STRATEGIES FOR IMPROVING THE TEACHING AND LEARNING OF QBASIC PROGRAMMING LANGUAGE IN COLLEGES OF EDUCATION IN ENUGU, EBONYI AND ANAMBRA STATES

VOCATIONAL TEACHER EDUCATION, UNIVERSITY OF NIGERIA NSUKKA IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF MASTER DEGREE IN COMPUTER EDUCATION

BY
OLELEWE, CHIJOIKE JONATHAN
PG/M.ED/05/40314

FEBRUARY, 2009

APPROVAL PAGE

This thesis has been approved for the Department of Vocational Teacher Education, University of Nigeria Nsukka.
………………………………… ……………………………………

PROJECT SUPERVISOR           RESEARCHER

Dr. E.E. AGOMUO   OLELEWE, CHIJOKE JONATHAN

………………………………...               …………………………

HEAD OF DEPARTMENT                        EXTERNAL EXAMINER

DR E.E. AGOMUO

…………………………………………………

DEAN FACULTY OF EDUCATION
OLELEWE, CHIJOIKE JONATHAN a postgraduate student in the department of Vocational Teacher Education, with registration number PG/M.ED/05/40314, has satisfactorily completed the requirement for course and research work for the degree of masters of education in Computer Education. The work embodied in this thesis is original and has not been submitted in parts or in full for any diploma or degree of this or any other University.
PROJECT SUPERVISOR          RESEARCHER

Dr. E.E. AGOMUO          OLELEWE, CHIJOKE JONATHAN
DEDICATION

This research is dedicated to God Almighty who is the author and finisher of my faith.
ACKNOWLEDGEMENTS
I acknowledge with immense appreciation the efforts of my supervisor Dr. E. E. Agomuo whose guidance, assistance and encouragement helped me to successfully complete this work. My sincere and inexplicable gratitude goes to my reader Prof. (Mrs.) C.A Obi for her contributions.

My special thanks to Prof. E.C. Osuala, Prof. S.O. Olaitan, Prof. (Sir.) S.C.O.A Ezeji, Dr. (Mrs.) T.C.Ogbiuanya, Dr. E. Osinem and Dr. Etonyeaku for their encouragement. And to all the Lecturers and Staff of the department of Vocational Teacher Education, University of Nigeria Nsukka, I say a big thank you for your ceaseless assistance in helping me actualize my dream.

I am also indebted to my friends and colleagues Amaka Ezeobi, Uncle Sam, Uncle Abbah Romanus, Prof. Jide, Ifeanyi Chielo, Getruode, Amaka Ekweghi, Bode Daniel, Kayode, Erinma, Nzeadibe and family, Maureen Nwanze (Mrs.) and Nnenna Ibezim (Mrs.) for their encouragement and support in diverse ways towards the completion of this works. On a special note, I thank my younger sister Juliana Lechi Olelewe for realizing how much I need a Laptop in other to accomplish this task.

Finally, my special thanks to my Godfather, Mr. Adibe N. Onyeokoro for his encouragement even though he was not happy with me for not completing this work on record time for no fault of mine.

Olelewe, Chijioke Jonathan
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title page</td>
<td>i</td>
</tr>
<tr>
<td>Approval Page</td>
<td>ii</td>
</tr>
<tr>
<td>Certification</td>
<td>iii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iv</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>v</td>
</tr>
<tr>
<td>Table of Contents</td>
<td>vi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>vii</td>
</tr>
<tr>
<td>Abstract</td>
<td>viii</td>
</tr>
</tbody>
</table>

CHAPTER I: INTRODUCTION.................................1

  Background of the Study.................................1

  Statement of the Problem...............................6

  Purpose of the Study....................................7

  Significance of the Study..............................8

  Research Questions.....................................9

  Hypotheses..............................................10
Delimitation ..................................................................................10

CHAPTER II: REVIEW OF RELATED LITERATURE............................12

   Conceptual and theoretical framework ........................................12

   The Need for Computer programming........................................22

   Methods of Teaching Computer programming..........................25

   Tools for Reliable Software Design in QB programming...............39

   Related Empirical Studies.........................................................45

   Summary of Literature Reviewed.............................................50

CHAPTER III: METHODOLOGY................................................52

   Design of the Study.................................................................52

   Area of the Study.................................................................52

   Population.............................................................................52

   Instrument for Data Collection...............................................53

   Validation of the Instrument...................................................54

   Reliability of the Instrument...................................................54

   Method of Data Collection.....................................................54

   Method of Data Analysis.......................................................54
CHAPTER IV: PRESENTATION AND ANALYSIS OF DATA..........................56

Research Question 1.................................................................56

Research Question 2.................................................................57

Research Question 3.................................................................59

Research Question 4.................................................................60

Research Question 5.................................................................62

Hypothesis 1.................................................................64

Hypothesis 2.................................................................66

Hypothesis 3.................................................................68

Findings of the Study.................................................................70

Discussions of the Findings.................................................................74

CHAPTER V: SUMMARY, CONCLUSIONS AND RECOMMENDATIONS......81

Re-statement of the Problem.................................................................81

Summary of the Procedures used.................................................................82
Summary of findings of the Study.........................................................83

Implications of the Study.................................................................84

Conclusions....................................................................................85

Recommendations..........................................................................86

Suggestions for further Study.........................................................87

References.....................................................................................88

Appendices

Appendix I: Performance of Students ........................................93

Appendix II: Letter to the Respondents .................................94

Appendix III: Letter for validation of Instrument .......................95

Appendix IV: Questionnaire.........................................................96

Appendix V: Reliability of Data ....................................................102
LIST OF TABLES

Tables                                      Pages

1. Mean and Standard deviation ratings of lecturers’ and Instructors’ Responses on technical skills strategies required for the teaching of QB Programming language in colleges of education…………………………...…………………………………………….56

2. Mean and standard deviation ratings of lecturers’ and students’ responses on strategies for improving the teaching methodology of QBasic programming in colleges of education………………………………………………………………………58

3. Mean and Standard deviation of teachers’ and students’ responses on the strategies for improving the utilization of instructional facilities in colleges of education……………………………………………………………………………59
4. Mean and Standard deviation of teachers’ and students’ responses on the strategies for improving the learning environment for the teaching and learning of QB in colleges of education……………………………………………………61

5. Mean and standard deviation of lecturers’ and students’ responses on the strategies for improving administration needs required for teaching and learning of QB programming in colleges of education…………………………………………………………….63

6. The t-test analysis of the mean responses of lecturers’ with B.Sc. degree in computer and those with HND on the technical skills required for teaching of QB programming in colleges of education……………………………………………………………………………………………………………………65

7. The t-test analysis of the mean responses of lecturers’ and Instructors’ with regard to the teaching methodology required for teaching and learning of QB programming in colleges of education………………………………………………………………………………………………67

8. The t-test analysis of the mean responses of lecturers’ and Instructors’ with regard to the instructional facilities utilization strategies required for teaching and learning of QB programming in colleges of education……………………………………………………………………………………………………………………69

Abstract

The study is focused on the strategies for improving the teaching and learning of QB programming language in colleges of education. This survey study was conducted in the south east geopolitical zone of Nigeria namely Ebonyi, Enugu and Anambra States aimed at identifying the strategies for enhancing teaching and learning of QB programming language. Data were collected with the use
of structured questionnaire containing 58 items. Descriptive statistics, mean, standard deviation and t-test statistical tool were used for analyzing data. No sample was taken. The population consisted of 295 respondents (35 Instructors, 25 lecturers, and 235 computer students). The instrument was subjected to face validation by three experts’, it was further pilot tested on 20 respondents who were used to establish the internal consistency of the instrument. The reliability coefficient was calculated to be 0.98, 0.99, 0.98, 0.98 and 0.99 respectively for each section of the instrument and 0.99 as the overall coefficient. Based on data analyzed, it was found out that technical skill strategies can be used to improve the teaching of QBasic programming. Also, teaching methodology strategies, instructional facilities utilization strategies among others can be employed by computer educators to improve the teaching and learning of QBasic programming in colleges of education. Consequently, it was recommended among others that school administrators should ensure that computer educators are sponsored on retraining programmes at least once every year by organizing workshops, seminars and conferences to enable them learn the modern technological skills in their chosen career. In order to make the art of programming responsive to the needs of the society, computer students should be made to spend at least six months in computer and ICT programming firms during their industrial work experience to enable them acquire the relevant skills required to excel in the world of work while School administrators should ensure that adequate instructional facilities are provided to every computer laboratory to enable students learn and master sound programming skills.

CHAPTER 1

INTRODUCTION

Background of the Study
Beginners All Purpose Symbolic Instruction Code (BASIC) occupies a central position in the teaching and learning of computer programming in higher institutions where computer education is offered as a course. This is because QuickBasic (QB) is the first introductory programming course of study offered by first year computer students in higher institutions, including colleges of education in Nigeria. QuickBasic is therefore highly portable and suitable for teaching BASIC programming to beginners. This is because its characteristic features make it a vital tool for teaching and learning of computer programming. The teaching and learning of QB language lies in its interactive user interface, which provides immediate feedback on syntax and run-time errors. Like other structured programming languages such as PASCAL, FORTRAN, PL/I, ADA, ALGOL etc, it is a procedure oriented language, which employs the technique of problem solving and uses English syntax in the solution of problems in a form understandable to human beings.

The teaching and learning of QB lies in its defining characteristics as a procedure oriented language, which among others the expression of a BASIC in an algorithmic form which is a series of finite sequence of precise, step-by-step instructions for performing some task (Bakpo, 2005). Then the use of flowchart which is a graphical or pictorial representation of flow of a computer program steps using connecting lines with arrows to indicate the flow of a computer program steps and finally the coding of the program steps which is sequentially executed.

It becomes imperative to teach and learn QB considering the goals of tertiary education as noted in the National Policy on Educational development of high level manpower training, development and inculcation of right type of values for the survival of the Nigerian society and
individuals, development of the individual’s intellectual capabilities for the acquisition of physical and intellectual skills necessary for individuals to be self-reliant and be useful members of the society (FGN, 2004).

These noble goals are to be pursued through teaching and learning, research and development, generation and dissemination of knowledge and so on (FGN, 2004). It is in line with these goals that the Nigerian Colleges of Education (NCE) offer a number of programming courses and as such began to harness the potentials of teaching and learning of computer programming for technological advancement. Thus the Nigerian Colleges of Education offers a number of programming courses such as BASIC (QUICKBASIC), FORTRAN, COBOL, PASCAL, and VISUALBASIC (VB) programming languages in other to equip the students with appropriate programming skills needed to become self-reliant as well as excel in his/her work related task. The National Commission for Colleges of Education minimum standard however maintained that BASIC Programming (CSC 112) among others should be taught to students of Nigerian colleges of education (NCCE, 2005). The degree of sophistication in numeracy through the use of computer has become very pronounced all over the world and Nigeria cannot afford to ignore the role which computer literacy plays in achieving the national goal of technological development. Hence her resolve to introduce the teaching and learning of object oriented programming languages in addition to the structured programming courses in the Computer Education curriculum of National Colleges of Education. To achieve these objectives and for effective acquisition of programming skills and knowledge as a school subject, it should be effectively taught by qualified computer educators. Effective teaching of QBASIC requires application of different strategies or techniques to make it interesting and meaningful. Akpan
(1991) opined that teaching (strategies) technique is the practice and refinement of presentation which a teacher uses to make his teaching more effective, interesting when using specific method or teaching aid. Obi (2005) observed that teaching techniques are strategies employed by the teacher to enhance his teaching.

According to Ogwo (2005), the teacher is essentially a facilitator of learning. As much as possible, the teacher facilitates learning by permitting the learner’s own interests, attitudes, aptitudes and experiences to influence the kind of learning that will take place. He maintained that the teacher is a parent to the students, a model, a dispenser, a cultivator and a technician. The computer teacher must be knowledgeable in programming, maintenance and troubleshooting, networking, installation, web design e.t.c. and above all, posses teaching qualification. A computer educator in Colleges of Education is therefore expected to have a minimum of B.Sc. (Ed) in computer education or a B.Sc. in computer science with a postgraduate diploma in education (NCCE, 2001).

Teaching according to Fafunwa (1982) is a conscious and deliberate effort made by a matured or experienced person to impart knowledge to immature or less experienced person with the intension that the latter will learn or come to believe that he is taught on good grounds. According to Obi (1996), learning is a process by which a learner acquires knowledge, skills, attitude, concepts, tradition, values and the ability to comprehend, analyze and evaluate through an active involvement and participation in the process which ultimately leads to mental, social, spiritual and physical growth and development. Obi (2005) identified four essential activities that constitute teaching and learning namely class organization, class control, lesson plan and evaluation. Class organization is concerned with the establishment of some routines understood
by all the class members in such matters as submission and collection of class assignments, sweeping of the classrooms and maintaining the chalkboard. Class controls enable the teacher to keep the students busy from the beginning to the end of the lesson. Course or lesson planning activity enables the teacher to prepare the scheme of work based on the school syllabus or curriculum. This scheme of work so planned as a whole is broken down into workable units to facilitate teaching and learning (Obi, 2005). Lesson plans are brief notes of how the teacher intends to go about his classroom instruction and lastly evaluation though a continuous process is conceived as the determination of the congruence between performance and the objectives (Mehrens and Lehmann, 1978). It provides the basis for re-teaching in the effort to develop mastery in learners.

Unfortunately, the goals of teaching and learning of computer programming have not been fully actualized considering the lack of interest accompanied with poor performance of students in the learning process of QB. The teaching and learning of QB is therefore one area of computer education curriculum in which students can be prepared to acquire appropriate skills, abilities, and competences as equipment for the individual to live in and contribute to the development of the society. The role of QB programming language in the developmental process is vital and is directly related to the effectiveness of the economy in providing the requirement of trained manpower for development process. The objectives of computer education for her graduates of Nigerian Colleges of Education are therefore to contribute to national development by producing teachers who will teach computer studies at the primary and junior secondary school levels, write programs and process data with maximum speed and accuracy, and
demonstrate reasonably high level of competence for further studies in computer science (NCCE, 2004: 37).

However, all these important national objectives are factors of major variables. For instance, we cannot boast of adequate skilled manpower to man the country’s economy, if the programme implementation for manpower development is faulty. Therefore, evaluation of students’ performance is essentially an evaluation of the programme itself (Okoro, 1993). The outcome of any educational curriculum is used to determine the index of success or failure of that system of education.

