EFFECT OF VIRTUAL CODING SIMULATIONS ON THE PROGRAMMING SKILLS OF SENIOR SECONDARY SCHOOL STUDENTS IN NSUKKA LOCAL GOVERNMENT AREA, ENUGU STATE, NIGERIA

BY

EZENWAKA, MICHAEL CHINEDU
GOU/U22/CSE/060

DEPARTMENT OF SCIENCE AND COMPUTER EDUCATION
FACULTY OF EDUCATION
GODFREY OKOYE UNIVERSITY
UGWUOMU-NIKE, ENUGU STATE, NIGERIA

JULY, 2026

EFFECT OF VIRTUAL CODING SIMULATIONS ON THE PROGRAMMING SKILLS OF SENIOR SECONDARY SCHOOL STUDENTS IN NSUKKA LOCAL GOVERNMENT AREA, ENUGU STATE, NIGERIA

BY

EZENWAKA, MICHAEL CHINEDU
GOU/U22/CSE/060

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF SCIENCE AND COMPUTER EDUCATION, FACULTY OF EDUCATION, GODFREY OKOYE UNIVERSITY, ENUGU, NIGERIA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF SCIENCE (B. Sc.) DEGREE IN COMPUTER SCIENCE EDUCATION

SUPERVISOR: DR. NKIRUKA EZIOKWU

JULY, 2026

APPROVAL PAGE

This project report titled "Effect of Virtual Coding Simulations on the Programming Skills of Senior Secondary School Students in Nsukka Local Government Area, Enugu State, Nigeria" has been approved for the Department of Science and Computer Education, Faculty of Education, Godfrey Okoye University, Enugu, Nigeria, by the undersigned supervisory panel.

Dr. Nkiruka Eziokwu (Project Supervisor)	**Date**
Dr. Vera Ude (Head of Department)	**Date**
Prof. Osuji Ekene Dean, Faculty of Education	**Date**
External Examiner	**Date**

CERTIFICATION

This is to certify that Ezenwaka Michael Chinedu an undergraduate student in the Department of Science and Computer Education with registration number GOU/U22/CSE/060 under the supervision of Dr. Eziokwu Nkiruka has completed the research required for the award of Bachelors of Science (B. Sc.) Degree in computer Science Education at Godfrey Okoye University, and that this project has not been submitted previously for any degree in this, or in any other institution.

_____ _____

Ezenwaka Michael **Date**
GOU/U22/CSE/060

DEDICATION

The entire resource I have built here is solely aimed at my dearly beloved parents, Mr. & Mrs. Ezenwaka for their tireless investments, sacrifices, and spiritual and emotional support in all my academic endeavors.

ACKNOWLEDGEMENTS

It is my pleasure to thank the All-Mighty God for giving me life, good health, and the determination required to complete this demanding research study. It would not have been possible for me to undertake such a challenging academic venture without the unceasing presence, guidance, and grace of God in my life.

I sincerely thank Dr. Nkiruka Eziokwu who supervised me during this research study. Her unceasing intellectual guidance and constructive criticisms changed this research project in totality. Her commitment towards excellence in academics shaped my perspective on research, which will serve me forever in my professional life.

Sincere gratitude is due the Head of Department, Dr V. C. Ude, for her vision, mentorship, and provision of a friendly environment for the students in this program.

Words would be inadequate to express my gratitude to the Dean of Faculty of Education, Prof Osuji Ekene and the entire faculty board, lecturers, and administrative staff. I appreciate for having provided the conducive learning environment and facilities for me to excel in this academic pursuit.

I must acknowledge my indebtedness to the school principals, Computer Studies teachers and the good students of St. Theresa's College, Icon Contemporary College, Shalom Academy, and Royal College in Nsukka Local Government Area. Your hospitality, cooperation, and readiness to participate made the field-testing stage of this work an enjoyable, successful and fulfilling experience.

To my dear brothers and sisters, Sandra, Amarachi and Emmanuel, I owe an indelible debt of gratitude which cannot be adequately expressed in words. Thank you for your unreserved love, prayers, sacrifice and support throughout this academic pursuit.

TABLE OF CONTENTS

LIST OF TABLES

ABSTRACT

This study investigated the effect of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area of Enugu State, Nigeria. The study was prompted by the persistent poor academic performance and low practical proficiency of secondary school graduates in terminal computing examinations, a challenge attributed to the dominant institutional reliance on obsolete, non-interactive "syntax-on-board" instructional methods. Three specific objectives, matching three research questions, guided the study to evaluate overall performance variations and gender-specific achievement metrics. The study adopted a Quasi-Experimental Pre-test, Post-test Control Group Research Design utilizing intact classes to preserve the natural learning setup. The target population comprised all Senior Secondary School Two (SS2) Computer Studies students in the area, from which a sample of 160 students was selected using a purposive sampling technique to filter schools based on laboratory infrastructure, followed by a random assignment of the institutions into experimental and control cohorts (80 students per group). Data collection was executed using a 30-item multiple-choice Programming Skills Achievement Test (PSAT), face and content validated by two measurement and evaluation experts, yielding a high reliability coefficient of 0.86 established via the Kuder-Richardson Formula 20 (KR-20). Collected quantitative data were systematically analyzed strictly using descriptive statistics, specifically the Mean ($\bar{X}$) and Standard Deviation (SD), to trace baseline parameters and terminal performance gains, entirely omitting inferential hypotheses and Analysis of Covariance (ANCOVA) frameworks. The descriptive findings revealed that students exposed to Virtual Coding Simulations achieved a remarkably high post-test mean score ($\bar{X} = 24.68, SD = 1.84$) and a substantial mean achievement gain of 13.44, whereas their counterparts in the conventional control group recorded a low post-test mean score ($\bar{X} = 14.35, SD = 2.46$) and a minor mean gain of 3.17. Gender-isolated evaluations showed that male students in the simulation group attained a post-test mean score of 24.81 (Mean Gain = 13.36) while female students in the same group achieved a post-test mean score of 24.54 (Mean Gain = 13.53). A cross-gender analysis demonstrated that the software intervention benefited both cohorts almost equally, compressing the terminal performance gap to a minimal difference of just 0.27 points. Based on these definitive outcomes, it was concluded that interactive virtual coding simulations successfully demystify programming logic, minimize student cognitive load, and serve as powerful instructional equalizers across gender lines. Consequently, it was recommended among others that Computer Studies instructors completely phase out passive whiteboard lecturing paradigms for practical software topics and that the Enugu State Ministry of Education organize intensive digital capacity-building workshops to equip computer teachers with competencies to deploy lightweight, open-source virtual simulation sandboxes in secondary classrooms.

CHAPTER ONE

INTRODUCTION

Background to the Study

Education in the 21st century has undergone an unprecedented global transformation driven by rapid technological innovations. Of all these innovations, computing education, digital literacy, and interactive learning materials have been identified as some of the most impactful components that have transformed the way people learn, teach, and engage with complicated logical environments (Okon & Asuquo, 2024). In today's knowledge-based economy, having a minimum level of digital literacy in terms of word processing or surfing the web is not enough to give one an upper hand or promote industrialization; instead, high school learners need to become creators of their computations rather than consumers of packaged technologies (Ezenwaka, 2026).

At the foundational core of this digital empowerment is computer education, specifically the acquisition of computer programming skills. The term computer programming entails a complex task of designing, writing, testing, and maintaining source codes for performing specific functions on the computer systems (Ugwu, 2023). In light of this, it is paramount that secondary school students learn programming since the field involves systematic problem-solving, analysis, and decomposition skills. The ability to acquire these technical skills will enable the young minds to come up with local technological solutions and create mobile and web applications while transitioning smoothly into career paths such as software engineering, artificial intelligence, and data science in universities.

Consequently, in educational systems of developing countries such as Nigeria, there has been a focus on computing as part of their national curriculum. As part of the Nigerian Educational Research and Development Council (NERDC), which comes under the Federal Ministry of

Education, Computer Studies is a key component in the curriculum of senior secondary school systems. It is expected that by the end of their senior secondary education, students should be competent in the basic aspects of structuring software, code structure through flow charts, application of conditional and loops, and language syntax (Ogu & Nwachukwu, 2021).

However, the empirical study of computer science classrooms shows that there is an enormous gap between the curriculum goals set for the subject and real classroom practices. The teaching of programming skills in most secondary schools is defined exclusively by outdated passive methodologies, especially in the Nsukka Local Government Area of Enugu State. Teachers rely mainly on lecture methods and the dictation and copying of information from textbooks. Students have to learn dynamic programming commands for a long time on static blackboards without any engagement with a real keyboard, IDE, and compiler, which practice is often called the "syntax-on-board" method of instruction (Odo & Ani, 2022). It turns out that traditional teacher-focused instruction makes computer programming a passive activity and deprives an engineering science of the applied component. Not being able to observe program execution immediately and understand the consequences of the changes made in the code, students experience serious cognitive discomfort, anxiety, and lack of motivation; therefore, it becomes impossible for them to build an adequate model of changes made to variables and loops (Ozioko, 2022).

In order to combat these inherent structural hurdles, remove cognitive roadblocks, and maximize learning efficiency, researchers in the field of educational technology have made a passionate call to introduce virtual coding simulations into institutions. Virtual coding simulations (VCS) pertain to interactive web-based software tools that mathematically simulate live computing processes, emulate reactions of the compiler, and enable users to dynamically write, modify, and

execute their scripts (Chinedu, 2022). In contrast to static methods of instruction, which include slide presentations and written manuals, virtual coding simulations provide a dynamic risk-free environment that enables abstract principles of programming to be concretely illustrated.

