in an introductory programming course	It was proven to only work for non-complex modules or concepts of programming courses.
Yao Lu, Xinjun MAO, Tao Wangi, Gang Yini, Zude Li [25] Improving students' programming quality with the continuous inspection process: a social coding perspective	Inspection process: a social coding perspective.	The paper is proven to improve students' programming quality using continuous inspection in a classroom setting. This approach also improves the collaboration of students' projects.	It is dependent on practicing programming in the laboratory.
Isong B A Methodology for Teaching Computer Programming: first year students' perspective. International Journal of Modern Education and Computer Science [13]	Agile practices of Peer tutoring/ pair programming/ Problem solving	Proven to improve communication among students and teachers. Students' communication among their peers is also improved. Proven to reduce the failure rate of students	There is need for the presence of teachers next to the students throughout the classes in order to get the class to meet up with expected course contents. It is dependent on fully equipped laboratories.
Sabarinath R & Quek C L G A case study investigating programming students' peer review of codes and their perceptions of the online learning environment. [26]	programming students' Peer review of codes and online learning.	The study contributes to the programming education in schools by investigating how students learn in an online programming environment while involved in peer review of codes. The results show that students were mostly able to partake meaningfully in their peer review activities and their ability to receive feedback was found to be one of the most dominating positive aspects besides improvement in students' programming skills [26].	Some students are intimidated because their codes would be exposed to their peers.
Porter L, Guzdial M, McDowell C & Simon B Success in introductory programming: What works? [28]	Pair programming, peer instruction, and media computation.	The research reports successes of adopting Pair programming, peer instruction, and media computation as approaches to teaching and improving programming courses.	More emphasis was made on the approaches to teaching programming courses while there was less focus in the available programming tools that could support learning.
Category: Blended Learning Approach			
Author/ Title	**Approach/ Strategy**	**Successes**	**Limitation**
Sohail Iqbal Malik & Roy Mathew & Rim Al-Nuaimi & Abir Al-Sideiri Jo Coldwell-Neilson Learning problem solving skills: Comparison of E-learning and M-learning in an introductory programming course [24]	E-learning and M-learning in an introductory programming course	The paper demonstrates the use of an application called PROBSOL, which is designed to focus on, and enhance, novice programmers' problem solving skills.	The proposed tool is only available for novices with laptops.

(*continued on next page*)

Table 2 (*continued*)

Wang Y, Li H, Feng Y, Jiang Y & Liu Y Assessment of programming language learning based on peer code review model: Implementation and experience report [30]	Peer code review model:	The paper is proven to allow students peer review programs written by other students, share ideas and make suggestions to achieve a collaborative and interactive learning.	Could be frustrating at times to work physically in pairs especially in large classes. This method is dependent on fully equipped laboratories.
Tiantian Wang*, Xiaohong Su, Peijun Ma, Yuying Wang, Kuanquan Wang Ability-training-oriented automated assessment in introductory programming [31].	AutoLEP, an automated learning and assessment system, was developed by us, to aid novice programmers to obtain programming skills.	The study developed an automated learning and assessment system called AutoLEP. AutoLEP is an ability-training-oriented system. It was developed to aid novice programmers in obtaining programming skills	This approach is only suitable for universities with equipped laboratories.
Malliarakis C, Satratzemi M, Xinogalos S Educational games for teaching computer Programming, Research on e-learning and ICT in Education [8].	Educational game method	Equips students with the skill of critical thinking. Students are able to work in groups and collaborate with one another. Students are motivated because of challenges presented to them in the game	The need for highly skilled teachers to design and develop games suitable for computer programming in line with the programming course outline.
Category: Experimental Learning Approach			
Author/ Title	**Approach/ Strategy**	**Successes**	**Limitation**
Chijioke Jonathan Olelewe, Emmanuel E. Agomuo, Peter Uzochukwu Obichukwu Effects of B-learning and F2F on college students' engagement and retention in QBASIC programming [18]	B-learning and F2F in QBASIC programming. This study focuses on effects of b-learning and face-to-face (F2F) on college students' engagement and retention in QBASIC programming.	The study focuses on effects of b-learning and face-to-face (F2F) on college students' engagement and retention in QBASIC programming. Findings showed that b-learning facilitated college students' engagement and improved knowledge retention in QBASIC programming compared to their counterparts in the F2F group.	The approach is limited to one language.
Batia B T, Gelderblomb H, Biljon J. A blended learning approach for teaching computer programming: Design for large classes in Sub-Saharan Africa [9].	Constructivist learning /open source technologies	Proven to improve problem solving and computer programming skill in students. Students were motivated to work in small groups, share ideas and learn from each other.	More time is spent during collaboration. This may not be suitable for large classes Course outlines may not be completely covered.
Drini M Using New Methodologies in Teaching Computer Programming [10]	Active learning and service learning	Proven to enable students use computer programming to solve real world problems since they carry out community projects. Students gain deeper understanding of the computer programming course.	May not be effective in a larger class Course outlines may not be completely covered because of time constraints.
Category: Integrated Learning Approach			
Author/ Title	**Approach/ Strategy**	**Successes**	**Limitation**
Stelios Xinogalos Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy [22]	Strategies, tools and pedagogy of designing and deploying programming courses	The study projects the usage of a pseudo-language for an introduction to programming to help improve programming courses that would be beneficial to non-CS majors.	More focus is pointed towards improving the learning pattern of students when studying introductory programming with less focus on the available tools needed for studying the designed programming courses.
Jun Tan, Xianping Guo, Weishi Zheng Ming zhong [29] Case-based teaching using the Laboratory Animal System for learning C/C++ programming.	This paper reports a study based on a case teaching model in C/C++ programming.	The study identifies case teaching model as an approach to making students active in programming courses, even as it relies on solid learning theory and pedagogical Strategies.	This approach is only suitable for universities with equipped laboratories.
Koorsse M, Cilliers C, & Calitz A [27] Programming assistance tools to support the learning of IT programming in South African secondary schools.	Programming assistance tools to support the learning of IT	The research determined if the use of a Programming Assistance tool (PAT) impacted IT scholars' understanding of programming concepts and their motivation towards programming in general .	The study focused on three (3) language dependent tools. One of which is available for both laptops and mobile devices while others are only available for laptops.