Besides, there are certain competencies a teacher of computer education should possess and be able to demonstrate in classroom. One of these competencies is the programming skill, which has to do with possession of problem solving abilities needed to analyze BASIC programming problems and also the pedagogical competency, which has to do with the skills in methods of imparting the various technical skills to the students. The quality of a teacher according to Ukeje (1997) in Gana (2003) is a strong predictor of student’s quality. It is widely acknowledged that “no education system can rise above the quality of its teachers” (NPE, 1981). From the foregoing, it is then clear that if certain strategies are adopted to improve the teaching and learning of QB programming in Colleges of Education it can help to motivate students’ interest thus yield better performance and competency since teaching and learning of QBASIC programming is a major step towards skill acquisition and manpower development.

**Statement of the Problem**
The cost of programming has risen because of shortages of skilled personnel as required for programming which have pushed up labour costs, whereas; the costs of hardware have fallen because of technical innovation and increased automation (French, 1996). He maintained that it will be ideally wrong to reduce programming standards in order to achieve greater programmer output because of the potentially damaging effects of errors, high correction and maintenance and difficulties experienced in transferring substandard programs from one computer to another.

Though there is need to reduce programming cost, and at the same time strive for increasing the quality of programming in colleges of education. Computer programming at the National Certificate on Education (NCE) level contributes to nation building through training and production of quality computer programmers that should man the industrial and technological sectors of our economy through self or paid employments. Unfortunately this singular objective is still far from being actualized most importantly when one looks at the general quality of NCE computer graduates. The performance of the students of computer education in QuickBasic programming course in Colleges of Education in Anambra, Enugu and Ebonyi States has consistently been very discouraging. Available records from fives Colleges of Education in Enugu, Anambra, and Ebonyi States for the year 2003 – 2007 as shown in appendix 1 showed an overall entries figure of 1331 candidates with only 607 complete passes representing 46%. This situation is rather pathetic and as such has become a matter of concern to a lot of Computer Educationists, employers of labour and parents in Anambra, Ebonyi and Enugu States.

The problem of mass failure in QB examinations negates ensuring adequate manpower training in the areas of computer education. It is therefore pertinent to find out the problems
militating against effective teaching and learning of QuickBasic programming language as well as the strategies that can be employed in order to improve the teaching and learning of QuickBasic programming language in Colleges of Education in Anambra, Enugu and Ebonyi States.

**Purpose of the Study**

The major purpose of the study is to determine the strategies for teaching and learning of QuickBasic (QB) Programming language in Colleges of Education in Anambra, Ebonyi and Enugu States. The study specifically sought to:

1. determine the strategies for improving the technical skills of the teachers for the teaching of QB in Colleges of Education in Anambra, Ebonyi and Enugu States.
2. determine the strategies for improving the teaching methodology required for the teaching and learning of QB in Colleges of Education in Anambra Ebonyi and Enugu States.
3. identify the strategies for improving the utilization of instructional facilities for the teaching and learning of QB programming in Colleges of Education in Enugu, Ebonyi and Anambra States.
4. identify the strategies for improving learning environment for the teaching and learning of QB programming in colleges of Education in Anambra, Ebonyi and Enugu States.
5. Identify strategies for improving administrative needs for the teaching and learning of QB in Colleges of Education in Anambra, Ebonyi and Enugu States.

**Significance of the Study**
This study was motivated as a result of growing evidence in poor performance of NCE Computer students in learning of QB programming and what effort could be made to encourage students’ interest in learning of computer programming. It is therefore hoped that, the findings of the study will in no little way provide adequate information on the strategies of enhancing the teaching and learning of QB programming. Hence the findings may be of benefit to the following ministries (Education, Labour and productivity), school administrators, information and communication industries, curriculum planners, students, lecturers, parents and the society at large.

The ministries at both the federal and states level is instrumental to policy formulations, the strategies that may be generated from this study might help the policy makers in making necessary review and further contribution in the policy formulation process that may encourage the employers in hiring NCE computer products. Besides, it may also help to erase the erroneous impression the society have about poor quality programme of computer education. The result of these findings will invariably stimulate the students’ interest on the bounties of opportunities associated in possession of skills of QB programming in being job creators rather than job seekers. Consequently, this will help to retain and improve the job opportunities, retraining and career progression for the students.

The findings that may be generated from this study will help to improve quality of educational institutions, facilities for training, teacher capability, curriculum issues and methods of knowledge delivery thereby enabling curriculum planners to conform to the societal and institutional needs. The findings of this study may also serve as a reference for redefining the remuneration pattern of the computer Lecturers, System Analysts and Programmers in general,
the cost of education, and prompt payment of staff salaries. All these may help the educational administrators (Provosts, H.O.D’s, Deans, Registrar) etc on a better management as it will motivate both staff and students.

The findings of this study may also help the administrators on suggestion of strategy on change of educational structure of computer programme so that the rightful candidates who are really interested in learning computer programming are truly admitted.

**Research Questions**

1. What are the strategies that can be employed to improve the technical skills of teachers for the teaching of QB in Colleges of Education in Anambra, Enugu and Ebonyi States?
2. What are the strategies that can be employed for improving teaching methodology of QB programming in Colleges of Education in Anambra, Enugu and Ebonyi States?
3. What are the strategies for improving the utilization of instructional facilities for the teaching and learning of QB programming in Colleges of Education in Anambra, Enugu and Ebonyi States?
4. What are the strategies for improving the learning environment for teaching and learning of QB programming in Colleges of Education in Anambra, Enugu and Ebonyi States?
5. What are the strategies for improving administration needs required for teaching and learning of QB programming in Colleges of Education in Anambra, Enugu and Ebonyi States?

**Hypotheses**
The following null hypotheses were formulated to guide the study and will be tested at 0.05 level of significance.

H01: There is no significant difference (p < 0.05) in the responses of lecturers’ possessing a B.Sc. degree in Computer Science/Education and those possessing HND degrees with regard to the technical skills required for teaching and learning of QB in Colleges of Education in Enugu, Ebonyi and Anambra States.

H02: There is no significant difference (p < 0.05) between the mean responses of Lecturers’ with B.Sc. degrees and those with HND degrees with regard to the teaching methodology required for teaching and learning of QB in colleges of education in Enugu, Ebonyi and Anambra States.

H03: There is no significant difference (p < 0.05) between the responses of lecturers’ with B.Sc. degree and those with HND degree with regard to the use of instructional facilities required for teaching and learning of QB Colleges of Education in Enugu, Ebonyi and Anambra States.

**Delimitation of the Study**

This study is delimited to the strategies for improving the teaching and learning of QB programming language in colleges of education in Enugu, Ebonyi and Anambra states.

The study is also limited to National Colleges of Education (NCE) computer science students and lecturers in colleges of education in Enugu, Ebonyi and Anambra States where QBASIC programming is offered as a course of study in computer science education department namely Federal College of Education Eha-Amufu, Enugu State College of Education Technical,
Ebonyi State college of Education Ikwo, College of Education Umunze and Nwafor Orizu College of Education Nsugbe in Anambra State.
CHAPTER II

REVIEW OF RELATED LITERATURE

The review of literature for this study was organized under the following sub-headings:

1. Conceptual and theoretical framework
2. The need for computer programming
3. Methods of teaching computer programming
4. Tools for reliable software design in QB programming
5. Review of related empirical studies
6. Summary of literature reviewed.

**Conceptual and theoretical framework**

QuickBasic programming is taught in two parts namely: theory and practical. In the teaching and learning of QB programming knowledge in the theory may not help the students to overcome problems encountered, therefore, students require the practical aspects to supplement the theoretical knowledge in order to have good mastery of the subject.

Theoretical knowledge of students alone in QB programming could not make them become skilled programmers, until they learn the practical drills associated with programming such as loading of compiler, entering or typing the program codes, running the codes, debugging the codes in other to yield valid results. Therefore, the computer laboratory provides much needed facilities for skill training in programming to compliment whatever background knowledge acquired in the classroom in order to develop the much needed entry level skill for programming (Bakpo, 2005). He observed that programming is the process of planning a sequence of instructions for a computer to follow in order to accomplish a specific task.
Programming according to Bakpo consists of two phases namely: problem solving and implementation phase. The later phase is therefore carried out in the computer laboratory for actual learning to be successful.

The computer laboratory is the technical office specifically designed for the purpose of impacting practical knowledge and skills to students in the school through drills and practice. A computer laboratory is specially earmarked for computer practical learning activities by the school administration. This area usually possesses the potentials required for development of programming skills and general computer proficiencies. It may be located in the computer department or at a fairly walking distance within the school. Ogbonna and Okoli (2007) opined that the laboratory has great importance in many dimensions namely: as a source of income to the school, as a store of value, as a source of transfer of knowledge, for developing skills in the students, for stimulating students interests in learning, for laboratory research, for recreation purposes and for improving background knowledge of the students.

Practical demonstration involves acquisition of skills in areas of programming, engineering, installation and maintenance, troubleshooting, word processing etc. In order to acquire these skills, computer laboratories are established since it is where most practical work in computer science is done.

Any college, individual or group of persons that wishes to embark upon computer training must first of all acquire or establish an adequate computer laboratory that certifies the National Commission for Colleges of Education (NCCE) standard minimum guide (FME, 2004). In other words, the computer laboratory plays an important role in the practical teaching and learning of computer programming in Colleges of Education and higher institutions in general.
The computer laboratory is therefore a demonstration laboratory where abstract concepts are made concrete in nature through proper teaching.

According to Obi (2005) a method or technique chosen by the teacher must be one in which he has competence so that his teaching would be effective. If the teacher is not conversant with a method or technique he uses in class, he may end up doing more harm to the learners than if he did not teach at all. She opined that if the method is well known to him then the teacher can use techniques that can be easily combined with different methods used in teaching different units of a course so as to achieve the objectives set by the teacher. Evaluation enables the teacher to determine if the goal has been attained or not. Evaluation is an integral part of teaching and learning. Saylor, Alexander and Lewis (1981) opined that evaluation is concomitant with learning in the mastery formula. Obi (2005) opined that evaluation provides the basis for re-teaching in the effort to develop mastery in teaching. According to Mehrens and Lehman (1978) evaluation is conceived as the determination of the congruence between performance and objectives. Evaluation is therefore vital in teaching and learning because it not only motivates the learners and teachers, but provides basis for locating errors resulting from learning. Obi (2005) maintained that evaluation provides students opportunities to express themselves, and helps the teacher to appraise what the students has learnt thus influencing the methods of teaching and study employed by the teacher and the learners.

Too difficult tasks may result to frustration and possible destruction of interest in computer learning pursuits. French (1996) emphasized that computer programming cannot be successful without genuine interest and positive attitude towards its practice. Interest is a force
which derives the students throughout the drudgery in order that they may meet the end that is set before them, that is goals and objectives to which the interest draws the students.

Learning theories are approaches that reveal strategies for effective teaching and learning of various vocational skills and concepts. Vocational skills are the abilities, knowledge and aptitudes which enable individuals to perform a given task according to specification. Computer education which is an aspect of vocational education is a special type of education that equips an individual with practical skills for self-reliance and to be productive in the society. Olaitan, Nwachukwu, Igbo, Onyemachi and Ekeong (1999) opined that if a man lacks the habit of thinking and that of doing his work, that man is of no use as a productive economic or social factor on that job.

**Learning Theories**

The choice of theory of learning greatly influences the instruction by a computer teacher. The learning theories that can be applied by a computer lecturer in the teaching and learning of computer programming are:

**Connectionism**

The learning theory of connectionism propounded by Thorndike represents the original S-R framework of behavioural psychology. Learning is the result of associations of forming between stimuli and responses. Such associations or habits become strengthened or weakened by the nature and frequency of parings. The paradigm for S-R theory was trial and error learning in which certain responses come to dominate others due to rewards. The hallmark of connectionism (like all behavioural theory) was that learning could be adequately explained without referring to any unobservable internal states.
Thorndike’s theory consists of three primary laws; (1) law of effect – responses to a situation which are followed by a rewarding state of affairs will be strengthened and become habitual responses to that situation, (2) law of readiness - a series of responses can be chained together to satisfy some goal which will result in annoyance if blocked, and (3) law of exercise – connections become strengthened with practice and weakened when practice is discontinued. A corollary of the law of effect was that responses that reduce the likelihood of achieving a reward state (i.e., punishments, failures) will decrease in strength.

The theory suggests that transfer of learning depends upon the presence of identical elements in the original and new learning situations; i.e., transfer is always specific, never general. In later versions of the theory, the concept of “belongingness” was introduced; connections are more readily established if the person perceives that stimuli or responses go together (Gestalt principles). Another concept introduced was “polarity” which specifies that connections occur more easily in the direction in which they were originally formed than the opposite. Thorndike also introduced the “spread of effect” idea, i.e., rewards affect not only the connection that produced them but temporally adjacent connections as well.

**The Gestalt Principle**

The word “Gestalt” means the way a thing has been “placed” or “put together”. It is translated in psychology as “pattern” or “configuration”. The principle assumes that analysis of parts, however thorough, cannot provide an understanding of the whole. Thus, the theory states that to comprehend the full nature from the whole, it is necessary to analyze the whole structure from the beginning to the end-the characteristics of its constituent parts. The whole may have
attributes that require a certain place, role and functions for each part in the whole, these attributes are not deducible from analysis of the parts in isolation.

In a gestalt according to Obi (2005), the nature of the parts is required by the characteristics of the whole, and the parts are fused and interdependent, interacting in a specific structural manner. Hence the theory assumes that learning is a process of organizing physical, psychological and other events or elements into an integrated unit or a whole. The basic process of learning is conceived not as a series of automatic arbitrary hookups but as discovering the structural, organized characteristics of the environment. This is where the Gestaltists disagree with the associationists. Thus, learning is based on insight and understanding as the basis for drill and repetition (Obi, 2005).

**Cognitive Flexibility Theory**

Cognitive flexibility theory focuses on the nature of learning in complex and ill-structured domains. Spiro and Jenhg (1990:165) stated that cognitive flexibility means the ability to spontaneously restructure one’s knowledge, in many ways, in adaptive response to radically changing situational demands. This is a function of both the way knowledge is represented (e.g. processes of schema assembly rather than intact schema retrieval). The theory is largely concerned with transfer of knowledge and skills beyond their initial situation. For this reason, emphasis is placed upon the presentation of information from multiple perspectives and use of many case studies that present diverse examples. The theory also asserts that effective learning is context – dependent, so instruction needs to be very specific. In addition, the theory stresses the importance of constructed knowledge; learners must be given an opportunity to develop their own representations of information in order to properly learn. Thus, computer learners are
greatly affected where they are not provided with adequate instructional materials to write and type their program codes, test-run, debug and view the computer result.

Cognitive flexibility theory though builds upon other constructivist theories (e.g., Bruner, Ausubel, and Piaget). It is especially formulated to support the use of interactive instructional technology (e.g. videodisc, hypertext). This theory will facilitate the teaching and learning of QBASIC programming hence it ensures that adequate interactive instructional technology to give the learners opportunities to develop their own programming skills and abilities.