The structural architecture of a virtual coding simulation is characterized by several core features designed to support novice learners. First, this system is built on an embedded interactive editor together with a real-time translation compiler, which translates scripts on the go without the need to install any bulky, dependent backend systems. Second, it uses real-time visual output monitoring, which shows the process of the code execution by showing animations for code execution paths and changes in variables' values. Finally, it provides sophisticated error detection capabilities, which do not just detect errors in the code but give informative and scaffolded feedback on what is wrong in the code.

There are several reasons why virtual coding simulations in computer science education matter in many dimensions. In particular, the provision of instant feedback enables these software applications to close the wide gap existing between the definition of syntax and technical application by facilitating learning through experimentation and trial and error. In addition, since such simulation applications are relatively easy to use and can be implemented using simple web browsers or even a local intranet, there is no requirement for any advanced hardware configuration (Chikwendu, 2025).

Across the globe, virtual coding simulations have resulted in revolutionary academic achievements in learning programming. With technologically advanced education systems like that found in countries such as the United States, Estonia, and China, the utilization of simulation tools like Scratch, code.org, and browser-based Python/JavaScript sandboxes is common practice in post primary schools (Dia, 2023). Within such areas, the use of simulations is done as a means

of supporting students in developing their computational skills, which has seen a recorded rise in retention of students, greater interest in pursuing software related careers, and elimination of gender gaps in performance in programming studies. The way programming is taught within such areas is as a logic game, where students get immediate feedback on their algorithmic thoughts.

In Nigeria, the utilization of coding simulations through the virtual world is increasingly being realized within top-tier, well-funded private schools and digital academies within big city centers such as Lagos and Abuja. Within such highly endowed environments, teachers utilize web simulations to involve students in the creation of computer programs, allowing them to develop functional games and web pages even before graduating. Studies carried out within such urban areas reveal that those who are taught using such active compilers perform way better when it comes to syntax and debugging than those taught through papers (Okon & Asuquo, 2024).

However, in the journey down the administrative hierarchy from national level to that of Enugu State, there is a great decline in the application of virtual coding simulation. Though the Ministry of Education, Enugu State has on some occasions provided some rudimentary computer facilities along with general IT literacy classes, the use of executable simulation techniques for teaching programming at secondary school level is virtually unheard of. The computer labs in Enugu State are mainly used for typing along with internet browsing purposes.

This educational crisis is most acute within senior secondary schools across the Nsukka Local Government Area. Notwithstanding the existence of the curriculum mandate, the virtual coding simulation is hardly put into practice by teachers teaching senior secondary school studies in Nsukka. According to the recent localized sources, the reason behind such total lack of practice is connected with a number of institutional challenges, which include an enormous shortage of operational computer systems, the lack of knowledge about the possibility of using open-source

web simulators, as well as the absence of special teacher training for computer teachers (Omeje, 2021). In Nsukka, teachers are not aware of the procedures related to creating, managing, and assessment of their students within the simulated coding environment. As a result, even in those local schools, where some operational desktop computers exist, the computer systems are left unused under a cover from dust and used only for simple lectures on hardware identification without any connection to programming compilers (Odo & Ani, 2022).

As a result of these resource constraints, secondary studies teachers in Nsukka still stick to the conventional method of teaching, which is the syntax-on-board approach. The traditional classroom environment involves the teacher being the sole source of information. The teacher reads the instructions given in obsolete textbooks while sketching out code lines and words on the normal black or whiteboard. The learners are then required to copy down all these pieces of text on their notebooks.

This traditional approach introduces severe systemic problems that stall the development of practical programming skills among learnersFirstly, since the coding language construction necessitates precision and attention to details, transferring codes from a static board to the notebooks entails human mistakes of transcription, resulting in incomplete and faulty code models for students. Secondly, the conventional method strips away the important cognitive loop of debugging, since the chalkboard lacks functionality of executing codes, flashing error messages, and compiling codes. Thus, students cannot comprehend why a certain code arrangement works but not another (Ozioko, 2022). Consequently, such inability to conduct experiments with coding creates an enormous cognitive load for students, forcing them to use excessive brain capacity in trying to visualize how the codes might work (Sweller, 2019). Eventually, such abstract theorization results in enormous frustration, technological phobia, and the total loss of motivation

among the students. It is the reason why secondary school leavers from Nsukka LGA are notorious for poor knowledge of syntaxes, have no capacity to debug even simple software codes, and do not know how to create algorithms.

Crucially, the negative impacts of these conventional learning inadequacies are further amplified and mitigated by another important aspect – the issue of student gender. In the socio-cultural environment of post-primary education in Nigeria, gender has for a long time been an important structural impediment impacting academic achievement, self-efficacy, and professional orientation in the field of STEM subjects. Conventional computer science learning spaces that are dominated by abstract lecture guidelines and challenging coding tasks on a static whiteboard display have for a long time exhibited a clear gender disparity where girls are disadvantaged (Anya & Okoye, 2022). Since out-of-school digital literacy experiences and socialization practices tend to provide more access to electronics to boys, the latter enter secondary school with high self-confidence in computers. On the other hand, when introduced to the daunting and dry 'syntax on board' style of teaching, girl students consistently experience higher levels of cognitive load, coding anxiety, and fear of public embarrassment. The absence of any interactive resources to make software algorithms easier results in most girls adopting surface-level rote learning approach which reinforces a negative societal stereotype that computer programming is a masculine domain.

Against this background, it is important to make a switch from the traditional paradigm of learning computers after primary school through blackboard copying to interactive digital platforms. Through the use of web-based and easy-to-use virtual coding simulations in secondary schools within the Nsukka Local Government Area, scientists will be able to find out whether giving the learners instant feedback through compilers and visual learning will overcome the limitation of resources locally available and cognitive friction and improve their programming

skills. This research aims at exploring the impact of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area of Enugu State, Nigeria.

Statement of the Problem

In recent years, in spite of a constant emphasis on capacity building and computing knowledge in secondary education in Nigeria, the performance of secondary school graduates remains remarkably poor in terms of academic results and practical skill set in computer science. The reports of annual chief examiners' evaluation of candidates' performances in the terminal examinations conducted by the West African Examination Council (WAEC, 2024) show that secondary school candidates face immense problems in coping with test items requiring programming practice. The majority of graduates are totally unable to recognize language syntax, identify errors, and create algorithms (Ogu & Nwachukwu, 2021). These persistent educational shortcomings leave primary school graduates totally ill-prepared for learning software engineering courses at the tertiary education level or taking any technical position in the changing digital world (Ezenwaka, 2026).

Empirical findings from published academic research confirm that the basic problem leading to such systemic failure is due to the fact that educational institutions continue to use outmoded and non-interactive instructional methods in computer systems in secondary schools. As Odo & Ani (2022) assert, the general dominance of the "syntax-on-board method," where teachers write the source codes of dynamic commands on either blackboards or whiteboards in order to memorize them, renders computer science education non-applied. The old-fashioned lecture method makes computer programming a mere grammatical exercise without involving any interactive compiler or actual laboratory manipulation (Ozioko, 2022). In addition to this, available

evidence shows that the teaching issues being experienced are greatly exacerbated by infrastructural inadequacies in the Nsukka Local Government Area. According to Omeje (2021), there are very important shortages of high-end computer hardware and electricity failures on the grid which render offline software development packages totally unusable in local school laboratories.

While interactive, internet-based coding simulations have proven globally to be an effective way of overcoming these constraints imposed by physical hardware and provide adaptive instant feedback from the compiler, there is clear evidence of a great shortage of such use by secondary educators in southeastern Nigeria. The research conducted by Ugwu (2023) showed that secondary school teachers in Enugu State largely refrain from using such simulation platforms because of the absence of training workshops on building digital skills and low level of institutional awareness of alternative light-weighted software solutions. Thus, the possibility to overcome conceptual challenges, free learners' cognitive loads (Sweller, 2019) and improve their programming skills is not utilized at all.

Furthermore, although there are abundant global sources regarding digital sandboxes, there is an utter lack of local empirical findings regarding the efficiency of fully executable simulation programs within post-secondary schools in the Nsukka Local Government Area, especially in terms of their influence on programming sub-skills based on gender lines (Anya & Okoye, 2022). Therefore, the problem of this study, grounded in verified educational challenges, is to investigate the effect of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area, Enugu State.

Purpose of the Study

The overall objective of this study is to determine the impact of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area of Enugu State, Nigeria. Specifically, the study seeks to determine:

1. The mean achievement scores of senior secondary school students taught programming using virtual coding simulations and those taught using conventional method.
2. The mean achievement scores of male secondary school students taught programming using virtual coding simulations.
3. The mean achievement scores of female secondary school students taught programming using virtual coding simulations.

Research Questions

1. What is the mean achievement score of senior secondary school students taught programming using virtual coding simulations and those taught using the conventional method?
2. What is the mean achievement score of male senior secondary school student taught programming using virtual coding simulations and those taught using the conventional method?
3. What is the mean achievement score of female senior secondary school student taught programming using virtual coding simulations and those taught using the conventional method?

Scope of the Study

Geographically, the study was be strictly conducted within the public and privately-owned coeducational secondary schools located in Nsukka Local Government Area of Enugu State,

Nigeria. In terms of content scope, the study was based strictly on effect of virtual coding simulation on computer programming skills.

The level scope is senior secondary two (SS2) students offering computer studies.

Significance of the Study

The findings of this study have both theoretical and practical significance.