(*continued on next page*)

Table 2 (*continued*)

Category: Constructivist Learning Approach			
Author/ Title	**Approach/ Strategy**	**Successes**	**Limitation**
Malik S I & Coldwell-Neilson J. A model for teaching an introductory programming course using ADRI [21]	Teaching an introductory programming course using ADRI - Approach, Deployment, Result, Improvement.	The paper shows that ADRI model promotes more programming practices as compared to traditional teaching to address the learning difficulties that students encounter when studying introductory programming.	This approach is only suitable for universities with equipped laboratories.
IlKyu Yoon, JaMee Kim, WonGyu Lee [23] The analysis and application of an educational programming language (RUR-PLE) for a pre-introductory computer science course.	Teaching programming Language (RUR-PLE) for a pre-introductory computer science Course.	The study reports how a novice programmer could program quickly and easily by using a Python based programming learning environment called RUR-PLE to improve their knowledge of programming.	The proposed tool is only available for novices with laptops.

Table 3
Classification of Computer Programming Languages.

Category	**Languages in this category**	**About**	**Examples**
Low Level Language	Machine language [40]	Machine languages are numeric codes for the operations that a particular computer can execute directly. These codes are strings of 0 s and 1 s, or binary digits (“bits”), which are frequently converted both from and to hexadecimal (base 16) for human viewing and modification.	MIPS architecture
	Assembly language	Assembly languages are above machine languages. Short mnemonic codes are used to represent instructions which allow the programmer to introduce names for blocks of memory that hold data.	IBM PC DOS and Turbo Pascal.
High Level Language	Procedural languages [41]	Procedural languages are computer Programming Languages that specify a series of well-structured steps and procedures within its programming context to compose a program.	C/C++, Java, ColdFusion and PASCAL.
	Imperative Programming Languages. [42]	These are languages where functions are implicitly coded in every step required to solve a problem.	C, C++, C# and Java.
	Declarative Programming Languages [43]	These languages are used by the programmer to define the problem to be solved rather than how to solve the problem.	Structure Query Language (SQL), Hyper Text Markup Language (HTML), Extended Markup Language (XML) and Cascading Style Sheet (CSS).
	Functional / Logic Programming Languages [43]	Functional languages are constructed by applying and composing functions. Logic Programming Languages are largely based on formal logic.	Haskell, JavaScript, Python, LISP, ML Elm and Clojure. Prolog,Maude
	Object Oriented Programming Languages (OOP). [44]	OOP languages use objects and implement real world entities like inheritance, hiding, polymorphism etc., in programming.	C++, C#, Cobol, Fortran, Java, JavaScript, Objective C, Pascal, Perl, PHP, Python, Swift etc.

Considering how this approach has been beneficial in teaching, we employ this approach in solving the problem of teaching Computer programming to students across levels in tertiary institutions with the inclusion of adopting the use of mobile compilers to help students learn and practice the Programming Languages they learn.

Programming languages

In proposing a more effective approach using the blended learning model, this paper considers the selection of Programming Languages to be taught in educational institutions. This section provides details on the classes of Programming Languages and their examples.

Table 4
Course scheme and programming languages taught in universities.

S/N	Name of University	Year 1	Year 2	Year 3	Year 4
1	University of Calabar (Computer Science degree runs for 4 years)	**Course Scheme:** Introduction to computer programming **First Semester:** C++ **Second Semester:** C++	**Course Scheme:** Introduction to Object-oriented programming Language **First Semester:** Java **Second Semester:** Java	**Course Scheme:** Data structure and algorithms, organization of programming languages and introduction to compiler construction **First Semester:** Java **Second Semester:** Java	**Course Scheme:** Software design and development **First Semester:** HTML and CSC. **Second Semester:,** SQL and PHP
2	Arthur Jarvis University (Computer Science degree runs for 4 years)	**Course Scheme:** Introduction to computer science. **First Semester:** N/A **Second Semester:** N/A	**Course Scheme:** problem solving, computer programming, fundamentals of data structures operating systems and structured programming. **First Semester:** C++/Java **Second Semester:** BASIC /Java/C/C++	**Course Scheme:** Compiler construction, systems analysis and design, computer Architecture, operating system, algorithm and complexity analyaia and data management **First Semester:** Java, and C++ **Second Semester:** FORTRAN and COBOL, SQL	**Course Scheme:** organization of programing languages, net-centric computing, software engineering, database management, artificial intelligence and compiler construction **First Semester:** HTML, CSS, Javascript and PHP **Second Semester:** LISP, SQL,
3	University of Nigeria (NSUKA). (Computer Science degree runs for 4 years)	**Course Scheme:** Introduction to computer programming **First Semester:** Introduction to programming, Q BASIC **Second Semester:** Q BASIC	**Course Scheme:** Data structure and algorithms, organization of programming languages and introduction to compiler construction **First Semester:** Java **Second Semester:** C++	**Course Scheme:** Database Management System **First Semester:** Database Management System 1 **Second Semester:** Database Management System 2.	**First Semester:** Structured Programming **Second Semester:** Structured Programming
4	Federal University of Technology Owerri (FUTO). (Computer Science degree runs for 5 years)	**Course Scheme:** General courses taught. **First Semester:** No programming language taught. **Second Semester:** General courses taught. No programming language taught.	**Course Scheme:** Introduction to computer programming. **First Semester:** Computer and Application I. No programming language taught. **Second Semester:** Computer and Application II. QBASIC and FORTRAN	**Course Scheme:** Data structure and algorithms, organization of programming languages and introduction to compiler construction **First Semester:** C++ **Second Semester:** Java 1	**Course Scheme:** Database Management System **First Semester:** SQL **Second Semester:** Java 2

There are many different types of Programming Languages which exist for a variety of reasons. Some Programming Languages were developed to be particularly suitable to solve a specific type of problem, and others were developed with specific goals such as to aid beginner programmers. Programming Languages are majorly classified into low level and high level languages as shown in Table 4**.**

Effective methods that could easily be adaptable to institutions in developing countries considering the peculiarities are described further.