This theory stipulates that there are several different types or levels of learning. The significance of these classifications is that each different type requires different types of instruction. Gagne identifies five major categories of learning: verbal information, intellectual skills, cognitive strategies, motor skills and attitudes. Different internal and external conditions are necessary for each type of learning. For example, for cognitive strategies to be learned, there must be a chance to practice developing of new solutions to problems; to learn attitudes, the learners must be exposed to a credible role model or persuasive arguments.

Gagne suggests that learning tasks for intellectual skills can be organized in a hierarchy according to complexity: stimulus recognition, response generation, procedure following, use of terminology, and discriminations, concepts formation, rule application, and problem solving. The primary significance of the hierarchy is to identify prerequisites that should be completed to facilitate learning at each level. Prerequisite are identified by doing a task analysis of a learning/training task. Learning hierarchies provides a basis for the sequencing of instruction.

In addition, the theory outlines nine instructional events and corresponding cognitive processes:
1. gaining attention (reception)
2. informing learners of the objective (expectancy)
3. stimulating recall of prior learning (retrieval)
4. presenting the stimulus (selective perception)
5. providing learning guidance (semantic encoding)
6. eliciting performance (responding)
7. providing feedback (reinforcement)
8. assessing performance (retrieval)
9. enhancing retention and transfer (generalization).

These events should satisfy or provide the necessary conditions for leaning and serve the basis for designing instruction and selection of appropriate media (Gagne, Briggs and Wager, 1992).

**Constructivist theory**

This theory was developed by Bruner in 1966. A major theme in the theoretical framework of Bruner is that learning is an active process in which learners construct new ideas or concepts based upon their current and past knowledge. The learner selects and transforms information, constructs hypotheses, and makes decisions, relying on a cognitive structure to do so. Cognitive structure (i.e., schema, mental models) provides meaning and organizations to experiences and allows the individual to “go beyond the information given”.

As far as instruction is concerned, the computer instructor should try and encourage students to discover principles by themselves. The instructor and students should engage in an active dialog (i.e. Socratic learning). The task of the instructor is to translate information to be learned into a format appropriate to the learner’s current state of understanding. The proponent of this theory maintained that curriculum should be organized in a spiral manner so that the student continually builds upon what they have already learned.
Bruner (1966) states that a theory of instruction should address four major aspects: (1) predisposition towards learning, (2) the ways in which a body of knowledge can be structured so that it can be most readily grasped by the learner, (3) the most effective sequences in which to present material, and (4) the nature and pacing of rewards and punishments. Good methods for structuring knowledge should result in simplifying, generating new propositions, and increasing the manipulation of (information).

In his more recent work, Bruner (1986, 1990, 1996) has expanded his theoretical framework to encompass the social and cultural aspects of learning as well as the practice of law. Bruner’s constructivist theory is a general framework for instruction based upon the study of cognition. Much of the theory is linked to child development research. The ideas outlined in Bruner (1960) originated from a conference focused on science and mathematics learning. Bruner illustrated his theory in the context of mathematics and social science programs for young children. For example the concept of prime numbers appears to be more readily grasped when the child, through construction, discovers that certain handfuls of beans cannot be laid out in completed rows in a single file or in an incomplete row-column design in which there is always one extra or one too few to fill the pattern. These patterns, the child learns, happens to be called prime. It is easy for the child to go from this step to the recognition that a multiple table, so called, is record sheet of quantities in completed multiple rows and columns. Here is factoring, multiplication and primes in a construction that can be visualized (Bruner, 1973).

**General Principle of Teaching**

The general rules that need to be adhered to in order for teaching to be effective are:
**General to Specific**

Under this rule the teaching should start from the general knowledge to the specific one that the learner is trying to study. For example, the importance of problem solving in programming will not be understood unless the teacher first teaches and demonstrate the uses of algorithm and flowcharting as a technique for problem solving to the students. The students who learnt the functions of the techniques of problem solving in computer programming will find it easier to apply algorithm as the first step toward problem solving involving programming.

**Known to Unknown**

For meaningful teaching to be achieved, the teacher should start his teaching with reference to a related knowledge to the subject matter, which he is sure that the learners have. For instance, students of computer education must be taught the basic principles of good programming before teaching them data structures used in writing programming.

**Simple to Complex**

Simple materials are, of course, easier to understand than complex ones. Such materials serve as ego buster to get the learner interested and prepared for the more complex task ahead. In teaching computer programming courses, the teacher teaches BASIC programming first before proceeding to teach FORTRAN, PASCAL and VISUALBASIC programming language. Obi (2005) opined that the subject to be learnt becomes more difficult as we move up the ladder.

**Concrete to Abstract**

A learner remembers what he saw much quicker than what he was told. A computer teacher who intends to teach his students the computer keyboard and how it is used in keying in data into the system may spend the whole year talking about the same subject matter without
achieving his objectives, if he does not bring a sample of computer keyboard to the class or a diagram of computer keyboard.

Out of these theories discussed, cognitive flexibility theory by Gagne (1962) matches the instructional approach for teaching and learning of computer education courses especially for developing computer programming and other technical skills in the learners. It is observed that for a proper transfer of programming skills to the learners, the learners must be given the opportunity to develop their own representations of information in order to properly learn and to do so, they must be provided with interactive instructional technology to facilitate teaching and learning. The theory of Constructivist propounded by Bruner (1966) will also be utilized in teaching and learning of computer programming so as to enable the students continually builds upon what they have already learnt.

However, other theories, according to Thorndike (1913) and Gestalt principle of learning may be utilized when they seem appropriate because emphasis is placed on the individuals (learners and teachers) rather than on any particular theory. This is to ensure mastery of QuickBasic programming skills by the students in old Anambra State of Nigeria.

THE NEED FOR COMPUTER PROGRAMMING

In recent times, there has been a growing concern among well meaning Nigerians over lack of computer programmers, System Analysts and Computer Scientists in general. One of the reasons advanced for this is the poor emphasis on sound practical experience in schools. Consequently, there have been calls for a functional education- a practical and vocational oriented curriculum and approach in the total educational process in schools.
Okoli (2005) maintained that teaching of computer programming in Nigerian Colleges of Education and other higher institutions is primarily limited to classroom instruction, with little or no emphases on hand-on experiences in computer installation and maintenance, trouble shooting, programming etc without occupational experience programs. Computer programming will no doubt evolve to a bookish, mere classroom experience with abstraction application and no practical work experience in the real world of work.

Agusiobo (1989) pointed out that the training environment in which the learner is trained would be a replica of the environment in which he subsequently works. The aim of teaching and learning of computer education is to produce individuals who will be able to put into practice what one has learnt. The need to program, maintain systems, network, installation etc in Nigeria demands that the students should be given every encouragement to develop interest in writing programming (Ogbonna and Okoli, 2007). This can only be achieved by exposing the students to skill acquisition so as to acquire sound programming experience.

The computer laboratory plays a significant role in the adoption of innovations and the transfer of skills in modern technological development of students. The computer laboratory is a pilot centre where scientific findings and innovations can be tried thoroughly and adjustments made before feedback is sent to researchers for improvements. It is a training ground where learners are exposed to sellable skills in computer science. While in the computer laboratory, students are able to handle computers, photocopiers, scanners, Internet resources etc and become involved in computer learning activities which make them become specialized in an aspect of programming.
As practice makes perfect, the computer laboratory makes the training of the computer students effective as it trains the individuals directly and specifically in the thinking and manipulative processes that are required in the world of programming. Experiences acquired in the computer laboratory lead to the right habits of doing and those of finished skills necessary for gainful employment.

In line with this, various governments in Nigeria have tried to improve practical teaching and learning of computer education in Colleges of Education by ensuring that practical in computer laboratory be conducted in the ratio of three students is to one computer (NCCE, 2004). Also, the introduction of student industries work experience (SIWES) which enables the students to embark on three (3) months industrial work experience in computer establishments in order to gain thorough practical experience needed in the real world of work. (FME, 2004).

Colleges of Education therefore establish computer laboratory on which the students receive practical experience and instruction in computer areas. In this case, the computer laboratory is regarded as an educational facility. The computer laboratory if well utilized can serve the following purposes:

1. To provide the students with knowledge and skills required in today’s automated office
2. To demonstrate programming practices or art for the students (good principles of programming)
3. To provide opportunities for students to practice what they have learnt in the classroom, that is, coordinate classroom theory and practice

Teachers and students of computer science should appreciate the importance of the school laboratory in transferring theory gained in the classroom into practice. The computer laboratory is the most convenient avenue for solving problems through practice, by test-running
and debugging learnt through observation and demonstration. Learning to solve real life problem through programming provides good ground for practical demonstration since success in programming is heavily dependent on problem solving abilities.

Computer programming activities in colleges of education would be interesting, relevant and effective on the part of the students, if there is a relationship between the content of the computer curriculum and the practical carried out in the computer laboratory.

Computer Science students, will no doubt be happier if what they are taught from the curriculum in the classroom is a replica of the activities carried out in the computer laboratory thus the acquired knowledge in the classroom will help them to develop faster in relevant skill related tasks.

**Method of Teaching Computer Programming**

One of the major problems in teaching programming is trying to implant food programming habits (principles). No matter how careful one is to introduce clean programming habits (restricted control flow, top-down design, meaningful data names, completeness in specifications and documentation, etc.) at the very beginning of students’ programming experience students too easily fall into quick and dirty programming. (Cherniak, 1976: 66-68; and Koffman & Friedman, 1976) have each proposed different approaches. Cherniak proposes that students should start as users of computer systems rather than as programmers. Students should be provided with proved program (i.e., canned programs) and with documentation on how to use them. Beginning students should be taught how to invoke these programs and be required to provide comments on their use. They should be introduced to language features in a progressive manner and be given programs that they can compare and modify. Some of the
advantages of this approach, as Chermiak indicates, are that the students sees the importance of program documentation, clarity, ease of use, modular structure, efficiency, etc. A different approach, taken by Russman and also by Koffman and Friedman, places more emphasis on the means to develop a program, rather than on the final goal (i.e. the program itself). Russman describes a course in which the computer’s ability to run programs is basically ignored by giving beginners, as their final goal, the development of structured flowcharts, instead of programs written in a particular programming language. Koffman and Friedman propose a similar approach along with the use of the computer to assist student during the development of their flow diagrams.

An important aspect of teaching programming, which has recently been considered, is the necessity of re-evaluating what should be taught in an introductory computer science courses (Gries, 1974; Hanson and Maly, 1975; and Cashman and Mein, (1975) in Ulloa (2005). As Hanson and Maly put it for a long time introductory computer science courses have degenerated into courses more appropriately titled introduction to the Programming Language X’ where X has historically been BASIC or FORTRAN and more recently PL/1 or APL.

New developments in Computer Science and programming technique, along with sound new methods of program development have also created a need to re-examine what is introduced to beginning programmers. As a result of this re-examination, Hanson and Maly propose “an introductory course which first and foremost, teaches problem-solving methodology with final stage of it being translation of an algorithm into a well-structure program.” To accomplish this objective, they present an algorithmic language with structured constructs that allows the programmer to concentrate on the logical structure of an algorithm by ignoring implementation
details such as declarations and data structures. The basic constructs of the language are the assignment statement, input and output (I/O), procedure call, and control structures that are essentially the same as those of PASCAL. Not until students learn how to develop algorithms in this language do they need to be introduced to an implementation language for writing the final program.

To enhance the means for writing structured algorithms and for developing programs in a stepwise refinement manner, program design language (PDL) are being introduced into the classroom (Bezanson 1975; Nanney, 1976; and Welch, 1976). Nanney (1976) describes a course in which structured thinking concepts are introduced with the aid of a pseudo-code which “at the least detailed level is structured form of (QBASIC).” An example of structured English is given by Bezanson, 1975 which he uses for teaching program development following a top-down, stepwise refinement approach. The algorithm, written in CLUE (Clear Unambiguous English) and given by Bezanson, is:

Algorithm for determining the greatest common divisor of two numbers.

1. Ask the user to give you two positive, integral numbers
2. Accept two numbers from the user, calling them M and N respectively
3. If the value of M is less than the value of N, then:
   3.1 Interchange the values of M and N
4. Repeat the following steps until instructed to stop:
   4.1 Divide the Value of M by the value of N, call the Remainder R.
   4.2 If the value of R is equal to 0, then;
      4.2.1 Inform the User that the answer is the value of N.
4.2.2 Terminate the algorithm.

4.3 Replace the value of M by the value of N.

4.4 Replace the Value of N by the value of R.

5. End of algorithm.

Another form of PDL that has been used in the classroom is reported by Welch (1996). Welch used the language for instructional purposes, such as describing actions of assembly language instructions, and students have used it to design algorithms and to document programs. The language includes special constructs for page headings (PROCEDURE, INSERT), paper references (CALL, SEE), and control structures (IF-ELSE, CASE, DO WHILE, DO UNTIL, WHEN). Statements are grouped into structures by indentations. An example given by Bakpo follows:

REM CALCULATE THE MEAN OF POSITIVE NUMBERS

PRINT “TO STOP, enter Zero”

LET SUM = 0

LET I = 1

Print “I = “; I; “ x = “;

INPUT X

DO WHILE X > 0

LET SUM = SUM + X

LET I = I + 1

PRINT “I = “; “x = “;

INPUT X
The idea of a student-oriented design language has been formalized and explored in more depth by (Ulloa, 1978). Ulloa describes a formal English-like PDL, a step-by-step procedure, and a set of software support tools for analyzing programming tasks and for developing programs in a stepwise refinement manner. The language consists of two subsets: one for describing the layout and requirements of a program’s input and output data, the other for writing algorithms and programs in various levels of abstraction. The program-development procedure presented is based on the idea of developing a program from a description of its inputs/output data and then incorporating the processing of the data by stepwise refinement. Automated tools used include a syntax and semantic analyzer, an I/O data-description checker, a data/program-consistency checker, tools for analyzing unrefined programs, and software for checking algorithms traces. Both the language and the procedure were developed, evaluated, and enhanced through their use in the classroom.

**Student-Oriented Systems**

There are two basic types of automated tools currently in vogue to help the teaching and learning of computer programming: interactive tools, many of which have been developed in an
effort to automate the teaching of introductory computer science courses (Nievergelt, 1974), and non interactive tools, most of which are simply student-oriented compilers.

**Interactive Tools**

The underlying philosophy behind providing interactive software is that highly responsive systems can serve as automated tutors to help students on an individual basis at their own pace and with their own particular problems. However, even when computer systems do not act as tutors, their highly responsive nature makes them nicely suitable for providing instructional models (Schweppe, 1973).