Theoretically, this study is anchored on the *Constructivist Learning Theory* by Jean Piaget (1970), the *Cognitive Load Theory* by John Sweller (2019), and *Kolb's Experiential Learning Theory (1984)*. These theories highlight that maximum learning occurs in situations where there is manipulation of the environment by the individual, formation of accurate internal representation of the environment through discoveries, and immediate feedback, while keeping the overload of working memory at bay. Virtual coding simulations provide practical support for these theoretical frameworks by demonstrating how immediate feedback from compilers helps with schema formation and cognitive load reduction.

Practically speaking, after its publication, the results of the current study will prove to be of great value to the following key stakeholders: Students, Computer Studies Teachers, School Administrators, Curriculum Planners and Policy makers, Future Researchers.

The current study will enable students to learn through empirical research about how interactive simulations may help them learn and understand complicated software ideas with the aim of developing their problem-solving skills and engaging in self-regulated learning.

The results of the current study will prove to be of great value to computer education teachers who will learn through practical research how they may teach computer programming using learner-oriented pedagogies that are not lecture-based, and designing an interactive laboratory environment.

For the school administrators, this research will give clear reasons why they should invest in digital school infrastructure, install local servers, and develop lightweight Web-based simulations. The study will be highly beneficial to **curriculum planners and policymakers** in the Ministry of Education by supplying data-driven evidence to guide necessary curriculum reforms, mandating simulation-led coding frameworks in secondary school policy layouts, and funding mandatory tech capacity-building workshops for teachers.

Finally, this study will be advantageous to future researchers because it will serve as a reference tool that offers a strong local base for future empirical studies regarding the integration of interactive technologies and STEM education development in developing countries.

CHAPTER TWO

REVIEW OF RELATED LITERATURE

In this chapter conceptual framework, theoretical framework, review of empirical studies and summary of view of related literature are discussed.

Conceptual Framework

- The Concept of Computer Programming Skills
- Virtual Coding Simulations
- Gender as a Moderating Factor in Technical Learning

Theoretical Framework

- Constructivist Learning Theory (Piaget, 1970; Vygotsky, 1978)
- Cognitive Load Theory (Sweller, 2019)
- Experiential Learning Theory (Kolb, 1984)

Review of Empirical Studies

- Studies on Virtual Learning Tools and Academic Achievement
- Studies on Technical Literacy and Gender Disparities in STEM

Summary of Review of Related Literature

Conceptual Framework

A conceptual framework serves as the structural scaffolding of an empirical study, systematically mapping out the intricate operational pathways, dependencies, and interactions that exist between the independent variable, the multi-dimensional sub-domains of the dependent variable, and the specified moderating variable (Ezenwaka, 2026). In the structure of the above-mentioned research project, the independent variable is seen in the form of an innovative teaching technique known as virtual coding simulations. The dependent variable is viewed as the complete set of programming competencies of the students in the senior secondary school, which is thoroughly dissected into three specific technical sub-domains such as syntax recognition competency, error debugging competency, and algorithmic logical thinking competency (Odo & Ani, 2022). In addition, gender is used as a moderating factor in order to investigate whether the benefits of the simulation technology are equally applicable to both genders in the Nsukka Local Government Area.

Under traditional, in the context of outdated instructional paradigms, such variables remain absolutely independent of each other. In the traditional approach known as "syntax on board," syntax rules are learned in the form of purely textual and passive information, while debugging is taught using the abstraction of verbal dictations, and logical thinking takes the form of static paper and pencil diagrams (Ozioko, 2022). It is impossible to create any coherent cognitive structure due to that. The usage of interactive virtual coding simulations, however, provides a single operational environment in which all such variables work together at once. The syntax chosen by the student controls the results of the programming, any errors in syntax instantly become the source of debugging exercises, and solving them involves continuous improvement of the algorithmic design

as a whole (Chinedu, 2022). This interaction drastically changes the process of information processing and enables novice programmers to develop very coherent technical skills.

The Concept of Computer Programming Skills

The contemporary digital landscape, the acquisition of computer programming skills has evolved from an isolated professional technical capability to a core foundational literacy essential for navigating the complex problem-solving demands of the modern knowledge economy (Okon & Asuquo, 2024). Computer programming proficiency is a complex cognitive skill that entails the ability to analyze a real-life problem, synthesis of logical solution thereof, and the translation of such solution into a very well-structured series of instructions that can be executed flawlessly by a computing machine (Ugwu, 2023). It entails much more than the memorization of language keywords alone but requires the application of complex cognitive skills like abstraction, decomposition, algorithmic thinking, and divergent thinking. Programming proficiency is the key to attaining a higher-order computational mind for a novice secondary senior student.

Programming skills are explicitly included in the structural curriculum requirements for senior secondary schools in Nigeria (Ogu & Nwachukwu, 2021). Students are expected to move from basic computer skills to structural programming approaches, whereby they learn how to manipulate computer software architecture, input/output processing, and how to write functional program codes. By acquiring good programming skills, the student optimizes his/her learning trajectory and makes tertiary STEM courses easy and enables him/her to innovate technological solutions locally.

Virtual Coding Simulations

Virtual coding simulations refer to interactive and technology-driven learning environments which allow for self-directed learning via internet browsers or stand-alone applications in the same

way that professional compilers do (Chikwendu, 2025). In contrast to non-interactive media like slide shows and digital books, virtual coding simulations create an interactive space where one can modify their own lines of codes, execute scripts, and obtain visual feedback on the operation of their programs instantly (Chinedu, 2022). They act as a mediator between abstract computing concepts and real-life actions by turning text into action.

In contexts with limited resources, For instance, virtual simulations are very strategic educational tools in the likes of the Nsukka Local Government Area, since due to the fact that they are based on either web interfaces or local servers, they eliminate the need for costly and cumbersome development tools that need high-end computer machines (Omeje, 2021). The simplification of the environment together with automated scaffolding makes virtual simulations eliminate any unnecessary operation barriers in order for novice learners to dedicate their mental efforts into learning programming.

Gender as a Moderating Factor in Technical Learning

Gender has long been a highly debated moderating factor within STEM education research, particularly in computer science fields across developing regions like Nigeria (Anya & Okoye, 2022). Historically, there has been the occurrence of a gender gap in computer studies classes that utilize conventional teaching techniques whereby male students tend to perform better due to increased access to technology outside class settings. Traditional teaching approaches such as lecturing and coding on whiteboards using texts have the potential of increasing anxiety levels and reducing the self-efficacy of female students.

Introducing highly sensitive, interactive instructional tools like virtual coding simulations aims to create an equitable learning environment (Dia, 2023). Computer-based simulators offer unbiased and non-evaluative responses to each individual learner without any bias from the

classroom. By exploring the moderating effect of gender, it is possible for researchers to know whether the use of interactive technology is an equalizer to enable male and female students to have similar technical skills.

Theoretical Framework

An empirical investigation into computer science pedagogy needs to have a strong theoretical foundation because otherwise it will not be possible to understand the influence of the intervention on cognitive structures. This study builds a multi-dimensional theoretical foundation by integrating three highly complementary pedagogical concepts: Jean Piaget's (1970) and Lev Vygotsky's (1978) *Constructivist Learning Theory*, John Sweller's (2019) *Cognitive Load Theory*, and David Kolb's (1984) *Experiential Learning Theory*. Collectively, all three models create a full picture describing the mechanisms of how novice secondary school students deal with language syntax rules, working memory constraints, and development of systematic bug-fixing and reasoning skills by means of technology.

Constructivist Learning Theory, which originates from the works of Jean Piaget (1970), further advanced by Lev Vygotsky (1978) in his social theories, suggests that knowledge creation is a constructive process through which each individual creates personal mental constructs due to direct interaction with the environment. Constructivist model strongly opposes any teacher-oriented instruction, viewing learners as active agents who are far from being simple receptacles for the information delivered in a lecture (Okon & Asuquo, 2024). In the field of computer science instruction, constructivist learning model claims that programming skills cannot be transferred through whiteboard lectures, but can only be developed through building, testing and fixing of the program code (Ugwu, 2023).

When virtual coding simulations are introduced into the classroom, they achieve this by creating an active learning environment (Chinedu, 2022). When the learner manipulates lines of codes on the simulation software, they develop models of program logic in their mind. If the program does not return the desired results, the simulator gives feedback immediately, causing cognitive dissonance and forcing the learner to question his/her current beliefs and fix the code accordingly (Ozioko, 2022).

Furthermore, the idea of scaffolding introduced by Vygotsky (1978), is directly implemented in modern simulation software by offering syntax suggestions and error explanations. Such assistance acts as a psychological shield for novice learners who are comfortable exploring complicated ideas which would have been impossible to explore without the aid of such a tool (Ogu & Nwachukwu, 2021).

Cognitive Load Theory

The Cognitive Load Theory, formulated by John Sweller (2019), offers a precise cognitive explanation for why traditional whiteboard-based computer programming instruction often fails. According to Sweller (2019), humans have a highly restricted capacity for working memory; it is designed such that it can store only a few discrete chunks of information at a single point in time. These types of loads include the following three categories: intrinsic load (difficulty of the content), extraneous load (difficulties arising from instruction), and germane load (productive processing of instruction).

Computer programming involves a tremendous amount of intrinsic cognitive load due to the fact that students have to deal with both the syntax rules and punctuation at once (Odo & Ani, 2022). If the lecturer adopts the "syntax on board" method, he adds huge extraneous cognitive load on the student's brain. The blackboard is static and thus the student has to devote a lot of effort to

visualizing the way the code works in reality, consuming all the available working memory capacity of his brain (Ozioko, 2022).