Proposed choice of programming languages

This research informs on the ideal Programming Languages to be taught at each level of study for easy comprehension and corresponding learning, taking into account these students' motivation, exposure and previous educational background.

Programming Languages are classified into seven programming paradigms. These Programming Languages have been thoroughly described in [Section 2.2.2**]** above. A sample of a recommended spread of Programming Languages that can be taught to undergraduate students in developing countries is shown in Table 4.

In justifying these recommended Computer Programming Languages, several approaches were adopted in this research which includes:

- A study of the University BMAS (Benchmark Minimum Academic Standards) from NUC (National Universities Commission) for Computer Science in Nigeria.
- A detailed analysis of the adopted programming courses for four Universities in Nigeria - (Federal University of Technology Owerri, University of Ibadan, Arthur Jarvis University, University of Calabar, and University of Nigeria).
- A review of a Study from RESEARCHGATE on what programming Languages to teach undergraduate students.
- A critical look at the explosion of Mobile Technology.

A study of the University BMAS from NUC for Computer Science.

A careful study of the BMAS shows that the Universities are given the right and privilege to choose individual Programming Languages to teach. However, the course content details concepts that the students are expected to learn at every level. This validates this research as it makes suggestions on Programming Languages to teach with supporting claims. Also, our suggested Languages and methods were chosen based on information from the BMAS. For example, from the excerpt of the BMAS, shown below, for teaching Problem Solving at the 100 level, which is the first course with some programming content, it is suggested that the implementation of algorithms in Programming Language be taught. We suggest that python and Java be taught at that level and we describe the reasons for our suggestions in subsequent sections. See excerpts from BMAS below:

CSC 101: Introduction to Computer Science (3 Units: LH 30,* PH*: 45) Survey of computers and information processing and their roles in society. *This course introduces a historical perspective of computing, hardware, software, information systems, and human resources and explores their integration and application in business and other segments of society. Students will be required to complete lab assignments using the PC's operating system, and several commonly used applications, such as word processors, spreadsheets, presentations, graphics and other applications. Internet and on-line resources, browsers and search engines.*

CSC 102: Introduction to Problem Solving (3 Units: LH 30,* PH *45)

Role of Algorithms in problem solving process, concepts and properties of Algorithms. Implementation strategies, Development of Flow Charts, Pseudo Codes. Program objects. Implementation of Algorithms in a programming Language - Visual BASIC/JAVA/C/C+ [45].

At the 200 level, the BMAS offers two programming courses to be taught as Computer Programming I and Computer Programming II. It is suggested that the Programming Language to be taught at this level should be: widely used and should be able to teach structured Programming Concepts. We suggest C++ and intermediate level Java. See an excerpt from the 2015 BMAS.

CSC 201: Computer Programming I (3 Units: LH 30,* PH *45).

Introduction to problem solving methods and algorithm development, designing, coding, debugging and documenting programmes using techniques of a good programming language style, programming language and programming algorithm development. A widely used programming language should be used in teaching the above.

CSC 202: Computer Programming II (3 Units: L30, P45).

Principles of good programming, structured programming concepts, Debugging and testing, string processing, internal searching and sorting, recursion. Use a programming language different from that in CSC 201 Eg., C Language [45].

At the 300 level, the courses that relate to programming from the BMAS(NUC, 2015) are Structured Programming, Object-Oriented Programming, Data Management, Compiler Construction and Survey of Programming Languages.

At this level, the student is expected to have had knowledge of most of the programming Languages, in surveying programming Languages, the student should be able to program in some classes of languages such as Logic and functional languages. We suggest Prolog as a logic language and Maude as a formal Language- our emphasis is on open source Languages. We also suggest front end scripting Languages to be taught at this level. Finally, we suggest database languages – Structured Query Language (SQL) to be taught to complement the theoretical Database course.

CSC 332: Survey of Programming Languages (4 Units: LH 45;* PH *45).

Overview of programming languages: History of programming languages, Brief survey of programming paradigms (Procedural languages, Object-oriented languages, Functional languages, Declarative – non-algorithmic languages, Scripting languages), the effects of scale on programming methodology; Language Description: Syntactic Structure (Expression notations, abstract Syntax Tree, Lexical Syntax, Grammars for Expressions, Variants of Grammars), Language Semantics (Informal semantics, Overview of formal semantics, Denotation semantics, Axiomatic semantics, Operational semantics); Declarations and types: The concept of types, Declaration models (binding, visibility, scope, and lifetime), Overview of type-checking, Garbage collection; Abstraction mechanisms: Procedures, function, and iterations as abstraction mechanisms, Parameterization mechanisms (reference vs. *value), Activation records and storage management, Type parameters and parameterized types, Modules in programming languages; Object oriented language paradigm; Functional and logic language paradigms* [45].

At the 400 level, we solidify on concepts that have been learnt already, so advanced java is suggested (any other Object Oriented Language can be taught here). Advanced SQL is also suggested (a NOSQL Language like Mungo DB can also be taught). Finally, Server side programming and relevant frameworks are suggested.