Interactive software tools have been developed for different purposes: to teach programming and documentation techniques (Fenichel, 1970 and Ward, 1974); to assist students to detect and analyze errors (Danielson, 1975; Tindall, 1975 and Gilleft, 1976), to teach about top-down programming (Danielson, 1975); to aid the development of flow charts (Koffman, and Friedman, 1976); and to provide a full learning laboratory (Barr and Bard, 1976).

**Debugging**

Experienced programmers agree that one of the major inconveniences in programming is debugging. Students without programming experience will, with more reasons, testify to this. For the experienced programmers debugging methods and tools have ranged from memory dumps and program traces in batch systems (Ferguson and Berner, 1963 and Satterthwai, 1972) to register and memory displays, breakpoint traces, forward and backward program stepping, and flow analysis in on-line or interactive systems (Balzer, 1969, Josephs, 1969, and Grishman, 1970). For experienced students, however, these method and tools have not been found suitable. Some methods require knowledge of low-level code, and some aids necessitate the learning of
special debugging languages or control statements. Basically, all of the aids demand the ability (generally acquired through experience) to know which tools to request, when and where to use them, and how to interpret the information. Therefore, aware of the learner’s problems, Tindall (1975), Davis (1975), and Gillett (1976) made efforts to provide student oriented debugging systems.

Detecting and correcting syntax errors have been the topics of much research (Leinuis, 1970 and Graham and Rhodes, 1975). Tindall was concerned with helping the student detect and analyze syntax errors. He built an interactive diagnostic system that can detect syntax errors as they are entered via a display terminal and can then provide the student with several levels of errors message and hints. The student has control of how much aid is needed, because the system is designed such that hints subsequent to the first error message are tailored according to the responses to prior message or hints. While Tindall’s system helps with compile-time errors Davis provides an interactive tool for analyzing run-time errors. This debugging system acts as a consultant by giving hints about possible causes of an error and inviting the user to either correct the error or ask for more assistance. Error analysis is aided with a table of common programming misconceptions and with several databases that correlate errors, causes of errors, and student misconceptions. Two basic types of analysis, static and dynamic, are available. First, the static analysis is carried out by examining, at the point of error, the static data structures of the program, the databases; and the table of misconceptions. From the analysis, a few “guessed” hints can be provided quickly, without much computer effort, in an attempt to make the student see the cause of error. This reverse execution is done with the aid of a history stack that contains changes in the values of variables, the flow of control, and the information of blocks. Basically,
this dynamic error analysis consists of keeping track of variables whose values, if altered, could
perhaps eliminate the detected error and of detecting and analyzing statements that can affect
these variables. If a possible cause of error is found in any of such statements, an appropriate hint
is shown to the students.

Syntax and run-time errors obviously comprise only a small part of students’ debugging
problems. Gillett proposed an Interactive Program Advising System (IPAS) to eliminate
“conceptual errors” from students’ programs and to increase the beginners’ understanding of
language constructs (Gillette, 1976). IPAS detects constructs that are the root of potential
conceptual errors and performs equivalent transformations that produce simpler and more
efficient source code. If the student agrees that the new code actually reflects what was originally
meant, then no conceptual error is involved. If on the other hand, the student does not agree, then
an error exists and must be corrected. To help the beginner understand the transformation
performed and correct the error, if any, the systems can show individual steps of the
transformation and suggest possible reasons for the error and corresponding corrective actions.

**Program Development Process:**

In an effort to make beginners understand the process of developing a program in a top-
down, stepwise refinement manner, Danielson (1975) provided an automated tutor for top-down
programming. This system mimics the action of a human tutor by presenting the student with a
programming problem to solve and by monitoring the attempt at solving it following a top-down
approach. The system’s basic objective is to interact with the student in a close-to-natural-
language form, to accept from the student refinements on how to solve the problem, to perform
some analysis on the validity of such refinements, and to provide hints or comments as
necessary. To accomplish this, the system contains databases with problems, acceptable methods of solution, and acceptable solutions. In addition, the system can provide multiple levels of hints and specific responses about certain types of errors. One highlight of the system is its ability to translate sufficiently precise descriptions given by the student into QB statements.

A different approach to teaching program development, given by Koffman and Friedman (1976), proposes an interactive system that can guide students in the development of “flow diagram descriptions of their algorithm to the system indicating any code.” The student must describe the algorithm to the system indicating the variables used, the flow diagrams needed, and the statements within the diagrams. During this process, the system provides hint about possible errors such as undefined variables, uninitialized loop parameters, and incorrect diagrams. Once refined variables to its lowest possible level, the students can translate the completed diagram to a particular programming language and execute the code. If the results are incorrect, the student can modify the diagram, change the code, and re-execute it. As Koffman and Friedman indicated, the assistance of the system in analyzing their diagrams should enable students to find most bugs in the flow diagram stage and eliminate subsequent patches to the program.

Barr and Beard (1996), describe an instructional system for learning QBASIC programming that plays the role of interactive tutor giving hints, correcting programming errors, and evaluating progress. The system include a curriculum of approximately 100 programming problems, a task selection and advising system, and an interpreter that provides numerous debugging aids. The task advising system provides help in problem solving by giving limit about the subtasks of a problem by indicating operations needed for each task, and by showing (upon student request) an algorithm flowchart of the problem. In addition, the system can provide
model solutions written in BASIC and check students’ solution by comparing the results of their programs with those of the model programs. The BASIC interpreter provides immediate feedback about syntax errors and gives limits about other types of errors as soon as they are detected. Some of these errors are run-time errors, program-structure errors detectable before execution, and errors associated with the requirements of the task or subtask being attempted. Debugging aids provided by the interpreter include TRACE and FLOW options for tracing variables and flow of control.

Ulloa (2006) however, noted that without certain precautions guiding the use of these range of student interactive tools that improve the learning environment of student programmers, these tools are likely to promote undesirable programming practices. He maintained that the system can provide extra-ordinary assistance; students can become quite careless, sometimes without realizing it. Perhaps more undesirably, they can be tempted to skip early program-development phases in order to get on with the program code and to obtain access to the system. Both manual and automated precautionary measures can reduce these undesirable effects. Manual measures are, basically, that the instructor requires the student to turn in flowcharts, trace tables, and program code inspection before being allowed to run the program and access the debugging tools. Obviously, this practice is not always feasible because of the lack of manpower, as well as human time and patience, especially with large classes. Automated precautionary measures, on the other hand, can be:

a. To require the student to access a system, such as that by Koffman and Friedman, before allowing access to program debugging tools;
b. To provide tools similar to that of Danielson that stress the importance of program development over that of producing program code;

c. To restrict the amount and type of help that the debugging tools provide

A possible solution is to impose a limit on the number and type of errors that qualify for system assistance at a given time. Recurrence of an error or type of error can be automatically analyzed or to warn about careless behaviour or bad programming practices. Although debugging tools have had a fair amount of attention, the same cannot be said of testing tools. Testing is an important programming activity that is closely related to debugging; it uncovers errors which debugging then takes over. Thorough testing of programs helps produce reliable programs and reliability of software is today a prime issue. Therefore, proper testing should be instilled in beginning programmers. Manual measures (example, requiring traces) to enforce sufficient testing are not always feasible. Therefore, automatic aids that emphasize the importance of testing and that can show beginners when and why their programs have not been properly tested can be very valuable (Ulloa, 2006).

Students would welcome more helpful with program documentation that is usually provided. Certainly any one can include comments within a program, but not everyone can provide a well-documented program. A basic problem faced by beginners in this area is not knowing where or what type of comments to include. As a result, many students either avoid including enough comments or expend their energies writing unnecessary or incorrect ones. Also, because of the lack of documentation enforcement, students tend to follow the prior practice of documenting their programs at the very end, after most of the debugging has been done. Ironically, one of the purposes of documentation is to aid in the debugging phase. Ward
(1974) aimed at teaching documentation techniques; there is still need to develop tools that can provide more help in this area.

**Student non-interactive tools for learning programming**

Student oriented compilers, also called diagnostic compilers, have been developed to improve the error detection, error diagnosis, and error recovery or correction capabilities of conventional compilers for widely-used languages such as BASIC, FORTRAN, PL/I, and COBOL.

QBASIC, an improved version of BASIC like WATFOR, and WATBOL etc are fast processors for BASIC, FORTRAN and COBOL respectively with similar capabilities. They can detect errors that are not normally detected by conventional compilers, and they provide better diagnostic messages that students can more easily understand (Ulloa, 2006). These compilers can recover from compile-time errors in such a way that execution of a program can be attempted even when errors have been detected. They provide easier-to-use debugging aids, such as flow of control tracing, and automatically give a trace back of subprogram calls upon detection of a run-time error. In particular, WATFIV provides numerous language extensions, such as free-format I/O and structured constructs. The PL/C compiler, implemented at Cornell University, provides a wide range of diagnostic features and debugging facilities for PL/1. Conway and Wilcox (1973) maintained that PL/C is unique in its tolerance of syntactic, semantic and execution errors. The compiler attempts a plausible repair of every source error so that execution may be attempted and the execution phase continues to repair errors until a user-specified cumulative error count is exceeded. In addition, the compiler provides various tracing and dumping facilities that the programmer can invoke and control as desired.
Student-oriented compilers can be used only after code has been written and is ready for execution and testing. Unlike these compilers, the software tools implemented in programming design can provide help in early, as well as late, phases of program development (Ulloa, 1978). Essentially, these tools can perform a variety of checks on input/output data descriptions, unrefined programs, and algorithm traces.

To write specifications, algorithms, and programs, the student uses a common vehicle of communication. This common vehicle is a student-oriented, English-like language with statements and constructions for writing data and unrefined programs descriptions as well as final executable programs. The language according to Ulloa (2006) consists of two highly compatible subsets: one for describing the layout and attributes of a program’s input and output data (the I/O data description); the other for describing high-level algorithms or programs at various levels of refinement (the Program Description). To develop a program, the student first writes the I/O Data Description (IODD), then generates a Program Description (PD) from the IODD, and finally refines, in a stepwise manner, the PD to obtain the final program code.

The software tools used during the development process can perform syntactic, semantic, consistency, and refinement checks. Five of these tools are: the Syntax and Semantic Analyzer, an Input Data Checker, the IODD-PD consistency checker, the PD Refinement Checker, and a Trace Checker. The analyzer sends parses, and checks the IODD, the PD, input data, and a trace sequence as given by the student. In addition, the analyzer builds the internal data structures and tokens for use by the other tools. The Input Data Checker Checks that any given input data description and the program description generated from it are consistent with each other. The consistency checker also checks that input and output statements and iteration and decision
structures in the PD are able to generate the I/O data and line patterns described in the IODD. The PD Refinement checker analyzes an algorithm or a program to find undefined or unrefined subtasks, computations, and variables. Finally, the trace checker can stimulate the execution of an unrefined or refined program using given input data to check if a trace sequence written by the student matches the stimulated flow of execution. To use the various tools, the programmer needs to submit only those parts handled by the individual tools hence individual tools can be invoked as the various phases of program development are completed (Ulloa, 2006).

Ulloa emphasized that as far as detection of syntactic and semantic errors is concerned, all of the diagnostic compilers can generally pinpoint an error or can provide an error message meaningful enough for many inexperienced programmer. According to Ulloa, the assistance provided to the beginner could be improved significantly if some of the error messages, particularly those by QB, WATRW and PL/C, were more to the point (e.g., “missing comma” instead of “missing operator”). Run-time error messages provide these compilers, on the other hand, are fairly clear about the error itself. However, they are not as informative as other error messages because of the nature of execution errors: their actual cause often lies somewhere away from the position where the error is detected. The beginner, lacking good perception of the sphere of possible causes of a run-time error, will sometimes opt for changing the statement cited by the message or will guess at its cause. To help alleviate this problem with run-time errors, the following possibilities were put forward according to Ulloa (2006).

a. Upon detection of a run-time error, provide both an error message explaining the error itself and a warning message explaining the possibility that another statement contains the actual cause of
the error. This warning could actually save the student some time and fruitless effort by a reminder of the problem that run-time errors present.

b. Perform some form of error analysis, similar to that described by Davis (1975) in order to produce, in addition to an error message, one or more hints about the possible cause of error.

c. Do more static analysis of the program (Semantic Checks) program structure checks, loop analysis, etc.) in an attempt to detect possible causes of errors before the errors actually appear during execution.

Considering that run-time errors present problems to many beginners, it should be pointed out that the highly touted ability of both PL/C and SP/K to repair execution-time errors is actually questionable (Ulloa, 2006). Even though the nature of the repair is reported to the student, the “correction” made by the system can produce unpredictable results: it can cause the program to produce unexpected output, it can be the cause of further error detection and repair, of, if everything does fall into place, it can actually, permit the program to execute “correctly” with the error present. To expect most inexperienced students to figure out the actual cause of a run-time error after getting wrong results, diagnostic messages, repair warnings, and so forth is highly optimistic. Provision of more intelligent diagnostic tools in earlier phases of program development is the answer to this dilemma (Ulloa, 2006).

Tools for Reliable Software Design in Programming

Software reliability has recently gained increased importance in the industry and in the business community (Ulloa, 2006). He emphasized the need for introducing beginning programmers to the area of software reliability early in their careers; summarizes some of the techniques and automated tools being used to enhance the reliability of programs in non-
educational environments, mentioned the techniques and tools that can be adapted to student environments to stress the importance of reliable software as well as assist students with some of their problems. Some of these software tools are the Program Evaluator and Tester System (PET), the AUDIT Program, the FORTRAN Automatic Code Evaluation System (FACES), the Product Assurance Confidence Evaluator System (PACE) the Extendable Debugging and Monitoring System (EXDAMS) and the PEARL System (Ulloa, 1978).

PET according to Stuart (1971) basic capabilities include: the ability to monitor the frequency of execution of source statements and of transfers taken; the ability to check user-written assertions for possible violations; and the ability to monitor the ranges of values of variables, arrays, and subscripts as requested by the programmer. Monitoring statement execution and transfers taken provides a simple yet very useful tool that shows how well test data exercises the program. As Knuth (1971) indicated profiles (that is, collections of frequency counts) should be available routinely to all programmers, this type of tool can be tailored so that the system shows explicitly what segments of code have not been tested and why, and perhaps provide suggestions on what to do to test them.

Stuart in Ulloa (2006) opined that assertion capabilities in PET are useful in three respects: (1), they allow the programmer to establish his own assertions at key points within his algorithms in the language he is now using, and to obtain dynamic analysis and feedback on the validity of his assertion. Secondly, assertions contribute to program documentation; and thirdly, assertions provide a means of stating and validating interface requirements between system moduls. In addition, assertion capabilities provide the programmer with a benefit intrinsic to the
use of assertions, that of forcing the programmer to consider the requirements and the program more carefully.