Virtual code simulations can help since they take care of the execution of the code and demonstrate it visually (Chikwendu, 2025). The visualization of the way variables change and loop's function takes away extraneous load and helps to save working memory capacity to use it for germane load.

Kolb's Experiential Learning Theory

David Kolb's (1984) Experiential Learning Theory provides an operational theory which explains the process of creation and transformation of knowledge through experiential learning. David Kolb (1984) presents a continuous four-step learning cycle through which one needs to pass in order to achieve mastery. These steps include concrete experience (CE), reflective observation (RO), abstract conceptualization (AC) and active experimentation (AE).

In a conventional computer science classroom where there are only syntax-based classes, students are totally removed from their cognitive cycle. Instead of giving them concrete experience first, they are immediately subjected to abstract conceptualization by teaching them the rules of syntax straight from the book (Odo & Ani, 2022).

An interactive virtual coding simulation serves as an excellent engine for Kolb's (1984) experiential learning cycle. Upon executing a script in the simulator, the learner is presented with an instantaneous Concrete Experience courtesy of the live compiler (Chinedu, 2022). In case of an error, the system alerts the student, who has to carry out Reflective Observation to juxtapose their expectations against the received error messages. The observation allows the learner to understand the nature of the problem, resulting in Abstract Conceptualization whereby the syntax rules regarding the loop or condition block become known to the student. With such a clear

conceptualization in mind, the learner engages in Active Experimentation by reprogramming the block and re-executing it, thus beginning another round of the learning process.

Review of Empirical Studies

Studies on Virtual Learning Tools and Academic Achievement

An increasing body of modern educational research highlights the potential of interactive digital tools to improve learning outcomes across STEM disciplines. Almulla (2023) conducted a comprehensive study examining the impact of constructivist digital tools on university-level classes. With a quasi-experimental design, the study sought to establish the effects of interactive digital systems on academic performance, critical thinking skills, and collaborative problem-solving skills through a selected population of undergraduate students. Empirical results indicated that students who used self-paced, interactive digital technologies performed better in post-test scores and showed more critical thinking skills compared to those instructed using traditional lecture styles (Almulla, 2023). According to Almulla, digital technologies work because they allow passive learning to be converted into active visual exploration, which enables direct manipulation of conceptual variables.

Even though Almulla (2023) presented compelling empirical evidence about the constructivist technology, his study was conducted among university students in an educationally developed environment. There is a geographical gap in the research, since its findings cannot be extended to the educationally developing countries such as Nigeria, where secondary school students do not have any basic exposure to technologies.

Similarly, Hu (2020) examined how the use of AI learning technologies affects the academic performance of postgraduates. In the course of research, which utilized content analysis and a qualitative methodology applied to a group of 300 participants selected from a larger number of

individuals, the efficiency of the usage of digital learning technologies and their impact on the individualization of educational process was analyzed. According to findings, the implementation of interactive software helped increase performance levels through improving the understanding of tasks and self-paced learning (Hu, 2020).

However, the methodology and population used in Hu's (2020) study are definitely creating the gap in research, as the analysis was qualitative and conducted on advanced postgraduates, while for novice secondary school students, who need to acquire basic skills, the quantitative approach is needed.

Studies on Technical Literacy and Gender Disparities in STEM

It is an important topic of research whether the interactive learning technology could be a solution to the gender gap in technical domains in terms of education. In the study conducted by Dia (2023), the impact of interactive digital agents and computer systems on the academic skills and motivation to learn of students was assessed through a mixed methods approach. The study included a sample of students that were taught via automated interactive learning systems. Based on quantitative results, there was a notable improvement in the performance of the students who received interactive learning, and the effect sizes were substantial (Dia, 2023). Importantly, there were no gender differences observed in terms of both motivation and performance.

However, In Dia's (2023) study, there is a high level of technological advancement in education in China, which results in the creation of a geographical divide and reduces the applicability of the study to the social and cultural environment in Enugu State, Nigeria.

Locally, Anya and Okoye (2022) investigated the use of basic educational software in teaching algorithmic logic to secondary school students in the southeastern region of Nigeria. In a quasi-experimental approach, the researchers examined changes in performance upon introduction of

basic interactive screens. The results revealed that using interactive platforms increased conceptual understanding more than a traditional lecturing method (Anya & Okoye, 2022).

The main disadvantage in Anya and Okoye's (2022) study is that the software used was non-executable. Therefore, students were only able to see the logical flow, and they were unable to run their own code or go through the compilation loop. Additionally, their study failed to identify gender as a moderating variable; thus, the gap in research remains on the effect of fully executable virtual coding simulations on the performance of different genders.

Summary of Review of Related Literature

The existing literature establishes that there is evidence that programming skills are fundamental for secondary school learners to cope in the emerging economy (Okon & Asuquo, 2024). Constructivism, Cognitive Load Theory, and Experiential Learning theories advocate strongly for interactive learning in order to deconstruct complex ideas for novice learners (Piaget, 1970; Sweller, 2019; Kolb, 1984). On the other hand, the literature shows that the overwhelming number of secondary schools in Nigeria are still using the conventional "syntax-on-board" teacher-centered approach in teaching computer lessons.

A critical review of the empirical literature highlights several distinct research gaps that this study intends to address:

- **Population and Contextual Gaps:** Most of the literature that has been carried out on virtual simulations is done in university or higher education institutions in developed countries (Almulla, 2023; Hu, 2020). Very little research has been done on novice senior secondary students from sub-Saharan Africa.
- **Content and Technological Gaps:** Most of the studies carried out locally within the Nigerian environment have depended heavily on static digital methods, such as the use of

PowerPoint presentation or simply designing a layout of presentation (Anya & Okoye, 2022). Such static methods do not have coding or compilation capability.

- **Analytical and Sub-Skill Gaps:** Most The majority of the local literature relies on measuring academic grades as an aggregated metric without dissecting the technical skills into its sub-domains (Chikwendu, 2025). Localized information is not available regarding the impact of simulations on syntax knowledge, bug fixing, and algorithm design as individual skills.

This study directly addresses these gaps by conducting a systematic, quasi-experimental investigation within actual secondary classrooms in the Nsukka Local Government Area of Enugu State. This research makes use of a complete and executable web-based simulation environment and careful measurement of changes made on an individual basis to specific technical skills to bridge theoretical educational principles and field application. This research is ultimately seeking to prove the validity of a teaching method that saves novice secondary school students from rote learning.

CHAPTER THREE

RESEARCH METHOD

In this chapter, we will outline the methods used to conduct this empirical investigation. Specifically, these includes area of the study, population of the study, sample and sampling techniques of the study, instrument for data collection, validation of instrument, reliability of instrument, method of data collection, method of data analysis are discussed.

Design of the Study

This empirical study used a quasi-experimental non-equivalent pre-test post-test control group design. This unique research design is ideal for conducting educational research due to its ability to help researchers carry out tests on real-world educational practices without disrupting the flow of administration in the school by randomly grouping participants. In such a case, the participants will form naturally occurring intact groups that remain untouched from their usual classroom settings. This will ensure that the research has high ecological validity, where extraneous factors, such as student development, can be controlled.

This design is considered appropriate because it ensured that not random individuals were selected from their respective classes but rather, intact classes were selected and allotted either for the simulation-based treatment or the lecture method. The same test was administered to both groups before the treatment, creating a benchmark of their skill level when it came to programming. A teaching period was conducted for multiple weeks, after which the same test was taken by both groups. In this way, it will be possible to assess the difference between the two test results, thereby determining the impact of the interactive software.

The quasi-experimental design is graphically illustrated below:

Experimental Group: O1 → X → O2

Control Group: O3 → - → O4

Where:

O1 and O3 = Pretest for experimental and control groups respectively

X = Intervention (virtual coding simulations)

O2 and O4 = Posttest for experimental and control groups respectively

- = No intervention (traditional teaching approach)

Area of the Study

This research project was carried out at secondary schools in Nsukka Local Government Area of Enugu State, Nigeria. Nsukka LGA is known for its rich academic history, having the University of Nigeria, Nsukka (UNN), among other universities and secondary schools. It comprises a variety of public schools, mission schools that are relatively well-resourced, and many new privately owned commercial schools. In this regard, it can be said to be an exact representation of the whole secondary school population in the state of Enugu.

Choosing Nsukka Local Government Area as our research location is very strategic, owing to how it reflects the actual difficulties that most developing country schools have in terms of infrastructure (Ezugwu, 2024). At such locations, schools have to deal with all sorts of things ranging from power outages to computer laboratories that lack adequate facilities, thereby making it difficult for teachers to utilize anything other than syntax-on-board techniques (Omeje, 2021). This way, our research becomes more applicable to all common schools in Nigeria as we test the effectiveness of our simulations in a real-world scenario (Chikwendu, 2025).

Population of the Study

The study involved a large sample of about 1,850 Senior Secondary Two (SS2) students from all 32 public and approved private secondary schools in addition to 1,900 secondary school teachers found in Nsukka Local Government Area (According to: Enugu State Post-Primary Schools Management Board [PPSMB], Annual Census Report, 2025). The reason why this particular group of SS2 students was selected is due to the fact that they had undergone computer literacy courses and were set to be introduced to basic principles of programming under the national curriculum (Federal Ministry of Education, 2020).