A detailed analysis of the adopted Programming Language courses for four Universities in Nigeria

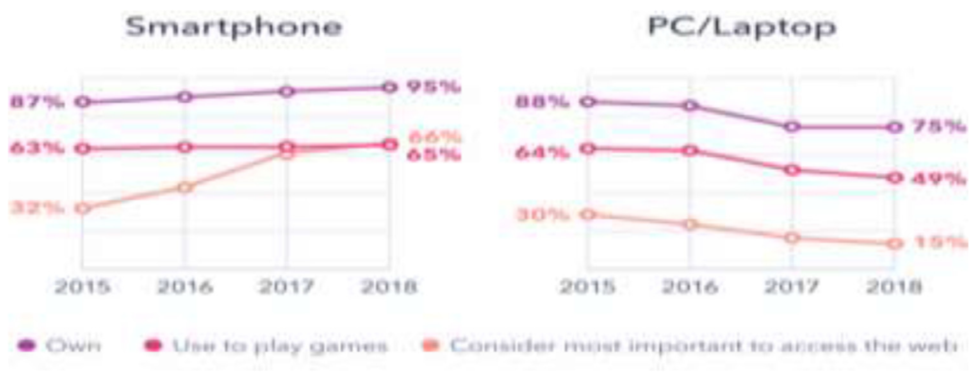

Fig. 4. Increase in the use of smartphones over PCs/Laptops [35].

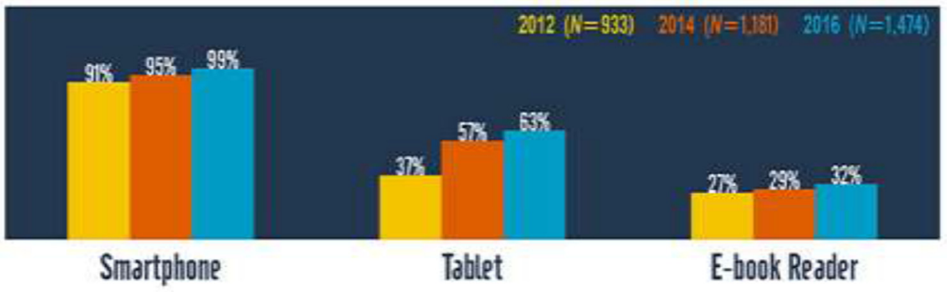

Fig. 5. Ownership of smart phones [36].

c) A review of a Study from RESEARCHGATE on what programming Languages to teach undergraduate students.

A study was conducted on one of the top European commercial social networking sites for scientists and researchers (ResearchGate) where papers are shared, questions are asked and answers are given to questions asked. A researcher from Arab American University asked 'Do you think it is a good idea to teach Python as the first programming course instead of C++?', [46]. Fourty (40) responses were given and most popular responses that had the highest recommendations emphasized that It is frustrating for fresh year students to learn things like pointers, references, headers, compilation, etc., without understanding in practice why they really need them. With Python, students understand programming concepts like data types, variables, control structures, loop structures functions, objects and classes more because python syntax is easy and it has a large standard library that enables students to do real applications, students will feel that they do something useful not just calculation and output of control structures (selection, repetition).

The second course recommended was C++. With C++ they would have deep understanding of object oriented programming concepts. They would also know the difference between a compiler, linker and loader.

Followed by a review done by Gupta [47], the third course this research recommends is Java because Java is better for teaching Algorithms and data structures. Java has a great collection of packages for data structures, such as List, Hashmap, dynamic arrays, etc., which are not available in C++.

The field of web development has become a hotspot in recent years. With websites gaining more and more traction, organizations have realized that to capture more leads they need to have a strong online presence. As a result, they seek out talented people who can make use of the latest technologies to design and develop the best looking and best-performing web applications for them [48]. This is why this research recommends the fourth and fifth courses to be HTML,CSS & JavaScript and SQL & PHP. These languages are suitable for students learning front end and back end web applications. These reasons also supported our suggestion of web programming Languages in the curriculum of Teaching Programming Languages to under graduate students especially students from developing countries.

A critical look at the explosion of Mobile Technology

Overtime, there has been an increase in owning smartphones over personal computers, laptops and other electronic gadgets. This was proven by a survey carried out by Trifonova [49]. The analysis also showed that owning a smart phone is also considered as most important in accessing the Internet, this is shown in Fig. 3 [49]. A study was conducted to show the high level of smart phone ownership compared to other devices. In their study, they found out that students owned smartphones more than other devices as shown in Fig. 4. Results also showed that students agreed to the use of mobile technology for academic purposes as shown in Fig. 5.

Apart from our findings on the acceptance of mobile technology for academic purposes, we also reviewed Mobile Integrated Development Environments (IDEs) for learning Computer Programming. We observed reviews from Google play store which is Google's official store and portal for Android applications, games and other content for Android powered phones or tablets. Two of these IDE seemed to be acceptable and more frequently used.

Mobile Integrated Development Environment (IDE) for learning computer programming

An IDE normally consists of a source code editor, build automation tools, debugger but is not limited only to them. IDE is more than any of these features as it provides the power to connect all of them at one place, besides it, almost all best IDEs have intelligent code auto-completion which suggests what would be the possible syntax you are trying to write at real-time. Based on strengths and constraints, based on our findings from the last section of the methodology – Section 3.1 - the following IDEs are recommended in Table 5.

There are other dedicated Mobile IDEs which includes; Cppdroid, Grasshopper, Mobile C, Coding C, Pydriod etc. These mobile compilers serve as a compiler for one to three related Programming Languages. Some of them offer advanced features like real time tutoring guide, interactive games, graphical punch commands, alarm triggers to remind users to catch up with schedule, etc.