In her discussion of methods for teaching program verification, Gerhart (1975) points out: the very act of making assertions and attempting a proof elicits numerous assumptions and forces a rigorous check of programs which can often reduce later debugging time, catch subtle errors which would escape detection during testing, and lead to more pointed and useful documentation. Even if a “Proof” is not attempted and only informal assertions similar to those available in PET are written, some of the benefits to student programmers, according to Gerhart requires them to “present” a prose ‘argument’ that their programs are correct”. As she indicated this technique does force the students to consider their program in more depth and also serves as a good standard for documentation. Although this method can, certainly the fact is that a beginner does not require much direct feedback in this way, except for the few comments received when the instructor inspects the program and the prose “argument”. More feedback could result if, instead of an “argument” of correctness, beginners were required to include in their programs some simple basic assertions in a language understood by a system comparable to PET. Such a system according to Ulloa (2006) could point out any assertions that were inconsistent with the program, indicate the reasons, and provide suggestions on what to do.

In contrast with PET, which is a validation as well as a verification tool, the AUDIT System according to Culpepper (1975) is basically a validation tool only. Its two main functions are to perform an analysis of the structural characteristics of a program and to conduct a behavioural analysis. The structural analysis is aimed at ensuring that the program compiles with pre-stated specifications and with the rules of ANSI (American National Standard Instruction
This static analysis of the system detects among other errors, infinite loops, illegal branches within DO Loops, improper nesting of DO Loops, and disagreements between parameters of subprogram calls and parameters specified in the subprogram interface definition. The behavioural analysis, on the other hand, simply checks the sensitivity of the program to different word lengths.

An unusual problem among novice programmers (or among experienced ones, either) is their proneness to misinterpret or to disregard part of the program specifications. This tendency can cause a serious situation, since the consequences are not usually experienced until the program is basically completed. An automated, student-oriented tool with some of the characteristics of AUDIT would provide a solution to this problem. Provided with major requirements of a programming assignment the system could indicate to the student, beginning with the very first run, whether or not the program compiled with the instructor’s specifications.

Ramamoorthy and Ho (1975) stated that two important features of FACES are its ability to analyze program codes in order to detect incorrect or unreliable code and its ability to generate documentation in the form of cross reference tables and program-structure diagrams. Interface checks ensure proper alignment (number, types, and dimensions of entries) of subroutine parameters as well as common block parameters. Loop analysis includes identification of improperly nested loops and of data-sensitive loops (i.e., loops with variable parameters). These and all other checks are performed only upon request by the user through a query language, documentation generation is also optional.

It is important to note that several of the errors detected by FACES and AUDIT (e.g. improper meeting, disagreement of parameters) are already detected by student-oriented
computers such as WATFIV and PLK. Some other errors such as error-prone constructions, redundant code etc can also be detected by an interactive program advising system (Gillett, 1976). Concerning FACES ability to generate documentation, it is worth considering the potential usefulness of a similar tool in an educational environment, especially its usefulness for beginning students. As documentation per se, cross-reference table would be basically useful to show students different methods of documenting programs and how they should document their own. As debugging aids, cross-reference tables and program-structure diagrams could be somewhat useful if students were taught to use them properly.

Another important software tool according to Brown (1973) is PACE which is a collection of tools used in various stages of testing. Two of its elements are an execution – monitor program and a test – analysis program. The monitor program monitors execution to provide a list of unexecuted subroutines and to provide statistics on. (a) The frequency of execution of individual statements, subprograms, and small segments of code, and (b) test-effectiveness ratios (percentages) of the total number of executable statements, subroutines, and termination points that were executed. For each unexecuted segment of code, the test-analysis program provides a listing of the segment, indicates why it was not executed, and displays all the statement and computations of variables involved in a possible transfer to the segment.

This type of analysis, in combination with simple assertion analysis, would make a student-oriented testing tool more desirable.

It is not unusual to come across one or more unrevealed bugs while looking for a known one. Good tools can help find particular bugs and can also increase the chances of detecting hidden ones. The Extended debugging and monitoring system (EXDAM) is one of such tools
that enhance reliability of program testing (Balzer, 1969). EXDAMS’ superb on-line debugging facilities can enhance the reliability of a program. EXDAMS permits the user to:

a. watch the execution of the program at various speeds, moving forwards or backward;

b. to switch between different aids during execution;

c. to obtain static displays of (1) the values of different variables at a point of error, (2) all values of a variable up to a given point, and (3) portions of the source code at any point in execution;

d. to request a trace back analysis for any particular value of a variable in order to inspect all preceding computations that contributed to the production of the specified value;

e. to request a trace back analysis of the flow of control between any two points during execution;

and

f. To obtain motion- picture displays (i.e. sensitive to execution time) of values of variables and labels ordered according to execution time.

Although EXAMS’ aids are most useful in the hands of experienced programmers, some of its features could prove quite useful for beginners if such features were properly adapted to their needs. Barr and Bear, (1976) opined that a difficulty among beginners is their incorrect or insufficient understanding of the program execution process. They pointed out that interactive graphic debugging systems are important instructional tools that can greatly assist the students’ conceptualization of program execution, as well as teach her useful debugging techniques. The debugging aids provided by their BIP system aim at helping students in that respect.

The last of program development tools is Interactive PEARL System. PEARL according to Snowdon (1973) is based on the top-down program-development approach introduced by Dijkstra (1970) and on the verification techniques developed by King (1969) and Elspas (1973)
and Katz (1973). From the user’s standpoint, PEARL is important because it enables the user to construct programs in a top-down fashion and to attempt the “execution” of incompletely specified programs.

Based on the top-down approach, the user can describe the whole program in terms of abstract “mechanics”. Each machine consists of two parts, specification and program. The specification defines the machine by specifying the data types and operations that the particular machine understands. The program part contains the actual action (that is program instructions) to be performed by the machine. During the top-down development, the user can define the operations of a machine in terms of other machines, or optionally, can leave them undefined. Each when such undefined operations exit, the programmer can request execution of the program. Even when such undefined operations exist, the programmer can request execution of the program. Upon encountering an undefined operation, PEARL asks the programmer for assistance in order to continue executing and to attempt a possible solution. As Snowden (1976) indicated, by this technique a possible solution can be attempted without the need to program all the details.

**Related Empirical Studies**

UNESCO (2002) carried out a research on the ways and the means of meeting the constantly increasing demand on education. The following are part of the result of their finding:

1. In about two-third of the countries, financial responsibility of education does not rest on only federal, state or local government, but on two or more levels.

2. In countries where financial responsibilities for education are shared in the way, the tendency is for part of the expenditure on school building, equipment and materials.
3. In two out of five of fifty-five countries, plan for financial specific education programmes exists which do not come within the orbit of ordinary budget and whose terms exceed two years.

4. In three out of five countries, funds for financing education derived from general receipts of various administration, while in one out of four countries there are additional taxes levied for specific educational purpose. Only in very few instance does taxes revenue represent the main source of funds.

5. The number of countries in which there is a tendency for the higher administrative levels to increase their share of financial responsibility for education and their ease the burden of the local administrators is slightly higher than the member countries in which the opposite tendency is discernible.

6. The items of expenditures in which the largest increase have been made in recent years are those relating to school building and teachers’ salaries; such increases have been affected, not enough decreases in other items, but through raising the total budget. The implication of this study to this research work is that funding strategies can be employed to raise fund enough to enable school administrators to increase their expenditures on instructional facilities and school buildings required for effective teaching and learning of computer education programmes. This study has a similarity withn the present work but actually differed in scope. While the author’s study broadly worked at the strategies for meeting the constantly increasing demand on education, the present study attempts to limit itself to the strategies for enhancing teaching and learning of QBASIC programming language (CSC 112) in colleges of education.

Bala (2001) reported the contribution of association and organizations towards facilities provision for vocational education laboratories in colleges of education in Benue and Plateau
State. He used one hundred and twenty vocational teachers as respondents. The findings show that parent teachers tax fund 30 percent; philanthropists/philanthropic organizations 13 percents, foreign aids 10 percent and levies from students was 25 percent. Chambers (1955) compared the sources of income of private and public higher institutions and come up with the following results: public school fees have 18 percent, endowment income 2 percent, gifts 3 percent, governments 72 percent and other sources 5 percent. It is evident that the sources of funds for public higher institutions were mainly from various governments’ sources. This is only complemented by the students’ fees, which other sources of income contributed negligible amounts. This study has a significant relationship with the present work in the sense that the author’s study looked at the contributions of organizations towards facilities provision for vocational education laboratories in colleges of education, the present study is focused on the ways and means of enhancing teaching and learning of QBASIC programming language in colleges of education. However, the work differs from this study as it looked only on facilities provision for teaching and learning in colleges of education while the present study is focused on the overall strategies for enhancing teaching and learning of QBASIC (CSC 112) in colleges of education.

Furthermore, Ogbonna and Okoli (2007) undertook a study on Education Administration and the challenges of Computer Education as a tool for the Achievement of functional literacy at secondary school level in Imo State. Three research questions were formulated to guide the study. An 18 item questionnaire titled Computer Education Survey Questionnaire (CESQ) was developed and administered to 1200 respondents including management staff (Principals, Vice Principals, and Deans of Studies) and teachers and students in junior and senior secondary
schools. Stratified random sampling technique was used to select 1200 targeted respondents. Statistical instrument of mean was used for data analysis. The result of the study portrays the need for educational administrators to address the issue of a number of teachers available for computer education/institution in the state, their qualifications, use of teaching aids and the conduct of practical for computer lessons. This is very much related to this research work in that both studies identified educational administration as one of the strategies for enhancing teaching and learning of computer education. However, the work differs slightly from this study in that it looked at educational administration and the challenges of computer education as a tool for achievement of functional literacy at secondary school level, while the present study is focused on the overall strategies for enhancing teaching and learning of QBASIC course (CSC 112) in colleges of education.

Another study carried out by Ede (1999) focused on how to determine the administrative and teaching strategies for increasing the interest of senior secondary school students in technical drawing. The population of the study was made up of 35 principals, 72 technical drawing student and the 35 schools that offered technical drawing in Enugu State. No sampling was carried out. Administrative and teaching strategies questionnaire (ASTSQ) was used for data collection. Mean scores, t-test and analysis of variance (ANOVA) were used for analyzing data on strategies needed to increase student’s interest in technical drawing; (ii) Twenty-two teaching strategies for creating students’ interest in technical drawing, (iii) Fourteen tasks that senior secondary school students should perform to increase their interest in technical drawing; (v) Sex tasks that technical drawing teachers should perform to increase students’ interest in technical drawing.
Recommendations based on the findings were made. They include all administrative strategies to be adopted by the principals; the teaching students should do to increase their interest in technical drawing. This study is very much related to the present study in that both studies are concerned with administrative and teaching strategies for enhancing students’ performance. However, the work differs slightly from the present study in that the author’s study looked at the administrative and teaching strategies for increasing the interest of senior secondary school students in technical drawing course while the present study is concerned with strategies for enhancing teaching and learning of QBASIC (CSC 112) programming course in colleges of education.

Also, Hamma (1973) undertook a study on the strategies for increasing female students’ enrolment in technical subjects in Bauchi State. Three research questions and three null hypotheses were formulated to guide the study. A 51 item questionnaire was developed and administered to 87 teachers and 192 students from 3 Government secondary schools. Mean and standard deviation were used for data analysis. The result of the study portrays the need to recruit qualified technical teachers and provide regular in-service training for them to master skills to fully equip them for effective teaching. All necessary facilities, infrastructure and other inputs for technical education be provided. The community and industrial sector should contribute in providing facilities in technical college. Some kind of incentive is paid to female technical students to stimulate their interest. This study has a similarity with the present work but differs in scope. While the author’s study broadly worked at the strategies for increasing female students’ enrolment in technical subjects, the present study attempts to determine the strategies for enhancing teaching and learning of QBASIC programming language in colleges of education.
Similarly Fakorede (2006) carried out a study on the strategies for enhancing the marketability of technical education in south western States of Nigeria. Four research questions and three hypotheses were formulated to guide the study. A 60 items structured questionnaire was used as the instrument for data collection and administered to 1360 respondents comprised of 278 technical teachers and 82 industrial employers. Descriptive statistics mean, standard deviation and t-test statistical tool were employed for analyses of data. The result of the study portrays that promotional strategy of the marketing can be used in marketing technical education; policy formulation and review will enhance marketing of technical education products. Recommendation based on the findings were made among others that strategies need to be put in place for training of technical education students for better marketability. This study has a similarity with the present work but actually differs in the scope. While the author’s study broadly looked at the strategies for enhancing the marketability of technical education, the present study is focused on strategies for enhancing teaching and learning of QBASIC programming language (CSC 112). Both studies are interested in strategies that can be employed to enhance quality of students so that they can be marketable upon graduation.

**Summary of Literature Reviewed**

The development of a good computer program is a difficult task. Program development, like any other creative process, requires skills and experience in various activities involved in the teaching/learning of QB programming such as problem solving, debugging, flowchart, algorithm, psuedocode and testing. Not surprisingly, student programmers encounter numerous problems, many instructors/teachers express uneasy feeling that the objectives of the introductory
programming course are often not met; hence many students fail to develop the programming competency.

In an attempt to improve the effectiveness of teaching and learning of QBASIC programming course, new methods for teaching QB have been considered along with increased emphasis on principles of good programming on structured programming, step-wise refinement and top-down design. Also, better automated teaching tools have been considered as the way out of the problems that face beginning students in creating, coding, debugging, testing and documenting programs using QB language.

The review of literature to this study revealed that the main problems faced by students in the learning of QBASIC programming are found in four major areas; problem solving which results into acquiring bad programming habits, difficulty in learning top-down programming designs and difficulty in designing algorithms in a structured manner; (2) Testing which leads to misinterpretation of specifications. Not testing in early stages and not testing thoroughly; (3) Debugging which leads to not debugging before coding, difficulty in learning how to debug and problems with compile-time, run-time, conceptual, and other errors and lastly (4) Documentation which leads to not documenting at the right time in program development, not knowing what types of program comment to write or where in the program to write them and difficulty in learning other documentation means besides program comments.

The literature so far reviewed have shown that no study has been conducted on strategies for enhancing the teaching and learning of QBASIC programming in colleges of education in Enugu, Ebonyi and Anambra States to fill up this gap. Lack of such study therefore necessitates this present research work.
CHAPTER III
METHODOLOGY

This chapter presents the procedure adopted for this study under the following sub-headings: Design of the study; Area of the study; Population and sample of the study; Instrument for data collection and procedure for data analysis.