Sample and Sampling Technique(s)

The sample for this study consisted of 160 Senior Secondary Two (SS2) Computer Studies students selected from four specific secondary schools within the Nsukka Local Government Area. To select this sample, a Purposive Sampling Technique was first employed to deliberately choose four target schools based on specific infrastructure criteria:

1. The schools must offer Computer Studies as a registered exam subject for senior secondary classes.
2. The schools must possess a functional computer laboratory equipped with basic, operational desktop units or laptops.
3. The schools must run co-educational layouts (enrolling both male and female students) to allow for the analytical moderation of gender parameters.

The four schools selected through this purposive filtering were St. Theresa's College, Icon Contemporary College, Shalom Academy, and Royal College, all located within Nsukka LGA.

Following the selection of these four secondary schools, Intact Classes were utilized within each school to prevent the administrative disruption of reshuffling students. To assign the treatment

conditions equitably, a simple Random Sampling Technique (by coin toss) was used to group the four selected schools into experimental and control groups:

* Experimental Group (Virtual Coding Simulations): Two schools (comprising 2 intact classes with a combined total of 80 students, 42 male and 38 female) were randomly assigned to the experimental treatment group. Students in these classrooms interacted directly with web-enabled, executable coding simulators.

* Control Group (Syntax-on-Board Paradigm): The remaining two schools (comprising 2 intact classes with a combined total of 80 students, 38 male and 42 female) were assigned to the control group. These students were taught the exact same topics but through traditional theoretical lecturing and blackboard syntax copying.

Instrument for Data Collection

The specific tool utilized for collecting the necessary data for this empirical research is called the Programming Skills Achievement Test (PSAT). In order to obtain accurate and precise information, the test contained 30 items that were precisely formulated. Instead of focusing merely on providing basic definitions, the test was deliberately divided into two fundamental technical sub-domains that reflect the major tasks of programming: (a) Syntax Recognition Skills (Nnaji, 2022), (b) Error Debugging Skills. and (c) Algorithmic Logic Skills (Ugwu, 2023).

All the test questions were designed based on the multiple-choice question model where there is absolutely no possibility of having grader subjectivity or any other form of biasness that would render the assessment subjective. Each right answer carried a point value of one point (1 mark) while each wrong answer got nothing (0 mark) resulting in a maximum score of 30 points. It is through this fine-grained measurement that the researcher was able to obtain highly detailed quantitative data on which aspects of technical subskills were easily acquired and which weren't.

Validation of Instrument

To ensure that the tool used in the assessment was absolutely valid, reliable, and precise in measuring exactly what it is supposed to measure, the first draft of the PSAT was submitted to one lecturer of the Department of Science Education, Godfrey Okoye University, with specialization in computer education and educational measurement; and One computer science teacher from St. Theresa's College, Nsukka LGA for proper validation. The experts carefully ensured that all questions were clear linguistically, technically correct, and based on the SS2 syllabus. All comments and suggestions from the validators were thoroughly incorporated in the final version of the instrument.

Reliability of Instrument

For purposes of ensuring stability and dependability of results obtained from the administration of the PSAT, an additional pilot test involving another sample of 20 students from SS2 Computer Studies who came from a different school, not included in the research sample, was done. Scores obtained from this pilot test were subjected to reliability analysis through Kuder-Richardson Formula 20 (KR-20). The reliability index obtained from the statistical analysis was 0.84. Considering that a value of 0.70 or higher is always considered highly reliable in educational sciences, the PSAT is considered dependable and reliable.

Method of Data Collection

The Field Research was conducted in a systematic manner within a period of exactly six weeks. The data was collected following a strict procedure from the selected secondary school:

1. First week (Pre-test Period): The Programming Skills Achievement Test (PSAT) was conducted as a pre-test to the students in both experimental and control groups to determine their existing programming skills.

2. Weeks 2 to 5 (Learning Period): The course content was divided into four programming modules that were taught each week separately. These include Basic Variables, Punctuation Rules, Conditional Logic, and Loop Structure. While the experimental group learned the course through virtual simulation on the school networks, the control group learned the same through traditional classroom theoretical learning and blackboard syntax writing. The Computer Studies teachers of the selected secondary schools were the primary instructors and research assistant for this study.

3. Week 6 (Post-testing Phase): Both groups had the same PSAT test instrument of 30 items applied again to gauge their performance after the intervention.

Control of Extraneous Variables

1. Instructional Bias: Lesson plans were equally standardized for all the instructors, thus uniformity in content and duration (40 minutes per period) was ensured in both the groups.

2. Hawthorne Effect: Naturally occurring intact classes were used, and the classroom teachers instructed them about the lesson without making them realize that they were part of an experiment.

3. Contamination: Schools selected were geographically separate and from different parts of Nsukka Local Government Area, thereby minimizing contamination.

4. Pre-test/Post-test Sensitization: There was a four-week instructional gap between the two tests, and the order of 30 multiple-choice questions and answer choices (A, B, C, D) was shuffled in the post-test.

Method of Data Analysis

The quantitative data collected from the implementation of the pre and post-tests of the Programming Skills Achievement Test (PSAT) were systematically classified, coded, and

analyzed using the Statistical Package for the Social Sciences (SPSS) software, version 26.0. In order to answer the three descriptive research questions that guide this study, the main statistical methods that were used are the mean ($\bar{X}$) and the standard deviation (SD). The Mean (): The arithmetic mean was utilized in calculating the initial level of performance of the subjects in experimental and control groups before the administration of the test. Mean scores was also be calculated after treatment to determine the final ability of the learners. Notably, Mean Achievement Gains (absolute arithmetic difference between the post and pre-test means) was be determined for each group:

The success achieved was used as the key descriptive criterion for the determination of whether the active virtual coding simulation environment delivers greater skills benefit than the conventional syntax-dominant "syntax-on-board" approach.

1. The Standard Deviation (): A standard deviation was obtained along with the mean scores in order to analyze the dispersion, variation, and homogeneity of the individual scores of the students relative to the average scores. The smaller the standard deviation after the test in a certain group, the more homogeneously the technical skills of the students are distributed relative to the mean, which shows that the educational impact was uniform for all of them.

CHAPTER FOUR

RESULTS AND DISCUSSION

In this chapter, presentation of findings, interpretation of finds and summary of finds are treated.

Presentation of Results

Research Question 1: What is the mean achievement score of senior secondary school students taught programming using virtual coding simulations and those taught using the conventional method?

To answer this research question, the pre-test and post-test scores for programming achievements were analyzed from the two groups of students; Virtual Coding Simulations (experimental group) and Conventional Syntax-on-Board method (control group). The summary of this analysis is presented in Table 1.

Table 1: Mean and Standard Deviation of Pre-test and Post-test Programming Achievement Scores of Students in the Experimental and Control Groups

Group	N	Pre-test $\bar{X}$	Pre-test SD	Post-test $\bar{X}$	Post-test SD	Mean Gain
Experimental (VCS)	80	11.24	2.15	24.68	1.84	13.44
Control (Conventional)	80	11.18	2.21	14.35	2.46	3.17

Source: Field Research, 2026 (Maximum Score = 30)

1. Interpretation of Data in Table 1

The data presented in Table 1, the mean achievement score before the experiment was 11.24 and the standard deviation was 2.15 for the experimental group taught with the help of virtual coding simulation, whereas the mean achievement score before the experiment was 11.18 and the standard deviation was 2.21 for the control group taught with the conventional technique. This means that the baseline mean scores were very similar and close to each other, meaning that both groups had almost equal programming skills before the experiment.

However, after undergoing the instructional intervention, the mean achievement score of the experimental group increased significantly to 24.68, with a standard deviation of 1.84, leading to a mean achievement gain of 13.44. On the other hand, the mean achievement score of the control group increased marginally to 14.35, with a standard deviation of 2.46, giving a mean achievement gain of 3.17. A comparative analysis of the results reveals that learners who underwent interactive virtual coding simulation performed much better in programming tests than those subjected to the traditional approach. Moreover, the post-test standard deviation of the experimental group (1.84) was smaller than that of the control group (2.46), showing that individual test scores of the learners taught through the simulation approach were closely grouped around the high mean score.

Research Question 2: What is the mean achievement score of male senior secondary school students taught programming using virtual coding simulations and those taught using the conventional method?

To answer this research question, the pre-test and post-test programming achievement scores of the male student cohorts across both the experimental and control groups were isolated and analyzed. The summary of the mean scores and standard deviations for the male participants is presented in Table 2.

Table 2: Mean and Standard Deviation of Pre-test and Post-test Programming Achievement Scores of Male Students

Group	N	Pre-test $\bar{X}$	Pre-test SD	Post-test $\bar{X}$	Post-test SD	Mean Gain
Experimental (Male)	42	11.45	2.08	24.81	1.76	13.36
Control (Male)	38	11.32	2.14	14.58	2.38	3.26

Source: Field Research, 2026 (Maximum Score = 30)

2. Interpretation of Data in Table 2

Table 2 illustrates where the academic performance indicators have been limited to only the male student participants. It can be observed that while the experimental male participants' average pre-test mean was 11.45 (SD=2.08), their post-test mean performance indicator was 24.81 (SD=1.76), which was an impressively high score.

In sharp contrast, the male students assigned to the control group started with a baseline pre-test mean score of 11.32 ($SD = 2.14$) and finished with a post-test mean score of 14.58

($SD = 2.38$), resulting in a marginal mean gain of 3.26. This dynamic descriptive data establishes that the interactive simulation intervention is highly effective for male learners, facilitating a significantly larger academic improvement in core programming sub-skills (syntax recognition, error debugging, and algorithmic logic design) than the conventional whiteboard copying paradigm.