Based on our findings from the five approaches raised in Section 3.2, we recommend that the following computer Languages be taught to undergraduate students in developing countries such as Nigeria. This is shown in Table 6.

Lessons from Universities with Success Stories in Training Programmers

Teach students to sit upright at a desk and a suitable chair, get paper and a pencil.

Teach basic concepts like "sets", trees, graphs, and combinatorics.

Teach them how to represent "computing problems" using these foundational concepts – Lessons from MIT courseware.

Teach them how to extract algorithmically desired computations from these basic "abstract data types".

Ask them to describe these algorithms operationally in pure English away from the computer.

If successful, start to move them onto your favourite programming system like Python etc.

Teach the students to develop their attention span.

Results and discussion

Discussion on approaches

Even though the collaborative learning approach has a lot of successes, there are a few limitations which are highlighted in Table 2 The blended approach adopts a blend of several techniques but depending on the components of the combined solution, it is still bedeviled with limitations such as the need for a laboratory and the need for highly skilled teachers. The experimental approach in spite of all its benefits may not be very practical in a large class. Finally, the integrated learning approach would also work well where there are facilities.

Research questions

Research question 1: What approaches/Models have been identified in previous research for teaching programming Languages in developing countries showing limitations and successes?

This question is answered in Table 2 category approach/strategy limitations-

Research question 2: What were the problems/challenges identified in previous research with current approaches to teaching programming Languages and suggested solutions? This question is answered in Table 3

Research question 3: What model/approach can be proposed that is less costly and can be adopted by any educational institution in developing countries especially after the COVID 19 pandemic? This is answered by our proposed approach in Section 4.0

Research question 4: What improvement can be identified from our proposed approach in particular for:

The use of cost-effective technology in teaching programming Languages to students?

The findings of this study shows that the use of mobile compilers installed on mobile phones using android operating systems is a less costly and an adaptable teaching approach.

Improving the performance of students' in learning programming Languages?

An initial empirical study on the effect of the introduction of mobile compilers for teaching Programming to undergraduate students was carried out. This study was conducted in the class of level 200 during their first semester session. A comparison of the performance of students in basic concepts of programming language before the introduction of mobile compilers and after is shown in Table 7, Figs. 6 and 7.

From our findings, it is clear that there is a significant improvement in the performance of students as shown in Figs. 7 and 8.

Table 5
Recommended mobile IDEs for learning computer programming languages.

S/N	Mobile Application IDE	Strengths	Constraints	Language support and features
1	Programming Hub	Contains lessons for learning various programming languages.		
Free courses are available for beginners. It has a rating of 4.7/5.0. It has a built in online compiler that supports over 20+ programming languages.				
Lessons can go on when offline.	Free courses available are fewer/ limited			
To access intermediate courses and other learning materials, payment for monthly/ yearly subscription is required.	Supports			
C, C++, c#, Java, HTML, Python, Javascript, swift, R and CSS etc.				
2	Solo learn	Contains lessons for learning various programming languages.		
Free courses are available for beginners. It has a rating of 4.6/5.0. It has a built in online compiler that supports over 20+ programming languages. Lessons can go on when offline.				
	To access more courses on programming language, payment for monthly/ yearly subscription is required.	Supports		
C, C++, c#, Java, HTML, Python, Javascript, swift, R and CSS, kotlin, SQL, GIT, Algorithms and Data structures, Ruby, PHP, etc.				
3	Dcoder	It has a rating of 4.4/5.0		
It has a built in online compiler that supports over 20+ programming languages. It has a rich text editor It provides community interactions. It is easy to share program files. Files can be Saved locally and remotely				
	The entire IDE is purely cloud based. This makes the IDE to only work online.			
It requires internet access to function. All users must sign up.	Supports			
C, C++, c#, Java, HTML, Python, Javascript, swift, R and CSS, kotlin, SQL, GIT, Algorithms and Data structures, Ruby, PHP, Rust, Haskell, clojure, Assembly, Scala, Perl, etc				
5	Online Console Compiler	This IDE has the same strengths as the DCoder		
It does not require Internet connections.				
It provides compilers for 23 Programming Languages, with no logical Programming Language compile	It has no online community within the application.	Supports		
C, C++, c#, Java, Python, Javascript, swift, R and CSS, kotlin, SQL, Go Language, Objective C, Ruby, clojure, Scala, Pascal, PHP, etc.				

Table 6
Recommended Computer Programming Language.

Year	Semester	Programming Language to be taught	Goals to be achieved each semester	Justification for choosing a selected language
First year	First semester	Python, Java (Introduction) *(Seminars should also be held with industry programmers, to spur interest, and encourage learning*	To understand the basics of computer programming and be encouraged to commit to learning.	With Python, students understand basic programming concepts more because its syntax is easy and it has a large standard library that enables students to do real applications.
	Second semester		To understand abstract concepts that can be applied to almost any Programming Language.	With Introductory Java, students begin to have a feel of basic programming concepts.
Second year	First semester Second semester	C++ Java (Intermediate)	To understand object oriented programming concepts, libraries, compilation, linkers and pointers To understand and implement principles of algorithms and data structures.	With C++, students are introduced into object oriented programming concepts, compilers, loaders and pointers. With Java, students are able to obtain good knowledge on data structures and algorithms. They will know when to use which data structure and compute the CPU and memory cost of programs written.
Third year	First semester	Hyper Text Markup Language (HTML), Cascading Style Sheet (CSS). Prolog(or any Logic Programming Language Maude or any functional Language	To understand web development and markup languages. To understand how to Program in Logic. This will form the basis for developing intelligent systems.	With HTML, and CSS, Students are able to design and implement web applications. Prolog Maude
	Second semester	JavaScript Structured query language (SQL) Introduction	To flexibly design and implement dynamic web applications. To design and implement database systems	JavaScript is essential for backend development. SQL is the most universal, easy to understand and common used database language. With SQL students can easily design and implement databases that can work with web applications.
Fourth year	First semester	(SQL)- Advanced Advanced Java	To implement advanced database concepts. To implement advanced programming concepts	SQL 2 Java 2
	Second semester	Hypertext Preprocessor (PHP) Common Frameworks like Bootstrap	To understand server-side programming.	PHP is one the best user-friendly Programming Languages. With PHP, students can develop complex dynamic and user-friendly web application.