Research Design

This study adopted a survey research design. A survey research design according to Olaitan and Nwoke (1999) is one in which the entire population or representative sample is studied by collecting and analyzing data from a group through the use of questionnaire. The design is considered suitable since this study solicited information from Instructors and lecturers of computer science department on the strategies for enhancing the teaching and learning of QBASIC programming language in Nigerian colleges of education through the use of structured questionnaire.

Area of the study

This study was conducted in the old Anambra State in the south east geopolitical zone of Nigeria namely Enugu, Ebonyi and Anambra States. The choice of this area was based on the fact that the three states were formerly old Anambra State before the creation of states in 1990. Moreover, they share the same socio-cultural, educational and economic ideologies.

Population

The population for the study is 295 respondents made up of the entire 60 Computer Educators (35 Instructors and 25 Lecturers) and 235 year three Computer students of all the five
colleges of education in Anambra, Enugu and Ebonyi States. No sample was taken since all the population was studied.

**Instrument for data collection**

The instrument used for data collection in this study is a structured questionnaire. The questionnaire contains items sub-divided into five sections ‘A’ ‘B’ ‘C’ ‘D’ and ‘E’. Section ‘A’ is comprised of 8 items designed to find out the strategies for improving teacher-skill required for teaching and learning of QB in colleges of education in Enugu, Anambra and Ebonyi states. Section ‘B’ is comprised of 14 items designed to find out the strategies for improving teaching methodology needed for teaching and learning of QB programming in colleges of education in Enugu, Anambra and Ebonyi. While Section ‘C’ is made up of 9 items designed to determine the strategies for improving instructional facilities for teaching and learning of QBASIC programming in colleges of education in Enugu, Anambra and Ebonyi State. Section ‘D’ contains 11 questionnaire items that sought information on strategies for improving learning environment required for teaching and learning of QBASIC programming in colleges of education in Enugu, Anambra and Ebonyi State.

Lastly section ‘E’ contains 12 questionnaire items that sought information on the strategies for enhancing administration needs for teaching and leaning of QBASIC programming in colleges of education in Enugu, Anambra and Ebonyi States. The questionnaire items were formulated based on five point Likert scale type. The response categories for section ‘A’ to ‘E’ was strongly agree (SA), agree (A), undecided (UD) disagreed (D) and strongly disagree (SD). These response categories were assigned numerical values of 5,4,3,2, and 1 respectively. The
respondents were required to check ( ) against the response category that best satisfy their opinion.

**Validation of the instrument**

The research instrument was subjected to a face validation by three experts in the department of vocational teacher education (VTE) University of Nigeria Nsukka. They were requested to use their expertise in determining the suitability of the instrument items for data collection. Observations and suggestions made were used to improve the quality of the instruments. Face validation according to Uzoagulu (1998) is carried out to ascertain the appropriateness of the questionnaire items.

**Reliability of the Instrument**

The reliability of the instrument was established using Cronbach Alpha reliability test. A pilot study of 15 students and five Lecturers and Instructors of Alvan Ikoku College of Education Owerri which is outside the study area. This yields a reliability index of 0.99. This method was adopted because it ascertains the internal consistency of the instrument.

**Methods of Data Collection**

The instrument was administered by the researcher with the help of three trained research assistant to 295 respondents through personal contact. Only dully completed questionnaire returned was used for analyzing and answering the research questions and the hypotheses of the study.

**Methods of Data Analysis**

The data collection from the use of the questionnaire was analyzed using mean and standard deviation to answer each of the five research questions. However, each of the three
hypotheses was tested using t-test statistic at 0.05 level of significance. The analysis was computer based with the use of the statistical package for Social Sciences (SPSS 12.0 version).

The mean for the response scale was 3.00. The lower limit of the mean is 2.50 while the upper limit is 3.50 with an interval scale of 0.5 from the mean. Item with mean value of 3.50 and above was accepted while item with the mean value of less than 2.50 was rejected. The three null hypotheses were tested using t-test and the p-values were compared with 0.05 in each of the cases. Any item where p-value is greater than the t-values of 0.05, the hypothesis of no significant difference was upheld at probability of 0.05 level of significance; but where the p-value is less than 0.05, the hypothesis of no significant difference was rejected at 0.05 level of significant and at 58 degree of freedom.
CHAPTER IV

PRESENTATION AND ANALYSIS OF DATA

This chapter presents the result of the data analyses for the study. The presentations were organized according to the research questions and null hypotheses formulated to guide the study.

Research Question1

What are the strategies that can be employed to improve the technical skills of teachers for the teaching of QB in colleges of education in Anambra, Enugu and Ebonyi States?

Table 1

Mean and Standard deviation ratings of technical skills of Lecturers and Instructors required for the teaching of QBasic programming language in colleges of education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>DECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Providing ample opportunity for in service training for computer educators</td>
<td>3.93</td>
<td>1.22</td>
<td>AGREE</td>
</tr>
<tr>
<td>2</td>
<td>Certifying computer educators in troubleshooting and computer maintenance skills</td>
<td>3.90</td>
<td>1.14</td>
<td>AGREE</td>
</tr>
<tr>
<td>3</td>
<td>Making web design and networking skills a prerequisite for all computer educators</td>
<td>3.88</td>
<td>1.25</td>
<td>AGREE</td>
</tr>
<tr>
<td>4</td>
<td>Making debugging skills essential in teaching computer programming</td>
<td>4.12</td>
<td>1.12</td>
<td>AGREE</td>
</tr>
<tr>
<td>5</td>
<td>Making keyboarding skills a prerequisite for computer educators</td>
<td>4.00</td>
<td>1.14</td>
<td>AGREE</td>
</tr>
<tr>
<td>6</td>
<td>Making identification of program bug task such as syntax and semantic error mandatory for computer educators</td>
<td>4.15</td>
<td>1.01</td>
<td>AGREE</td>
</tr>
<tr>
<td>7</td>
<td>Making participation in at least three workshops, conference and seminars compulsory for computer education</td>
<td>3.95</td>
<td>1.05</td>
<td>AGREE</td>
</tr>
<tr>
<td>8</td>
<td>Making computer educators professionally certified.</td>
<td>3.72</td>
<td>1.19</td>
<td>AGREE</td>
</tr>
</tbody>
</table>
The data represented in Table 1 revealed that, the mean responses of computer educators and students to items 1,2,3,4,5,6,7 and 8 are greater than the cut-off point of 3.50. This indicated that majority of the respondents used for this study agreed with these items as the technical skills required for teaching of QB programming in colleges of education. The standard deviation of items 1-8 ranged from 1.01 – 1.25. This revealed that the respondents’ were close to one another in their opinion thus indicating that the respondents’ are not far from the mean.

Research Question 2

What are the strategies that can be employed for improving teaching methodology in teaching and learning of QB programming in colleges of education in Anambra, Enugu and Ebonyi States?

Computer educators (Instructors and Lecturers) and computer students were required to respond to this question. The responses obtained are presented in Table 2.
Table 2
Mean and standard deviation ratings lecturers’ and students responses on strategies that can be employed to improving the teaching methodology of QB programming in colleges of education in Anambra, Enugu and Ebonyi States

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>DECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Making teaching of good programming principles a topmost priority.</td>
<td>4.05</td>
<td>1.07</td>
<td>AGREE</td>
</tr>
<tr>
<td>2</td>
<td>Providing students with proved program and documentations on how to use</td>
<td>3.95</td>
<td>1.16</td>
<td>AGREE</td>
</tr>
<tr>
<td></td>
<td>them.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Teaching basic steps on how to load or invoke QB compilers.</td>
<td>4.09</td>
<td>1.05</td>
<td>AGREE</td>
</tr>
<tr>
<td>4</td>
<td>Teaching with use of canned programs that students can compare and modify.</td>
<td>3.77</td>
<td>1.19</td>
<td>AGREE</td>
</tr>
<tr>
<td>5</td>
<td>Teaching first the methodologies of problem-solving before the translation of algorithm into a well-structured program.</td>
<td>3.97</td>
<td>1.17</td>
<td>AGREE</td>
</tr>
<tr>
<td>6</td>
<td>Teaching algorithm development before implementation of the program code.</td>
<td>3.95</td>
<td>1.22</td>
<td>AGREE</td>
</tr>
<tr>
<td>7</td>
<td>Making top-down step-wise refinement approach compulsory in program development.</td>
<td>4.05</td>
<td>1.09</td>
<td>AGREE</td>
</tr>
<tr>
<td>8</td>
<td>Making automated teaching tools essential requirement for teaching programming courses.</td>
<td>3.97</td>
<td>1.13</td>
<td>AGREE</td>
</tr>
<tr>
<td>9</td>
<td>Making the use of Automated learning tools to enhance reliability of the programming.</td>
<td>4.09</td>
<td>1.09</td>
<td>AGREE</td>
</tr>
<tr>
<td>10</td>
<td>Identification of summary of students problems in programming lessons</td>
<td>4.14</td>
<td>1.02</td>
<td>AGREE</td>
</tr>
<tr>
<td>11</td>
<td>Making use of individual and group projects in teaching QB programming.</td>
<td>3.98</td>
<td>1.15</td>
<td>AGREE</td>
</tr>
<tr>
<td>12</td>
<td>Using collaborative participation in on-line tutorial lessons in programming.</td>
<td>3.58</td>
<td>1.30</td>
<td>AGREE</td>
</tr>
<tr>
<td>13</td>
<td>Giving assignments after every QB programming lessons</td>
<td>3.58</td>
<td>1.32</td>
<td>AGREE</td>
</tr>
<tr>
<td>14</td>
<td>Making use of Continuous Assessment Test (CAT) as appropriate means of evaluation of students.</td>
<td>3.96</td>
<td>1.15</td>
<td>AGREE</td>
</tr>
<tr>
<td>15</td>
<td>Making assessment of computer practical 60% and theory 40%.</td>
<td>3.85</td>
<td>1.31</td>
<td>AGREE</td>
</tr>
</tbody>
</table>

KEY: $\bar{x}$ = MEAN, A = AGREE, SD = STANDARD DEVIATION, D = DISAGREE

The data presented in Table 2 revealed that, items (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, and 14) had their mean above the cut-off point of 3.50. Therefore, majority of the respondents used for this study agreed with these items as strategies that could be used in improving the teaching of
QBasic programming in colleges of education. The standard deviation of the teaching methodology strategies ranged from 1.05-1.31. This showed that, the respondents were closed to one another in their responses and that their responses are not far fro the mean.

**Research Question 3**

*What are the strategies for improving the utilization of instructional facilities for the teaching and learning of QB programming in college of education in Anambra, Enugu and Ebonyi States?*

Computer educators (Instructors and Lecturers) and computer students were required to respond to this question. The responses obtained are presented in Table 3

**Table 3**

Mean and Standard deviation of teachers’ and students’ responses on the strategies for improving the utilization of instructional facilities in colleges of education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>¯x</th>
<th>SD</th>
<th>DECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Making students have free access to Internet</td>
<td>3.88</td>
<td>1.21</td>
<td>AGREE</td>
</tr>
<tr>
<td>2</td>
<td>Provision of standard library for every computer department by donor agencies.</td>
<td>4.03</td>
<td>1.07</td>
<td>AGREE</td>
</tr>
<tr>
<td>3</td>
<td>Use of computer for practical in ratios of one computer is to one student</td>
<td>3.39</td>
<td>1.37</td>
<td>AGREE</td>
</tr>
<tr>
<td>4</td>
<td>Making host communities contribute in donating computers and other teaching aids to computer department.</td>
<td>4.01</td>
<td>1.07</td>
<td>AGREE</td>
</tr>
<tr>
<td>5</td>
<td>Providing every computer educator with desktop and/laptop for class room demonstration.</td>
<td>3.93</td>
<td>1.10</td>
<td>AGREE</td>
</tr>
<tr>
<td>6</td>
<td>Provision of VSAT for internet connectivity for every computer department.</td>
<td>3.65</td>
<td>1.24</td>
<td>AGREE</td>
</tr>
<tr>
<td>7</td>
<td>Making the conduct of practical a sole function of computer Instructors.</td>
<td>3.55</td>
<td>1.41</td>
<td>AGREE</td>
</tr>
<tr>
<td>8</td>
<td>Provision of standby generators as a criterion for accreditation of every computer department.</td>
<td>3.37</td>
<td>1.31</td>
<td>AGREE</td>
</tr>
<tr>
<td>9</td>
<td>Making attendance to practical class mandatory for computer students.</td>
<td>3.97</td>
<td>1.09</td>
<td>AGREE</td>
</tr>
<tr>
<td>10</td>
<td>Providing every classroom with desktop computer for classroom demonstration by the teachers.</td>
<td>3.76</td>
<td>1.18</td>
<td>AGREE</td>
</tr>
</tbody>
</table>

**KEY:**  ¯x = MEAN, SD = STANDARD DEVIATION, A=AGREE, D = DISAGREE, DEC = DECISION

Table 3 showed that items 3, ‘use of computer for practical in ratios of one computer is to one student’ with mean of 3.39 and standard deviation of 1.37 and also item 8 ‘provision of
standby generators as a criterion for accreditation of every computer department with mean rating of 3.37 and standard deviation of 1.31 were agreed upon by the respondents. Also, the table further revealed that ten suggested strategies (items, 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10) to be used for utilization of instructional materials for teaching of QB in colleges of education, had their mean ratings above the cut-off point of 3.50. The standard deviation of the items ranged from 1.07-1.31. This revealed the respondents’ closeness to one another in their opinions thus, further revealed that respondents are not far from the mean.