Table 3: Mean and Standard Deviation of Pre-test and Post-test Programming Achievement Scores of Female Students

Group	N	Pre-test $\bar{X}$	Pre-test SD	Post-test $\bar{X}$	Post-test SD	Mean Gain
Experimental (Female)	38	11.01	2.22	24.54	1.92	13.53
Control (Female)	42	11.05	2.28	14.14	2.52	3.09

Source: Field Research, 2026 (Maximum Score = 30)

3. Interpretation of Data in Table 3

Table 3 documents the academic performance metrics isolated specifically for the female student population. The data reveals that female students in the experimental group possessed a baseline pre-test mean score of 11.01 ($SD = 2.22$) and achieved a post-test mean score of 24.54 ($SD = 1.92$), creating a substantial, transformative mean achievement gain of 13.53.

Conversely, the female students in the control group recorded a pre-test mean score of 11.05 ($SD = 2.28$) and finished with a post-test mean score of 14.14 ($SD = 2.52$), showing a minor mean gain of 3.09. A comparative assessment of the mean gains reveals that female students

exposed to virtual coding simulations improved by an average of 13.53 points, whereas their female counterparts exposed to conventional methods only improved by 3.09.

Importantly, comparing the post-test mean scores of the experimental males (24.81, Table 2) and experimental females (24.54, Table 3) demonstrates that the simulation software benefited both genders almost equally, showing a compressed gap of just 0.27 points. This demonstrates that the interactive simulation environment effectively supports learning for both male and female students when they are removed from the constraints of the traditional syntax-on-board classroom layout.

Summary of Findings

Based on the descriptive statistical analysis of the data collected from the field experiment, the following findings are summarized:

1. Senior secondary school students taught computer programming concepts using Virtual Coding Simulations (VCS) achieved a higher post-test mean programming achievement score ($\bar{X} = 24.68, SD = 1.84$) and a higher mean achievement gain (13.44) than students taught using the conventional syntax-on-board method ($\bar{X} = 14.35, SD = 2.46$; Mean Gain = 3.17).
2. Male senior secondary school students taught programming using Virtual Coding Simulations achieved a substantially higher post-test mean achievement score ($\bar{X} = 24.81, SD = 1.76$) and a higher mean achievement gain (13.36) than male students taught using the conventional syntax-on-board technique ($\bar{X} = 14.58, SD = 2.38$; Mean Gain = 3.26).
3. Female senior secondary school students taught programming using Virtual Coding Simulations achieved a high post-test mean achievement score ($\bar{X} = 24.54, SD = 1.92$)

and a higher mean achievement gain (13.53) than female students taught using the conventional method ($\bar{X} = 14.14, SD = 2.52$; Mean Gain = 3.09).

4. In a cross-gender analysis of the experimental group, it was found that the use of simulation software was beneficial for both male and female groups nearly equally, narrowing down the gap between their performance on the terminal post-test to 0.27 marks only, which shows that virtual coding simulations are an effective teaching equalizer.

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

Discussion of Findings

The descriptive statistical findings obtained from this study demonstrate that the integration of interactive virtual coding simulations structurally optimizes the programming skill acquisition of senior secondary school students compared to traditional teaching methods. Baseline Similarities among Pre-test Means Scores across the Three Tables show that the existing programming skills of students from the Nsukka Local Government Area were relatively the same, hence poor, thus empirically resulting from an over-reliance on purely theoretical and text heavy lecturing parameters (Odo & Ani, 2022). The large mean achievement of 13.44 gained from the experimental group compared to the small achievement of 3.17 of the control group proves that active digital sandboxes are better than the old syntax-on-board learning method in secondary computer studies.

Such academic gains fit well with John Sweller's (2019) Cognitive Load Theory because the old method overburdens the students' working memory since they have to visually imagine dynamic code executions from static blackboards (Ozioko, 2022). Virtual coding simulators not only facilitate automated execution of codes but also provide instant output feedback to the learners visually, thereby eliminating the unnecessary cognitive load from the mind and freeing it to focus on algorithm designing. Moreover, the very small gap of just 0.27 points between the post-test mean scores of the experimental group of male and female students (24.81 vs 24.54), respectively, proves beyond doubt that interactive technology is an excellent equalizer in teaching (Dia, 2023). In a neutral setting where the learners can experiment and learn through trial and error, both male and female student groups demonstrate similar technical proficiency skills.

Conclusions

Given the evident, consistent data gathered from the quasi-experiment, one can safely argue that teaching computer programming via virtual coding simulations is an extremely efficient means compared to the outdated whiteboard approach. While the syntax on board method remains popular at local educational institutions, it is absolutely obsolete and cannot accommodate any of the modern realities associated with software development (Odo & Ani, 2022). The approach forces learners into a vicious circle of mindless learning and renders them totally unfit for dealing with actual technical problems (Attah & Onyishi, 2023).

It is also clear that interactive simulation software has been able to succeed due to its seamless compatibility with the natural learning process in the human brain (Sweller, 2019). Through removing the load of visualizations from the working memory of the student and putting them through the trial-and-error experience, such interactive tools have been successful in achieving excellence academically within all technical subskills (Almulla, 2023). Most importantly, considering their success with both boys and girls, these educational aids provide a non-partisan option for education boards to use as an effective solution in bridging the gender gap in STEM subjects in Enugu State (Enugu State Ministry of Education, 2023).

Educational Implications of the Study

The results of structural analyses obtained in the course of this study have evident implications for education in regard to future development of computer science teaching methodologies. The continued use of outdated theoretical methods such as the syntax-on-board method would mean that schools keep training students who are lacking in the necessary skills of software engineering and who would not be prepared for STEM classes at a tertiary education level or the modern digital economy. However, the use of virtual coding simulations shows that interactive visualization

techniques can effectively overcome the limitations imposed by real-world laboratory equipment and cognitive load.

Limitations of the Study

However, while this research has provided some descriptive knowledge that can be gained from this work, it has some constraints. For instance, geographically, this experiment was conducted only among selected secondary schools in Nsukka Local Government Area in Enugu State and not in computer studies classes located in very urban areas or very rural areas of Nigeria. In terms of content, this study considered only fundamental programming principles found in the SS2 syllabus; hence, its results cannot be generalized to other more advanced software applications.

Recommendations for Action

In light of the conclusive results of this research, the following recommendations are therefore advanced:

1. The Computer Studies teachers should totally eliminate the practice of using passive lecture methods in coding, and instead use online virtual coding simulations and sandboxes within their weekly teaching schedules.
2. The Enugu State Ministry of Education and Post-Primary Schools Management Board (PPSMB) should conduct capacity building seminars aimed at equipping secondary school computer teachers with the necessary skills to use, manage, and assess students' performance through the use of lightweight open-source virtual simulations.
3. The school authorities should work to improve the current computer lab facilities by setting up the internet or intranet software applications in order to give students continuous access to an active compilation environment.

Suggestions for Further Research

As part of recommendations for future research into expanding the results of this study, the following points have been made:

1. It would be advisable for there to be a repetition of this study over a larger geographical region that would comprise several local government areas and even states.
2. There should be further research into the retention period of virtual coding simulations with regards to the programming ability of students through longitudinal studies.
3. There should be a comparative study of the effectiveness of visual-based simulations like Scratch and text-based sandbox environments like Python/JavaScript editors on the logical thinking skills of junior and senior secondary school students.

Summary of the Study

This study investigated the effect of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area, Enugu State, Nigeria. The educational problem that prompted the study was the continual low practical skills and academic achievements of secondary school graduates in the terminal computer examination, owing to the use of outmoded "syntax-on-board" instructional techniques. The study was anchored on three clear research questions dealing with performance differences between the overall and specific genders of learners who underwent both conditions.

The quasi-experimental pre-test, post-test control group design was used, involving a selected sample of 160 Senior Secondary Two (SS2) Computer Studies learners chosen purposefully from co-educational institutions and randomly assigned into experimental and control groups (80 learners each). The data were collected through a valid 30-item Programming Skills Achievement Test (PSAT) having a high reliability score of 0.86. Data obtained were analyzed descriptively

through Mean ($\bar{X}$) and Standard Deviation (SD) only, without any hypothesis testing and ANCOVA analysis.

The descriptive findings revealed that:

1. The students that were taught through virtual coding simulations showed a significantly higher post-test mean score (24.68) and mean gain (13.44), when compared to the students that were taught using the traditional method of learning (post-test Mean = 14.35, Mean Gain = 3.17).
2. The male students in the simulation class showed a high post-test mean score of 24.81 and a mean gain of 13.36, as opposed to control male students whose post-test mean was 14.58 and mean gain was 3.26.
3. The female students in the simulation class showed a post-test mean score of 24.54 and a mean gain of 13.53, while the control female students had a post-test mean of 14.14 and a mean gain of 3.09.
4. The simulation intervention positively impacted both genders almost equally by showing a highly compressed difference of 0.27 points between experimental males and females.

REFERENCES

Aboyade, W. A., Oladokun, B. D., & Aboyade, M. A. (2025). Awareness and utilization of plagiarism checker for effective research output among undergraduate students in universities in Kogi State, Nigeria. *The Serials Librarian,* 86(1), 45-58.

Adachi, S., Hayashi, K., Shimamura, E., & Iba, T. (2023). *Academic writing patterns: A pattern language for writing creative research papers.* Conference on Pattern Languages of Programs, 30(2), 112-128.

Agbaji, J. C., & Eze, C. O. (2021). Assessing technical literacy and digital skills acquisition paths in Nigerian public schools. *African Journal of Science and Computer Education,* 14(3), 201-215.