Table 7
Summary of Performance of Students Before and After the use of a mobile compiler in learning a Programming Language.

Taught Concepts	Duration	Method Employed	Percentage Pass (%)	Percentage Fail(%)	Remarks
Case 1: Outputting data, variable declarations, data types and comments using C++ programming language.	1 hr	Without the use of mobile compiler.	30	70	Very poor
		With the use of mobile compiler. In this case Dcoder.	80	20	Very Good
Case 2: Arithmetic computations and control structures using C++ Programming language	1hr	Without the use of mobile compiler.	23	77	Very poor
		With the use of mobile compiler. In this case Dcoder	80	20	Very good

Conclusion

In conclusion, this approach could improve the performance of students in computer programming courses. In addition to the findings of this study, this research recommends programming languages that should be taught at different levels in the tertiary institutions.

The study performed a systematic review and analysis of findings as expressed by scholars, on various programming pedagogical methods and their effectiveness. In this study, limitations associated with the existing programming teaching

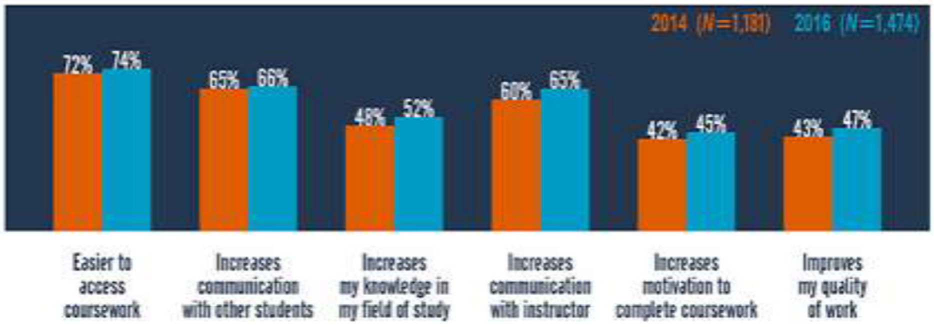

Fig. 6. Acceptance of mobile technology for academic purposes [50].

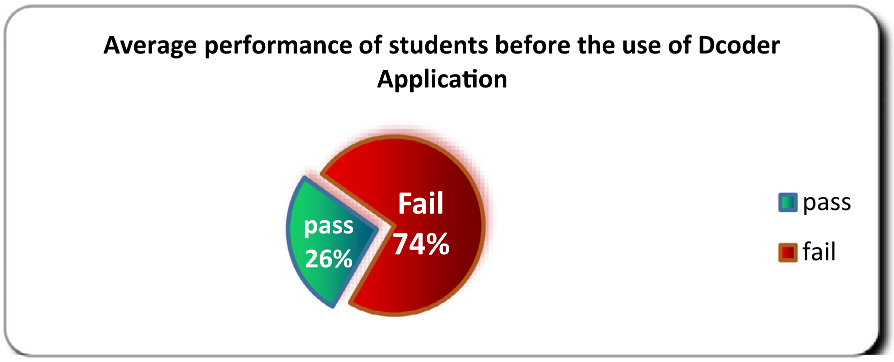

Fig. 7. Average performance of students before the use of Dcoder Application.

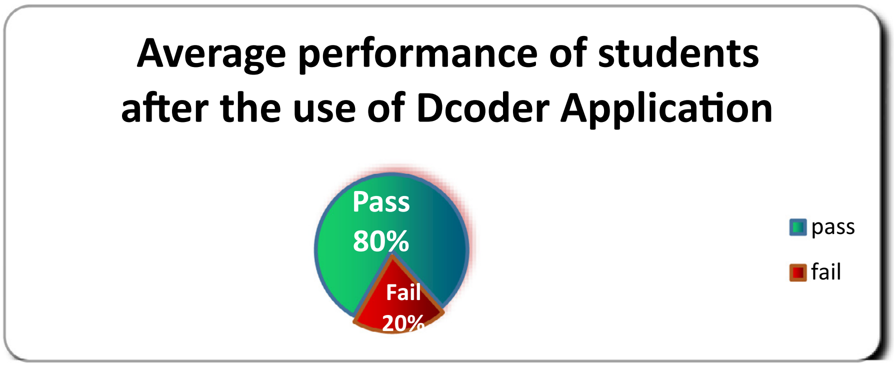

Fig. 8. Average performance of students before the use of Dcoder Application.

methods were identified and summarized in Table 2. This research considered the peculiarity in developing countries and attempted to synergize efficient programming pedagogies that would adapt suitably in these developing countries. The study proposed and tested a more efficient and less costly approach of teaching computer programming language courses to students in tertiary institutions. This approach adopts the blended learning incorporated with the use of mobile compilers (such as Dcoder) as a tool for enabling students practice more. We also emphasize understanding abstract elements of programming language concepts. In addition to this approach, the study recommends programming languages that should be taught at different levels in tertiary institutions. This recommendation is backed up by the research that was carried out to study the University BMAS (Benchmark Minimum Academic Standards) from NUC (National Universities Commission) for Computer Science, to analyze the adopted programming courses in four Universities in Nigeria - (Federal University of Technology Owerri, University of Ibadan, Arthur Jarvis University, University of Calabar, and University of Nigeria), to review a Study from RESEARCHGATE on what programming Languages to teach Since there is need for highly skilled programmers to boost economic development and increase national growth. The approach this research proposes will help teach students to program professionally and to appreciate the value of Computer Programming.