**Research Question 4**

*What are the strategies for improving the learning environment for teaching and learning of QB in colleges of education in Anambra, Enugu and Ebonyi States?*

Computer educators (Instructors and Lecturers) and computer students were required to respond to this question. The responses obtained are presented in Table 4
### Table 4

**Mean and Standard deviation of teachers’ and students’ responses on the strategies for improving the learning environment for the teaching and learning of QB in colleges of education**

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>( \bar{x} )</th>
<th>SD</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Limiting each computer class stream to not more than 35 students.</td>
<td>4.03</td>
<td>1.06</td>
<td>AGREE</td>
</tr>
<tr>
<td>2</td>
<td>Using computers for practical in the ratio of one student is to one computer.</td>
<td>3.92</td>
<td>1.22</td>
<td>AGREE</td>
</tr>
<tr>
<td>3</td>
<td>Providing students with free access to computer laboratory in batches.</td>
<td>3.50</td>
<td>1.31</td>
<td>AGREE</td>
</tr>
<tr>
<td>4</td>
<td>Launching nation-wide program on computer oriented programmes</td>
<td>3.67</td>
<td>1.27</td>
<td>AGREE</td>
</tr>
<tr>
<td>5</td>
<td>Making computer students to participate in SIWES in computer related establishments only.</td>
<td>3.93</td>
<td>1.13</td>
<td>AGREE</td>
</tr>
<tr>
<td>6</td>
<td>Providing every computer department with standard library.</td>
<td>3.05</td>
<td>1.23</td>
<td>AGREE</td>
</tr>
<tr>
<td>7</td>
<td>Citing computer schools in metropolitan areas.</td>
<td>3.98</td>
<td>1.01</td>
<td>AGREE</td>
</tr>
<tr>
<td>8</td>
<td>Making access to electricity and basic infrastructures a pre-requisite for computer programmes.</td>
<td>2.97</td>
<td>1.18</td>
<td>D</td>
</tr>
<tr>
<td>9</td>
<td>Provision of computer centers and cyber cafes within and around collages of education.</td>
<td>3.91</td>
<td>1.04</td>
<td>AGREE</td>
</tr>
<tr>
<td>10</td>
<td>Provision of access to virtual libraries by students and teacher.</td>
<td>3.46</td>
<td>1.39</td>
<td>AGREE</td>
</tr>
</tbody>
</table>

**KEY:** A=AGREE, DEC = DECISION, SD=STANDARD DEVIATION, D =DISAGREE

The data in Table revealed that, items 1, 2, 3, 4, 6, 5, 7, 9 and 10 had their mean above cut-off point of 3.50. Therefore, the majority of the respondents used for this study agreed with these items as strategies for improving the learning environment for the teaching and learning of QB in colleges of education. Item 4, ‘limiting each computer class stream to not more than 35 students’ has the highest mean (4.03) on the table. The standard deviation of the learning environment strategies ranged from 1.06 to 1.39. This showed that the respondents were closed to one another in their responses and that their responses are not far from the mean.

The Table further revealed that the respondents disagreed on whether or not item 8 were among the learning environment strategies that could be used in enhancing the teaching and
learning of QB programming in colleges of education. The mean rating of this item was 2.97 fell below the cut-off point. Also the table revealed that, the standard deviation of the items ranged from 1.39-1.23. This indicates the respondents’ closeness to one another in their opinions and thus, further revealed that respondents are not far from the mean.

**Research Question 5**

*What are the strategies for improving the administration needs required for the teaching and learning of QB programming in colleges of education in Anambra, Enugu and Ebonyi States?*

Computer educators (Instructors and Lecturers) and computer students were required to respond to this question. The responses obtained are presented in Table 5
Table 5

Mean and standard deviation of lecturers’ and students’ responses on the strategies for improving administration needs required for teaching and learning of QB programming in Colleges of Education in Anambra, Enugu and Ebonyi States

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>DECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Given admission only to prospective candidates who possess minimum entry requirement.</td>
<td>4.05</td>
<td>1.04</td>
<td>AGREE</td>
</tr>
<tr>
<td>2</td>
<td>Payment of enhanced SIWES allowance to participating students of computer studies.</td>
<td>3.46</td>
<td>1.39</td>
<td>DISAGREE</td>
</tr>
<tr>
<td>3</td>
<td>Making school administrators collaborate with private sectors in provision of teaching aids.</td>
<td>3.90</td>
<td>1.08</td>
<td>AGREE</td>
</tr>
<tr>
<td>4</td>
<td>Making computer studies a top priority both at the national and state level.</td>
<td>3.81</td>
<td>1.20</td>
<td>AGREE</td>
</tr>
<tr>
<td>5</td>
<td>Using criterion reference test for evaluation of computer programme.</td>
<td>3.27</td>
<td>1.42</td>
<td>AGREE</td>
</tr>
<tr>
<td>6</td>
<td>Involving parents in funding and provision of instructional materials.</td>
<td>3.44</td>
<td>1.34</td>
<td>AGREE</td>
</tr>
<tr>
<td>7</td>
<td>Making Post UME/PCE screening test a criterion for admission into computer science program.</td>
<td>3.17</td>
<td>1.32</td>
<td>AGREE</td>
</tr>
<tr>
<td>8</td>
<td>Making school management sponsor computer educators in seminars and workshops.</td>
<td>3.41</td>
<td>1.29</td>
<td>AGREE</td>
</tr>
<tr>
<td>9</td>
<td>Payment of scarce skill allowance to computer instructors.</td>
<td>3.68</td>
<td>1.24</td>
<td>AGREE</td>
</tr>
<tr>
<td>10</td>
<td>Making teaching conditions of service for computer educators commensurate with those of private sectors.</td>
<td>4.03</td>
<td>1.04</td>
<td>AGREE</td>
</tr>
<tr>
<td>11</td>
<td>Providing special fund for retraining of computer educators.</td>
<td>3.57</td>
<td>1.32</td>
<td>AGREE</td>
</tr>
<tr>
<td>12</td>
<td>Payment of special allowance to teachers for SIWES supervision.</td>
<td>3.77</td>
<td>1.19</td>
<td>AGREE</td>
</tr>
<tr>
<td>13</td>
<td>Prompt payment of salaries of computer educators.</td>
<td>3.84</td>
<td>1.18</td>
<td>AGREE</td>
</tr>
<tr>
<td>14</td>
<td>Awarding of scholarship to outstanding students in programming courses.</td>
<td>3.88</td>
<td>1.08</td>
<td>AGREE</td>
</tr>
<tr>
<td>15</td>
<td>Providing students who distinguished themselves in programming courses with automatic employment.</td>
<td>3.73</td>
<td>1.15</td>
<td>AGREE</td>
</tr>
</tbody>
</table>

The data in Table 5 revealed that items 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15 had their mean above cut-off point of 3.50. Therefore, the respondents used for this study agreed with these items as strategies for improving the administration needs required for teaching and
learning of QB programming language in colleges of education. The data revealed that items 1 and 10 “giving admission only to prospective candidates who possess minimum entry requirement and “making teaching conditions of service for computer education commensurate with those of the private sectors’ have the highest mean ratings of 4.05 and 4.03 respectively. The standard deviation on the administration needs strategies required for teaching and learning of QB programming in colleges of education ranged from 1.04 - 1.42. This shows that, the respondents were close to one another in their responses and that their responses are not far from the mean.

**Hypothesis 1**

\[ H_0: \text{There is no mean significant difference (p < 0.05) in the responses of lecturers’ possessing a B.Sc. degree in Computer Science/Education and those possessing HND degrees with regard to the technical skills required for teaching and learning of QB in Colleges of Education in Enugu, Ebonyi and Anambra States.} \]

Computer educators (Instructors and Lecturers) and computer students were required to respond to this hypothesis. The responses obtained are presented in Table 6
### Table 6

The t-test analysis of the mean responses of lecturers with B.Sc. degrees in computer and those with HND on the technical skills required for teaching of QBasic programming in colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>Items</th>
<th>Group</th>
<th>N</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>$t$</th>
<th>df</th>
<th>Sig. (2-tailed)</th>
<th>DE-C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Providing ample opportunity for in service training for computer educators</td>
<td>HND</td>
<td>35</td>
<td>3.60</td>
<td>3.72</td>
<td>-1.33</td>
<td>58</td>
<td>.212</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.44</td>
<td>1.44</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Certifying computer educators in troubleshooting and computer maintenance skills</td>
<td>HND</td>
<td>35</td>
<td>3.85</td>
<td>4.04</td>
<td>-1.30</td>
<td>58</td>
<td>.220</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>4.04</td>
<td>1.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Making web design and networking skills a prerequisite for all computer educators</td>
<td>HND</td>
<td>35</td>
<td>3.25</td>
<td>3.92</td>
<td>-1.17</td>
<td>58</td>
<td>.282</td>
<td>S</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.92</td>
<td>1.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Making debugging skills essential in teaching computer programming</td>
<td>HND</td>
<td>35</td>
<td>3.94</td>
<td>4.04</td>
<td>-1.28</td>
<td>58</td>
<td>.280</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>4.04</td>
<td>1.20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Making keyboarding skills a prerequisite for computer educators</td>
<td>HND</td>
<td>35</td>
<td>3.60</td>
<td>3.84</td>
<td>-1.39</td>
<td>58</td>
<td>.212</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.84</td>
<td>1.24</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Making identification of program bug task such as syntax and semantic error mandatory for computer educators</td>
<td>HND</td>
<td>35</td>
<td>3.85</td>
<td>3.76</td>
<td>-1.21</td>
<td>58</td>
<td>.282</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.76</td>
<td>1.42</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Making participation in at least three workshops, conference and seminars compulsory for computer education</td>
<td>HND</td>
<td>35</td>
<td>3.68</td>
<td>3.96</td>
<td>-1.20</td>
<td>58</td>
<td>.282</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.96</td>
<td>1.17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Making computer educators professionally certified.</td>
<td>HND</td>
<td>35</td>
<td>3.61</td>
<td>3.85</td>
<td>-1.27</td>
<td>58</td>
<td>.282</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td>3.85</td>
<td>1.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GRANDMEAN

<table>
<thead>
<tr>
<th>Group</th>
<th>N</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>$t$</th>
<th>df</th>
<th>Sig. (2-tailed)</th>
<th>DE-C</th>
</tr>
</thead>
<tbody>
<tr>
<td>HND</td>
<td>35</td>
<td>3.61</td>
<td>3.85</td>
<td>1.27</td>
<td>58</td>
<td>.282</td>
<td>NS</td>
</tr>
<tr>
<td>B.SC</td>
<td>25</td>
<td>3.85</td>
<td>1.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

KEY: df = degree of freedom, P = level of significance (0.05), SD=Standard deviation, NS = not significant, S = significant,

The result in Table 6 showed the mean difference, standard deviation, t-value, sig.(2-tailed) and degree of freedom of t-test values calculated using SPSS (version 12.0) on the respondents responses on the technical skills required for teaching of QBasic programming in Colleges of Education. The result indicated that seven of the technical skills strategies items (1, 2, 4, 5, 6, 7, and 8) had their t-values on 58 degree of freedom and its p-values greater than 0.05
level. Therefore, the null hypotheses was accepted not to have any significant difference between
the mean ratings of responses of the two groups of respondent (B.Sc. lecturers and HND
Instructors) on the technical skills of teachers required for teaching of QB programming.

Meanwhile, the result of item 3 revealed that the value of t (-2.072) at 58 degree of
freedom and at 0.05 level of significance has its p-value, sig. (2-tailed) is 0.043 which is less
than 0.05 level. Therefore, the t-test result is significant at 0.05 level of significance. Thus the
null hypothesis is rejected showing that there is a difference in the opinions of Lectures’ and
Instructors on item 3.

However, the Table further revealed the grand mean with t-value on 58 degree of
freedom (-0.772) and p-value sig.(2-tailed) 0.444 greater than 0.05 level. Therefore, the null
hypotheses was accepted not to have any significant difference between the mean ratings of
responses of the two groups of respondents (Instructors and Lecturers) on the technical skills
strategies required for improving the teaching and learning of QBasic programming in colleges
of education.

Hypothesis2

\textbf{H02: There is no mean significant difference (p < 0.05) between the mean responses of
Lecturers’ with B.Sc. degrees and those with HND degrees with regard to the teaching
methodology required for teaching and learning of QB in colleges of education in Enugu,
Ebonyi and Anambra States.}

Computer educators (Instructors and Lecturers) and computer students were required to
respond to this hypothesis. The responses obtained are presented in Table 7
Table 7

The t-test analysis of the mean respondents of lecturers and Instructors with regard to the teaching methodology required for teaching and learning of QBasic programming in Colleges of Education

<table>
<thead>
<tr>
<th>SN</th>
<th>ITEMS</th>
<th>GROUP</th>
<th>N</th>
<th>$\bar{x}$</th>
<th>SD</th>
<th>$t$</th>
<th>df</th>
<th>Sig.(2-tailed)</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Making teaching of good programming principles a topmost priority.</td>
<td>HND</td>
<td>35</td>
<td>3.51</td>
<td>3.84</td>
<td>1.37</td>
<td>1.21</td>
<td>- .947</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>Providing students with proved program and documentations on how to use them.</td>
<td>B.SC</td>
<td>25</td>
<td>3.34</td>
<td>3.68</td>
<td>1.34</td>
<td>1.24</td>
<td>-.984</td>
<td>58</td>
</tr>
<tr>
<td>3</td>
<td>Teaching basic steps on how to load or invoke QB compilers.</td>
<td>HND</td>
<td>35</td>
<td>3.91</td>
<td>3.84</td>
<td>1.09</td>
<td>1.28</td>
<td>.241</td>
<td>58</td>
</tr>
<tr>
<td>4</td>
<td>Teaching with use of canned programs that students can compare and modify.</td>
<td>B.SC</td>
<td>25</td>
<td>3.71</td>
<td>3.64</td>
<td>1.17</td>
<td>1.38</td>
<td>-.224</td>
<td>58</td>
</tr>
<tr>
<td>5</td>
<td>Teaching first the methodologies of problem-solving before the translation of algorithm into a well-structured program.</td>
<td>HND</td>
<td>35</td>
<td>3.34</td>
<td>3.68</td>
<td>1.47</td>
<td>1.28</td>
<td>-.921</td>
<td>58</td>
</tr>
<tr>
<td>6</td>
<td>Teaching algorithm development before implementation of the program code.</td>
<td>B.SC</td>
<td>25</td>
<td>3.48</td>
<td>3.76</td>
<td>1.37</td>
<td>1.20</td>
<td>- .801</td>
<td>58</td>
</tr>
<tr>
<td>7</td>
<td>Making top-down step-wise refinement approach compulsory in program development.</td>
<td>HND</td>
<td>35</td>
<td>3.57</td>
<td>3.32</td>
<td>1.03</td>
<td>1.40</td>
<td>.798</td>
<td>58</td>
</tr>
<tr>
<td>8</td>
<td>Making automated teaching tools essential requirement for teaching programming courses.</td>
<td>B.SC</td>
<td>25</td>
<td>3.62</td>
<td>3.72</td>
<td>1.28</td>
<td>1.48</td>
<td>-.254</td>
<td>58</td>
</tr>
<tr>
<td>9</td>
<td>Making the use of Automated learning tools to enhance reliability of the programming.</td>
<td>HND</td>
<td>35</td>
<td>3.65</td>
<td>4.12</td>
<td>1.30</td>
<td>1.20</td>
<td>-1.399</td>
<td>58</td>
</tr>
<tr>
<td>10</td>
<td>Identification of summary of students problems in programming lessons</td>
<td>B.SC</td>
<td>25</td>
<td>3.74</td>
<td>4.08</td>
<td>1.17</td>
<td>1.11</td>
<td>-1.121</td>
<td>58</td>
</tr>
<tr>
<td>11</td>
<td>Making use of individual and group projects in teaching QB programming.</td>
<td>HND</td>
<td>35</td>
<td>3.22</td>
<td>3.48</td>
<td>1.55</td>
<td>1.44</td>
<td>-.635</td>
<td>58</td>
</tr>
<tr>
<td>12</td>
<td>Using collaborative participation in on-line tutorial lessons in programming.</td>
<td>B.SC</td>
<td>25</td>
<td>3.05</td>
<td>3.56</td>
<td>1.58</td>
<td>1.38</td>
<td>-1.273</td>
<td>58</td>
</tr>
<tr>
<td>13</td>
<td>Giving assignments after every QB programming lessons</td>
<td>HND</td>
<td>35</td>
<td>3.34</td>
<td>3.60</td>
<td>1.47</td>
<td>1.29</td>
<td>-.701</td>
<td>58</td>
</tr>
<tr>
<td>14</td>
<td>Making use of Continuous Assessment Test (CAT) as appropriate means of evaluation of students.</td>
<td>B.SC</td>
<td>25</td>
<td>3.60</td>
<td>3.20</td>
<td>1.39</td>
<td>1.35</td>
<td>1.107</td>
<td>58</td>
</tr>
<tr>
<td>15</td>
<td>Making assessment of computer practical 60% and theory 40%.</td>
<td>HND</td>
<td>35</td>
<td>2.80</td>
<td>3.28</td>
<td>1.38</td>
<td>1.54</td>
<td>-1.261</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>GRAND MEAN</td>
<td>B.SC</td>
<td>25</td>
<td>3.46</td>
<td>3.65</td>
<td>1.29</td>
<td>1.27</td>
<td>-.565</td>
<td>58</td>
</tr>
</tbody>
</table>
Data presented in Table 7 presented statistics of the distribution of differences between the paired scores (paired difference) at 0.05 level of significance, the values of t, its degree of freedom, and its p-values sig.(2-tailed). The output revealed that, each of the items on teaching methodology strategies that could be used in improving teaching and learning of QBasic programming in Colleges of education had their p-values at 58 degree of freedom on items (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15) greater than the 0.05 level. This indicated that there was no significance difference between the mean response of lecturers and Instructors on the teaching methodology strategies that could be used in improving the teaching and learning of QBasic programming language. With this result the null hypothesis (Ho) of no significant difference was upheld at 0.05 level of significance.