Almulla, M. A. (2023). Constructivism learning theory: A paradigm for students' critical thinking, creativity, and problem solving to affect academic performance in higher education. *Cogent Education,* 10(1), 214-228.

Alua, M. A., & Asiedu, N. K. (2023). Students' perception on plagiarism and usage of Turnitin anti-plagiarism software: The role of the library. *Journal of Library and Information Services,* 12(4), 310-322.

Amirjalili, F., Neysani, M., & Nikbakht, A. (2024). Exploring the boundaries of authorship: A comparative analysis of AI-generated text and human academic writing in English literature. *Frontiers in Education,* 9(1), 88-102.

Anya, C. U., & Okoye, F. N. (2022). Gender disparities and self-efficacy levels in post-primary STEM classrooms within South-Eastern Nigeria. *Journal of Science and Computer Education*, 11(2), 142–155.

Astuti, D., Darmahusni, D., & Sumarni, S. (2023). The use of Grammarly in the academic writing of undergraduate students: Advantages, weaknesses, and challenges. *English Language and Education Journal*, 8(2), 145-159.

Attah, F. N., & Onyishi, A. I. (2023). Investigating cognitive friction and problem-solving barriers in secondary school algorithms modules. *Journal of Science Education and Technology,* 19(2), 99-114.

Bao, T., Zhao, Y., Mao, J., & Zhang, C. (2025). Examining linguistic shifts in academic writing before and after the launch of generative AI models. *Journal of Higher Education and Informatics,* 17(1), 12-29.

Chikwendu, U. S. (2025). The efficacy of virtual compile environments in secondary computer studies instruction. *West African Journal of Educational Technology,* 22(1), 40-56.

Chinedu, S. (2022). Global perspectives on digital simulation platforms and computational reasoning in emerging economies. *International Journal of STEM Learning*, 13(4), 410–425.

Dia, Y. (2023). Interactive virtual agents and digital systems: Mitigating gender disparities in modern technical fields. *Journal of Educational Technology & Society*, 26(3), 88–103.

Enugu State Ministry of Education. (2023). *Annual educational statistics report and performance indices in STEM frameworks.* Enugu State Government Press.

Eze, M. I., & Okechukwu, P. A. (2023). Paradigm shifts in computer engineering and coding literacy within secondary schools. *Nigerian Journal of Computer Education,* 11(2), 77-91.

Ezeilo, K. C. (2023). Examining standard performance markers in Enugu state secondary technical assessments. *Journal of Educational Evaluation,* 15(3), 180-195.

Ezenwaka, M. C. (2026). Digital workspace demands and secondary computer studies alignment: Bridging the skills gap. *Nigerian Journal of Educational Research*, 22(1), 15–31.

Ezenwaka, M. C. (2026). *Effect of virtual coding simulations on the programming skills of senior secondary school students in Nsukka Local Government Area, Enugu State, Nigeria* (Unpublished bachelor's project report). Godfrey Okoye University, Enugu, Nigeria.

Ezugwu, J. N. (2024). High school technical facilities and local instruction models in Nsukka LGA schools. *Nsukka Journal of Educational Studies,* 31(1), 54-69.

Federal Ministry of Education. (2020). National curriculum for senior secondary schools: Computer studies modules. *Nigerian Educational Research and Development Council (NERDC).*

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development.* Prentice-Hall.

Nnaji, S. E. (2022). Algorithmic design, logical flowcharts, and introductory loops acquisition paths in secondary computer classes. *African Educational Architectures,* 9(2), 142-156.

Obi, C. O., Agwu, E. K., & Chinedu, S. (2022). Global perspectives on digital simulation platforms and computational reasoning in emerging economies. *International Journal of STEM Learning,* 13(4), 410-425.

Odo, L. C., & Ani, P. O. (2022). The syntax-on-board instructional technique versus active laboratory computing paradigms. *Journal of Professional Teacher Education*, 24(1), 89-104.

Ogu, R. N., & Nwachukwu, O. T. (2021). Systemic bottlenecks in Nigerian computer science secondary layouts. *West African Journal of Curriculum Development,* 18(3), 112-127.

Okon, E. B., & Asuquo, M. E. (2024). Engineering early access: Re-structuring the secondary curriculum for software architectural demands. *African Journal of Digital Innovation*, 6(1), 22-38.

Omeje, I. A. (2021). Electrical micro-grids, laboratory downtime, and cognitive performance indices in post-primary computer tasks. *Nsukka Scientific Forum,* 19(2), 101-115.

Ozioko, A. N. (2022). Dynamic software design versus teacher-centered lectures in post-primary technical modules. *Enugu State Educational Outlook,* 14(2), 130-144.

Piaget, J. (1970). Science of education and the psychology of the child. Orion Press.

Sweller, J. (2019). Cognitive load theory and educational technology. *Educational Technology Research and Development,* 67(2), 261-274.

Ugwu, C. D. (2023). Concrete visualization paths in abstract software language instruction. *Journal of Computer and Science Instruction,* 16(2), 205-219.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard University Press.

West African Examinations Council. (2024). *Chief examiners' diagnostic report: Computer studies and practical applications.* WAEC Press.

APPENDIX A

VALIDATION OF RESEARCH INSTRUMENT

Department of Science and Computer Education
Faculty of Education
Godfrey Okoye University, Enugu
March, 2026

Dear Sir/Madam,

I am Ezenwaka Michael Chinedu, an undergraduate student from the Department of Science and Computer Education, Faculty of Education, Godfrey Okoye University, Enugu. I am undertaking an undergraduate research study.

I am currently carrying out a research project titled: **"Effect of Virtual Coding Simulations on the Programming Skills of Senior Secondary School Students in Nsukka Local Government Area, Enugu State, Nigeria"**.

To achieve the goals of this study, I have designed a **Programming Skills Achievement Test (PSAT)**, a **Table of Specification (TOS)**, and **Instructional Lesson Plans**.

As an expert in this field, you are kindly requested to evaluate these instruments for face and content validity. Please review the following items in terms of clarity of language, content, and relevance to the research questions. Your comments and overall judgment will be very helpful in improving these documents.

Copies of the research questions, PSAT, TOS, and the lesson plan are attached.

Thank you for your time and cooperation.

Yours faithfully,

Ezenwaka Michael Chinedu
Researcher

APPENDIX B

REQUEST FOR PARTICIPATION IN RESEARCH STUDY

Department of Science and Computer Education

Faculty of Education

Godfrey Okoye University

Thinkers Corner, Enugu State

March, 2026

Dear Sir/Madam,

I am Ezenwaka Michael Chinedu, I am a student at Godfrey Okoye University, Enugu. This is a research study entitled: "Effect of Virtual Coding Simulations on the Programming Skills of Senior Secondary School Students in Nsukka Local Government Area, Enugu State, Nigeria".

I want to ask for your permission to use your Senior Secondary School 2 (SS2) Computer Studies students along with their teacher in this research. The procedure is very simple, including a short baseline test, a short coding lesson and finally a post-test.

I can assure you that all information gathered will be treated in utmost confidentiality and will be used only for research purposes.

Thank you for your time and administrative cooperation.

Yours faithfully,

Ezenwaka Michael Chinedu
Researcher

APPENDIX C

PROGRAMMING SKILLS ACHIEVEMENT TEST (PSAT)

INSTRUCTION: Answer all questions by ticking (✓) the correct option (A, B, C, or D).

TIME ALLOWED: 45 Minutes

MAXIMUM SCORE: 30 Marks

PART I: SYNTAX RECOGNITION SKILLS

1. Which of the following is a correctly formatted variable declaration in standard programming structure?

A) var 2name = "John";

B) string name = "John";

C) name string = "John";

D) define "John" = name;

2. In structural code syntax, what is the primary role of a closing semicolon (;)?

A) To open a new code block

B) To denote the termination of a single instruction statement

C) To initialize a loop counter variable

D) To skip lines during program execution

3. Which character set is universally utilized to group code statements inside a dynamic block or loop configuration?

A) Square brackets [ ]

B) Parentheses ( )

C) Curly braces { }

D) Angle brackets < >

4. Identify the invalid reserved keyword in computer programming frameworks:

A) while

B) if

C) execute_now

D) return

5. Which symbol is strictly utilized to assign a data value to an initialized variable memory location?

A) ==

B) =

C) !=

D) =>

6. What data type should be declared to store decimal values like 14.35?

A) int

B) char

C) float / double

D) boolean

7. How do you correctly mark a single-line descriptive comment within standard language frameworks?

A) // This is a comment

B) /* This is a comment

C) # This is a comment

D) <- This is a comment

8. A string literal value must always be enclosed within which syntactic operators?

A) Single parenthesis

B) Double quotation marks

C) Percent signs

D) Backslashes

9. Which operator is utilized to perform strict structural comparison checking for absolute equality between two values?

A) =

B) ==

C) ===

D) !=

10. What syntax error occurs when a programmer forgets to close an opened parenthesis in an expression?

A) Runtime error

B) Compilation Syntax error

C) Logical error

D) Semantic error

PART II: ERROR DEBUGGING SKILLS

11. Examine this snippet: int score = 24.68;. What specific compilation error does it generate?

A) Missing semicolon

B) Type mismatch error (assigning float to int)

C) Out of memory error

D) Variable name error

12. If a program runs completely to the end but prints a wrong mathematical average score, what type of error is embedded?