Future work

Further work to the research could be to test this approach for a period of at least two (2) sessions, collate feedbacks, and analyze the results.

Furthermore, an extended systematic review of approaches adopted in developed countries should be done and the approaches that can be adjusted for developing countries can also be tested and tried.

Availability of data and material

Emperical data obtained from Arthur Jarvis University Akpabuyo, Cross River State, Nigeria.

Code availability

Not Applicable.

Funding

Not applicable.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We acknowledge the Arthur Jarvis University for offering us an opportunity to try out our model in teaching programming to its undergraduate students.

References

[1] F. Jejdling, How Technology Can Transform Africa, World Economic Forum, 2015 2015 https://www.weforum.org/agenda/2015/06/how-technology-can-transform-africa/ Accessed 26 March 2020.
[2] G. Anabela, A.J. Mendes, Learning to program - difficulties and solutions, in: Proceedings of the International Conference On Engineering Education, ICEE, 2007, pp. 283–287.
[3] J. Oroma, H. Wanga, F. Ngumbuke, Challenges of teaching and learning computer programming in developing countries: lessons from Tumani University, in: Proceedings of the INTED 2012 Conference, 5th-7th March 2012, Valencia (Spain), 2012.
[4] F. Shahzeb, The 9 most common problems new programmers face, Simple Progr. (2017) https://simpleprogrammer.com/9-common-problems-new-programmers-face/. Accessed 26 March 2020.
[5] J.P. Chimombo, Issues in basic education in developing countries: an exploration of policy options for improved delivery, CICE Hiroshima University, J. Int. Coop. Educ. 81 (2005) 129–152.
[6] A. Damon, P. Glewwe, S. Wisniewski, B. Sun, Education in Developing Countries – What Policies and Programmes Affect Learning and Time in School?, Expertgruppen för biståndsanalys (EBA), Stockholm, Sweden, 2016.
[7] K.A. Sarpong, J.K. Arthur, P.Y. Owusu, Causes of failure of students in computer programming courses: the teacher – learner perspective, Int. J. Comput. Appl. 7712 (2013) 0975–8887.
[8] C. Malliarakis, M. Satratzemi, S. Xinogalos, Educational games for teaching computer Programming, Research On E-Learning and ICT in Education, Springer, New York, 2014.
[9] B.T. Batia, H. Gelderblomb, J. Biljon, A blended learning approach for teaching computer programming: design for large classes in Sub-Saharan Africa, Comput. Sci. Educ. 241 (2014) 71–99.
[10] M. Drini, in: Using New Methodologies in Teaching Computer Programming, IEEE Integrated STEM Conference (ISEC), Princeton, NJ, 2018, pp. 120–124.
[11] S. Dasuki, A. Quaye, Undergraduate students' failure in programming courses in institutions of higher education in developing countries: a nigerian perspective, Electron. J. Inf. Syst. Dev. Ctries. 768 (2016) 1–18.
[12] A. Kerr, Teaching and Learning in Large Class At Ontario Universities: An exploratory Study, Higher Education Quality Council of Ontario, Toronto, 2011.
[13] B. Isong, A methodology for teaching computer programming: first year students' perspective, Int. J. Mod. Educ. Comput. Sci. 9 (2014) 15–21.
[14] C. Bravo, R. Duque, J. Gallardo, J. Garcia, G.P. Ansola, A groupware system for distributed collaborative programming: usability issues and lessons learned, in: Proceedings of the International Workshop on Tools Support and Requirements Management for Globally Distributed Software Development, 2007 p. 6.
[15] S. Goel, V. Kathuria, A novel approach for collaborative pair programming, J. Inf. Technol. Educ. Res. 9 (2010) 183–196.
[16] A.L. Asnawi, A. Ahmad, F.N. Mohammed, K. Ismail, Z.A. Jusoh, N.S. Ibrahim, A.H. Ramli, The needs of collaborative tool for practicing pair programming in educational setting, Int. J. Interact. Mob. Technol. 13 (7) (2019) 201.
[17] O.S. Akinola, B.I. Ayinla, An empirical study of the optimum team size requirement in a collaborative computer programming/learning environment, J. Softw. Eng. Appl. 7 (2014) 1008–1018 Irvine CA 92619-4821, USA http://dx.doi.org/10.4236/jsea.2014.712088.
[18] C.J. Olelewe, E.E. Agomuo, P.U. Obichukwu, Effects of B-learning and F2F on college students' engagement and retention in QBASIC programming, Educ. Inf. Technol. 245 (2019) 2701–2726.
[19] J. Owolabi, O.A. Adedayo, A.O. Amao-Kehinde, T.A. Olayanju, Effects of solo and pair programming instructional strategies on students' academic achievement in visual-basic. net computer programming language, GSTF J. Comput. 33 (2013) 1–4 (JoC).
[20] M.B. Garcia, Cooperative learning in computer programming: a quasi-experimental evaluation of Jigsaw teaching strategy with novice programmers, Edu. Inf. Technol. 264 (2021) 4839–4856.
[21] S.I.&. Malik, J. Coldwell-Neilson, A model for teaching an introductory programming course using ADRI, Educ. Inf. Technol. 223 (2017) 1089–1120.
[22] S. Xinogalos, Designing and deploying programming courses: strategies, tools, difficulties and pedagogy, Educ. Inf. Technol. 213 (2016) 559–588.