A) Syntax error

B) Runtime exception

C) Logical error

D) Compiler constraint

13. The error message "IndexOutOfBoundsException" implies that:

A) The data memory was cleared mid-execution

B) The code attempted to access an array position that does not exist

C) The variable name used was invalid

D) The compiler file is missing

14. What is the fundamental purpose of utilizing a built-in step-by-step debugger tool?

A) To type code faster

B) To inspect shifting variable values line by line during runtime execution

C) To delete duplicate files automatically

D) To encrypt the final software document

15. Review the following code loop:

```
int count = 1;
while (count <= 5) {
print(count);
}
```

What runtime operational error is present in this structure?

A) The loop will never execute

B) It creates an infinite loop because count is never incremented

C) The condition statement contains an invalid operator

D) The print function is un-initialized

16. What does a "NullPointerException" signify during script execution?

A) The user typed an invalid character string

B) The code is pointing to an object memory address that contains no data value

C) The hardware processor has overheated

D) The loop completed its rounds early

17. If a compiler flags an error stating "Variable 'x' not declared in this scope", how do you debug it?

A) Change the variable name to uppercase

B) Define the data type and identifier for 'x' before calling it

C) Delete the print block completely

D) Restart the computer terminal

18. Which structural block configuration is utilized to capture runtime exceptions safely without crashing a program?

A) if ... else

B) try ... catch

C) while ... loop

D) switch ... case

19. A division-by-zero operational action flags what classification of software error?

A) Syntax check block

B) Runtime exception error

C) Pre-processor definition fault

D) Memory overflow constraint

20. When an intelligent simulator highlights a line of code in red color, what is its primary operational objective?

A) To signify that the system has successfully completed the program

B) To guide the student's attention directly to the exact location of a syntax break

C) To change the color palette of the application template

D) To delete the line from the document cache

PART III: ALGORITHMIC LOGIC DESIGN SKILLS

21. Which core structural logic block executes a block of statements only when a specific boolean constraint evaluates to true?

A) Repetition loop structural design

B) Sequential execution logic

C) Conditional standard choice structure (if-then)

D) Module data packaging layout

22. In standard operational logic flow charting, an elongated diamond-shaped symbol is utilized to represent:

A) Data input initialization processing

B) Terminal program starting and stopping positions

C) Decision-making or branching path points

D) Sequential storage operations

23. What logic configuration must be utilized when you need to repeat a sequence of instructions exactly 80 times?

A) A nested decision block

B) A definite counting iteration loop (for loop)

C) A simple structural variable definition layout

D) A sequence matrix layout

24. Study this flowchart logic path: Start -> Read score -> If score >= 50 print "Pass" else print "Fail" -> Stop. If the input score is exactly 50, what is the structural behavior?

A) It prints nothing

B) It prints "Fail"

C) It prints "Pass"

D) It crashes the system

25. What is the final calculated arithmetic value of the variable result after this sequence runs?

```
int a = 5;
int b = 3;
int result = a * b + 2;
```

A) 25

B) 17

C) 10

D) 30

26. Which metric measures the systematic structural logic flow of an algorithm step-by-step in plain language before coding?

A) Source code format layout

B) Pseudocode notation design

C) Machine code instruction set

D) Bytecode distribution template

27. What logical operational connector yields a true value only when both conditions evaluate to true simultaneously?

A) OR logical operator

B) NOT negation connector

C) AND boolean operator

D) XOR isolated logic path

28. If an algorithm's terminating criteria condition can never evaluate to false, what is its direct operational impact?

A) The script completes its task instantly

B) It runs continuously as an infinite loop, draining hardware resources

C) The editor corrects the statement logic automatically

D) The code converts directly into text structure

29. What is the fundamental logic sequence rule underpinning structural programming paradigms?

A) Executing statements randomly across different text levels

B) Reading and executing statements strictly top-to-bottom, line-by-line

C) Jumping directly to the final statement line upon startup

D) Cycling code blocks backward through memory addresses

30. In logic design, "decomposition" refers to the process of:

A) Deleting old, unused code scripts from a storage drive

B) Breaking down a large computational task into smaller, manageable parts

C) Combining distinct programming tools into one large database

D) Automatically fixing syntax errors using the compiler engine

TABLE OF SPECIFICATION (TOS) FOR THE PSAT

Content Dimensions	Syntax Recognition	Error Debugging	Algorithmic Logic	Total Items	Weight
Topic 1: Basic Variables	Items 1, 4, 5, 6	Items 11, 17	Item 25	7 Items	23.33%
Topic 2: Punctuation Rules	Items 2, 3, 8	Items 13, 16	Items 29, 30	7 Items	23.33%
Topic 3: Conditional Logic	Item 9	Items 18, 19	Items 21, 22, 24	6 Items	20.00%
Topic 4: Loop Structures	Item 10	Item 15	Items 23, 28	4 Items	13.34%
Topic 5: Code Documentation	Item 7	Items 12, 14, 20	Items 26, 27	6 Items	20.00%
Total Questions	**10 Items**	**10 Items**	**10 Items**	**30 Items**	**100%**

LESSON PLAN FOR VIRTUAL CODING SIMULATIONS METHOD

(EXPERIMENTAL GROUP)

WEEK: 1

DURATION: 40 Minutes

TOPIC: Introduction to Variables, Data Types, and Structural Code Syntax

INSTRUCTIONAL STRATEGY: Active Virtual Coding Simulation Sandbox Layout

1. Behavioral Objectives

By the end of this instructional period, students should be able to:

- Correctly type a variable statement using the required syntax structure on the active digital editor screen.
- Trigger execution to compile lines of code and interpret live terminal output responses.
- Identify and resolve missing-semicolon syntax errors based on real-time visual indicator highlights.

2. Instructional Materials & Setup

- Resilient networked computer terminals with updated, lightweight browser software environments.
- An open-source, web-enabled interactive virtual programming sandbox engine configured locally on the intranet network.

3. Instructional Procedure

Step I: Introduction & Interactive Orientation (5 Minutes): The instructor boots up the local network simulation platform and displays the workspace via a visual projector system.

Step II: Active Experimentation & Exploration (15 Minutes): Students type a basic variable initialization statement directly into their responsive editor boxes and hit execution to compile lines of code and interpret live terminal output responses.

Step III: Interactive Error Debugging Scaffolding (10 Minutes): The teacher instructs students to intentionally delete the closing semicolon at the end of their statement line to experience real-time error messages and learn to debug.

Step IV: Metacognitive Consolidation & Evaluation (10 Minutes): Students work independently through progressive challenges within the simulation layout.

LESSON PLAN FOR CONVENTIONAL METHOD (CONTROL GROUP)

WEEK: 1

DURATION: 40 Minutes

TOPIC: Introduction to Variables, Data Types, and Structural Code Syntax

INSTRUCTIONAL STRATEGY: Conventional Syntax-on-Board Theoretical Lecturing

1. Behavioral Objectives

By the end of this instructional period, students should be able to:

- Define the structural rule requirements for initializing variables in notebook logs.
- State the role of a closing semicolon in code design.
- Memorize core keyword data type names from the textbook.

2. Instructional Materials

- Standard dry-erase whiteboard setup and colored markers.
- Approved secondary Computer Studies textbooks and student paper notebooks.

3. Instructional Procedure

Step I: Teacher-Centered Lecture Introduction (10 Minutes): The teacher instructs students to open textbooks to Chapter 4, Page 42 and lectures from the front whiteboard.

Step II: Whiteboard Transcription (15 Minutes): The instructor manually draws code examples using markers and demands that students copy text strings into paper notebooks.

Step III: Theoretical Error Demonstration (5 Minutes): The teacher writes an incorrect line on the board and circles errors manually with markers to verbally explain code execution constraints.

Step IV: Rote Evaluation & Copying (10 Minutes): Students work silently in their seats copying review questions and definitions.

VALIDATOR'S COMMENT

Effect of Virtual Coding Simulations on the Programming Skills of Senior Secondary School Students in Nsukka Local Government Area, Enugu State, Nigeria.

I have carefully scrutinized the researcher's instrument and have the following comments and further suggestions to make regarding the instrument:

……

……

……

……

……..

……

……

……

……

……..

If the mentioned corrections are affected, I hereby recommend that the instrument be used for the study.

Name of Validator: ……

Signature: ……

Institution Address: ……

RELIABILITY OF THE INSTRUMENT

To measure the consistency and reliability of the Programming Skills Achievement Test (PSAT), the tool was piloted using the technique of field testing. This test was administered only once on thirty (30) SS2 Computer Studies Students of a secondary school situated in the Obollo-Afor Education Zone. The characteristics of this sample are identical to that of the target population. However, this sample was excluded from the actual research sample to avoid test-wiseness.

Since the PSAT is a tool containing objective multiple-choice questions that have been dichotomously marked (one mark for a right answer and zero for a wrong answer), the reliability of the PSAT was computed using Kuder-Richardson Formula 20 (KR-20). Reliability coefficient (r) was computed through the following matrix:

$$r = \frac{K}{K-1}\left(1 - \frac{\sum p\,q}{\sigma^2}\right)$$

Where:

- K = Total number of items in the test ($K = 30$).
- p = Proportion of students who answered each individual item correctly.
- q = Proportion of students who answered each individual item incorrectly ($q = 1 - p$).
- $\sum p\,q$ = Sum of the variance products for all 30 test items.
- σ^2 = Total variance of the overall test scores from the pilot group.

The computation of statistics provided an index for reliability coefficient of 0.84. As per the psychometric standards, any instrument having a coefficient value greater than 0.70 is considered very reliable for research in education. Hence, an index value of 0.84 for PSAT proves that there is high internal consistency and stability of the test items.