[23] I. Yoon, J.&. Kim, W. Lee, The analysis and application of an educational programming language (RUR-PLE) for a pre-introductory computer science course, Clust. Comput. 191 (2016) 529–546.
[24] S.I. Malik, R. Mathew, R. Al-Nuaimi, A. Al-Sideiri, J. Coldwell-Neilson, Learning problem solving skills: comparison of E-learning and M-learning in an introductory programming course, Educ. Inf. Technol. 245 (2019) 2779–2796.
[25] Y. Lu, X. Mao, T. Wang, G.&. Yin, Z. Li, Improving students' programming quality with the continuous inspection process: a social coding perspective, Front. Comput. Sci. 145 (2020) 1–18.
[26] R. Sabarinath, C.L.G. Quek, A case study investigating programming students' peer review of codes and their perceptions of the online learning environment, Educ. Inf. Technol. 255 (2020) 3553–3575.
[27] M. Koorsse, C. Cilliers, A. Calitz, Programming assistance tools to support the learning of IT programming in South African secondary schools, Comput. Educ. 82 (2015) 162–178.
[28] L. Porter, M. Guzdial, C. McDowell, B. Simon, Success in introductory programming: what works? Commun. ACM 568 (2013) 34–36.
[29] J. Tan, X. Guo, W. Zheng, Case-based teaching using the laboratory animal system for learning C/C++ programming, Comput. Educ. 77 (2014) 39–49.
[30] Y. Wang, H. Li, Y. Feng, Y. Jiang, Y. Liu, Assessment of programming language learning based on peer code review model: implementation and experience report, Comput. Educ. 592 (2012) 412–422.
[31] T. Wang, X. Su, P. Ma, Y. Wang, K. Wang, Ability-training-oriented automated assessment in introductory programming course, Comput. Educ. 561 (2011) 220–226.
[32] O.A. Solomon, Computer programming skill and gender difference: an empirical study, Am. J. Sci. Ind. Res. 71 (2016) 1–9.
[33] O. Deperlioglu, I. Kose, The effectiveness and experiences of blended learning approaches to computer programming education, Comput. Appl. Eng. Educ. 212 (2013) 328–342.
[34] S. Djenic, R. Krneta, J. Miti, Blended learning of programming in the internet-age, IEEE Trans. Educ. 542 (2011) 247–254.
[35] S. Mohorovičić, D. Tijan, Blended learning model of teaching programming in higher education, Int. J. Knowl. Lear. 71 (2011) 86–99.
[36] S. Hadjerrouit, Towards a blended learning model for teaching and learning computer programming: a case study, Inf. Educ. 72 (2008) 181–210.
[37] K. Law, V. Lee, Y. Yu, Learning motivation in e-learning facilitated computer programming courses, Comput. Educ. 551 (2010) 218–228.
[38] I.E. Eteng, J.C. Okoro, A collaborative web-based learning forum, Int. j. Nat. Appl. Sci. 812 (2013) 48–55.
[39] I.E. Eteng, S.O. Oladimeji, Development of an online collaboration tool for research and innovation in the university, J. Manag. Sci. Bus. Intell. (2019) 25–31, doi:10.5281/zenodo.3269863.
[40] Hemmendinger, David. "machine language". Encyclopedia Britannica, 13 Oct. 2016, https://www.britannica.com/technology/machine-language. Accessed 8 June 2022.
[41] "Procedural Language." Techopedia, 23 Feb. 2017, https://www.techopedia.com/definition/8982/procedural-language. Accessed 8 June 2022.
[42] Wouk, Kris."5 Functional Programming Languages You Should Know." MUO, 10 May. 2019, https://www.makeuseof.com/tag/functional-programming-languages/. Accessed 8 June 2022.
[43] Bertram, Adam. "Declarative programming." TechTarget, Oct. 2021, https://www.techtarget.com/searchitoperations/definition/declarative-programming?_gl=1%25C3%25971mtiv3d*_ga*LVZKVG4zM1ZHZjE1OGkxZDd6YnMxZldFdVZRNmQzUnZNVnpIdjdRNzJVT200RmZOUTg5MHRSME1WaHJDbW1Dcg. Accessed 8 June 2022.
[44] Prabhu, Rishabh. "Object Oriented Programming (OOPs) Concept in Java." Geeks for Geeks, 13 May. 2022, https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/?ref=gcse. Accessed 8 June 2022.
[45] Adewumi, Adewole. "*Benchmark Minimum Academic Standards for Undergraduate Programme in Nigerian Universities*" http://eprints.covenantuniversity.edu.ng/8483/1/Sciences%20Draft%20BMAS.pdf. Accessed Accessed 21 May 2020.
[46] Sami A. (2017) Do you think it is a good idea to teach Python as the first programming course instead of C++?, https://www.researchgate.net/post/Do_you_think_it_is_a_good_idea_to_teach_Python_as_the_first_programming_course_instead_of_C. Accessed 21 May 2020
[47] Gupta K.(2018) Which is better for coding in algorithms, C++, Python or Java?. https://www.freelancinggig.com/blog/2018/07/20/which-is-better-for-coding-in-algorithms-c-python-or-java/. Accessed 21 May 2020
[48] Palak, Singhal. "Front-end vs. Back-end." Geeks for Geeks, 30 May. 2022, https://www.geeksforgeeks.org/frontend-vs-backend/. Accessed 8 June 2022
[49] Trifonova, Viktoriya. "How Device Usage Changed in 2018 and What it Means for 2019." GWI, 20 Nov. 2018, https://blog.gwi.com/trends/device-usage-2019/. Accessed 8 June 2022.
[50] R. Seilhamer, B. Chen, S. Bauer, A. Salter, L. Bennett, Changing mobile learning practices: a multiyear study 2012–2016, EDUCAUSE Rev. (2018) https://er.educause.edu/articles/2018/4/changing-mobile-learning-practices-a-multiyear-study-2012-2016.