**Hypothesis 3**

*H03: There is no mean significant difference (p < 0.05) between the responses of lecturers’ with B.Sc. degree and those with HND degree with regard to the use of instructional facilities required for teaching and learning of QB Colleges of Education in Enugu, Ebonyi and Anambra States.*

Computer educators (Instructors and Lecturers) and computer students were required to respond to this hypothesis. The responses obtained are presented in Table 8
Table 8

The t-test analysis of the mean respondents of lecturers and Instructors with regard to the instructional facilities utilization strategies required for teaching and learning of QBasic programming in Colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>GROUP</th>
<th>N</th>
<th>x</th>
<th>SD</th>
<th>t</th>
<th>df</th>
<th>Sig.(2-Tailed)</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Making students have free access to Internet</td>
<td>HND</td>
<td>35</td>
<td>3.34</td>
<td>3.36</td>
<td>-0.044</td>
<td>58</td>
<td>.965</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Provision of standard library for every computer department by donor agencies.</td>
<td>HND</td>
<td>35</td>
<td>3.41</td>
<td>3.60</td>
<td>0.241</td>
<td>58</td>
<td>.836</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Use of computer for practical in ratios of one computer is to one student</td>
<td>HND</td>
<td>35</td>
<td>3.54</td>
<td>3.88</td>
<td>1.22</td>
<td>106</td>
<td>.293</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Making host communities contribute in donating computers and other teaching aids to computer department.</td>
<td>HND</td>
<td>35</td>
<td>3.25</td>
<td>3.04</td>
<td>1.40</td>
<td>1.33</td>
<td>.603</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Providing every computer educator with desktop and/laptop for class room demonstration.</td>
<td>HND</td>
<td>35</td>
<td>3.67</td>
<td>3.28</td>
<td>1.24</td>
<td>1.20</td>
<td>-1.668</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Provision of VSAT for internet connectivity for every computer department.</td>
<td>HND</td>
<td>35</td>
<td>3.57</td>
<td>3.24</td>
<td>1.24</td>
<td>1.20</td>
<td>-0.968</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Making the conduct of practical a sole function of computer Instructors.</td>
<td>HND</td>
<td>35</td>
<td>3.57</td>
<td>2.84</td>
<td>1.35</td>
<td>1.40</td>
<td>-2.029</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Provision of standby generators as a criterion for accreditation of every computer department.</td>
<td>HND</td>
<td>35</td>
<td>2.91</td>
<td>2.88</td>
<td>1.06</td>
<td>1.33</td>
<td>-0.111</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Making attendance to practical class mandatory for computer students.</td>
<td>HND</td>
<td>35</td>
<td>3.45</td>
<td>2.80</td>
<td>1.23</td>
<td>1.47</td>
<td>-1.628</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Providing every classroom with desktop computer for classroom demonstration by the teachers.</td>
<td>HND</td>
<td>35</td>
<td>3.45</td>
<td>3.12</td>
<td>1.29</td>
<td>1.45</td>
<td>-0.946</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GRAND MEAN</td>
<td>HND</td>
<td>35</td>
<td>3.40</td>
<td>3.37</td>
<td>1.24</td>
<td>1.28</td>
<td>-0.065</td>
<td>NS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B.SC</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data presented in Table 8 revealed that, each of the items on Instructional facilities utilization strategies that could be used in improving teaching and learning of QBasic
programming language in colleges of education had p-values, sig.(2-tailed) on items (1, 2, 3, 4, 5, 6, 8, 9, and 10) greater than the t-values at 0.05 level of significance and 58 degree of freedom. This indicated that there was no significance difference between the mean responses of lecturers’ and Instructors’ on the instructional facilities utilization that could be used in improving the teaching and learning of QBASIC programming language. With this result the null hypotheses (Ho) of no significant difference was upheld at 0.05 level of significance.

Meanwhile, item 7, “making conduct of practical a sole function of computer Instructors”, had its t-value at 58 degree of freedom (2.029) with its p-value, sig.(2-tailed) (0.047) which is less than 0.05. Therefore, this result indicated that the t-test is significant at 0.05 level of significance thus, the null hypothesis is rejected. This means that there is a significant difference in the opinions of Lecturers and Instructors on making the conduct of practical a sole function of Instructors.

However, the Table further revealed the grand mean with t-value at 58 degree of freedom (.065) and p-value sig.(2-tailed) (0.949) greater than 0.05 level. Therefore, the null hypotheses was accepted not to have any significant difference between the mean ratings of responses of the two groups of respondents (Instructors and Lecturers) on the instructional facilities utilization for improving the teaching and learning of QBASIC programming in colleges of education.

Findings of the Study

The following findings emerged from the study based on the research questions and hypotheses tested:

1. Providing ample opportunity for in service training for all computer educators was accepted as a strategy for improving the technical skills of computer educators.
2. Certifying computer educators in troubleshooting and computer maintenance skills as essential strategy for improving the technical skills of teachers.

3. Making the teaching of debugging skills essential in teaching computer programming by computer educators was accepted as strategy for improving the teaching of QBasic.

4. That identification of program bug task, such as syntax and semantic error become mandatory for computer educators was accepted as an essential strategy for teaching QBasic programming language.

5. Making teaching of good programming principles a topmost priority for teaching and learning of QBasic programming language.

6. That providing students with proved program and documentations on how to use them was accepted as a strategy for improving teaching and learning of QBasic language.

7. Teaching basic steps on how to load or invoke QB compilers was accepted as a strategy for improving the teaching and learning of QBasic programming language.

8. Teaching first the methodologies of problem-solving before the translation of algorithm into a well-structured program was accepted as a way of improving teaching of QBasic programming language.

9. Teaching first the development of algorithm before implementation of the program code was accepted as a strategy for improving teaching of QBasic programming language.

10. That making top-down step-wise refinement approach compulsory in teaching program development.

11. Making the use of automated learning tools essential in other to enhance the reliability of programming development.
12. Identification of summary of students’ problems in programming lessons was accepted as an important strategy for improving teaching of QBasic programming language.

13. Making use of Continuous Assessment Test (CAT) as appropriate means of evaluation of students in programming lessons.

14. Provision of standard library for every computer department by donor agencies was accepted as a strategy for improving the instructional facilities utilization.

15. Use of computer for practical in ratios of one computer is to one student was accepted as a strategy for improving instructional facilities utilization in QBasic programming.

16. Providing every computer educator with desktop and laptop for class room demonstration was accepted as a strategy for improving instructional facilities utilization in QBasic programming language in colleges of education.

17. Provision of VSAT for internet connectivity for every computer department was accepted as a strategy for improving instructional facilities utilization in teaching and learning of QBasic programming in colleges of education.

18. Making attendance to practical class mandatory for computer students was accepted as a strategy for improving the teaching methodology strategy for learning of QBasic programming language in colleges of education.

19. Limiting each computer class stream to not more than 35 students was accepted as a strategy for improving instructional facilities utilization in teaching and learning of QBasic programming in colleges of education.
20. Making computer students to participate in SIWES in computer related establishments only was accepted as a strategy for improving instructional facilities utilization required for enhancing the teaching and learning of QBasic programming.

21. Provision of computer centers and cyber cafes within and around collages of education was accepted as a strategy for improving the learning environment required for the teaching and learning of QBasic programming in colleges of education.

22. Given admission only to prospective candidates who possess minimum entry requirement was accepted as a strategy for improving administration needs required for teaching and learning of QBasic programming in colleges of education.

23. Payment of enhance SIWES allowance to participating students of computer studies was accepted as a strategy for improving instructional facilities utilization required for teaching and learning of QBasic programming.

24. Making school management sponsor computer educators in seminars and workshops were accepted as a strategy for improving administration needs required for the teaching and learning of QBasic programming.

25. Making teaching conditions of service for computer educators commensurate with those of private sector was accepted as a strategy for improving administration needs required for the teaching and learning of QBasic programming language in colleges of education.

26. There was no mean significant difference between the responses of Lecturers with B.Sc. and Instructors with HND on the technical skills required for teaching of QBasic programming language in colleges of education.
27 There was no mean significant difference between the responses of lecturers with B.Sc. degree and Instructors with HND degree on the teaching methodology strategies required for the teaching and learning of QBasic programming language in colleges of education.

38 There was no mean significant difference between the responses of lecturers and Instructors with B.Sc. degree in computer and those with HND degree on the instructional facilities utilization strategies required for the teaching and learning of QBasic programming language in colleges of education in Nigeria.

Discussion of findings

The Technical skills of teachers that will improve the teaching of QBasic programming language in Colleges of Education

The data presented in Table 1 provide answers to the research questions one. This finding is in consonance with Okoli (2005) remark that computer instruction is done with little or no emphasis on hand-on experiences in computer installation, trouble shooting, maintenance, programming etc without occupational experience programs. In the same vain, Uloa (2005) earlier pointed out the need for teachers to help students in detecting and correcting syntax errors. He emphasized the use of interactive diagnostic system that can detect syntax errors as they are entered through a display terminal thus providing the students with several levels of error message and hint. The respondents of this study also responded with high mean on the provision of computer educators with in service training to enable teachers acquire and improve on their technical skills so that the learners can easily benefit from their wealth of experience.

Since QBasic programming language is a skill oriented programme aimed at manpower development, this study found out that sound technical skills training is required by educators in
order to enable the learners understand the skills of QBasic programming language needed for self-reliance.

**Strategies for improving the teaching methodology required for teaching and learning of QBasic programming language in Colleges of Education**

The data presented in Table 2, provided answers to research question 2. The findings revealed that appropriate teaching methodology will help to improve the understanding of students in QBasic programming language. The teaching of good programming principles such as top-down step-wise refinement approach should be made a top-priority by computer educators. This finding is in line with Cherniak in Uloa (2005) that inability of teachers to implanting good programming principles in the learners at the beginning stage of students’ programming experience; they will easily fall into quick and dirty programming.

The findings of this study further revealed that computer educators should re-evaluate what is taught to students in an introductory computer science course such as QBasic programming. Teachers are expected to provide beginners programmers with proved programs and documentations on how to use them, basic steps on how to load QBasic compiler, teaching algorithm development before implementation of the program code will go a long way to improve students understanding of QBasic programming language.

Lack of appropriate teaching methods in teaching programming courses especially to beginners’ programmers may cause them to change courses or even overlook programming courses because of the relative difficulty in understanding the skills of QBasic language. The reality is that the cost of programming continues to rise because of shortages of skilled personnel required for programming (French, 1996). It is therefore obvious that computer educators need to
master these principles so as to equip the learners with appropriate programming skill training required to excel in their related work tasks.

The findings further revealed that automated teaching tools are essential requirement for teaching programming courses. With automated teaching tools available, beginners programmers can be encouraged to participate in collaborative on-line tutorial lessons in programming courses like QBaBasic thus they will improve their programming skills. This is in line with Uloa (2005) remark that providing students with interactive software that is highly responsive system can serve as automated tutors that can help students on an individual basis at their own pace and with their own problems. The reality is that interactive software (automated) tools have been developed for different purposes and therefore can assists the learners to learn programming and documentation techniques, detecting and analyzing errors, to teach top-down programming principle as well as to aid the development of flow charts (Uloa, 2005).

The findings further revealed that evaluation of students’ performance on programming skills should be based upon 60% for practical and 40% for theoretical work. This will facilitate sound academic performance in the students since grades obtained in theory alone will not be enough to pass them.

**Strategies for improving the utilization of Instructional facilities for the teaching and learning of QBasic programming language in Colleges of Education**

The data presented in table 3 provided answers to research question three. The findings revealed that QBaBasic programming language will be responsive if instructional materials are appropriately utilized for knowledge delivery. This is in line with Bakpo (2005) assertion that computer laboratory should be provided with adequate instructional facilities for skill training in programming to compliment whatever background knowledge acquired in the classroom in order
to develop the much needed entry level skills for programming by National colleges of education computer students.

The findings further revealed that students should be made to have free access to Internet, providing students with adequate computer systems through collaboration with school administrators, host communities and donor agencies such that the learners can use these systems in ratios that can facilitate increased use of computers for practical drills unhindered.

**Strategies for improving the learning environment required for the teaching and learning of QBasic programming in Colleges of Education**

The data presented in Table four provided answers to research question four. The findings revealed that teaching and learning of QBasic programming language can be improved considerably if the classroom environment is fashioned to become what it should be. The principal findings indicated that every computer class stream should be limited to not more than 35 students such that the students will be able to use the computers in a ratio of one student is to one computer during practical drills. This findings is in line with Ogbonna and Okoli (2007) that the need to program, maintain systems, network and perform installations demands that the students should be given encouragement to develop interest in programming.

The findings further revealed that citing computer schools in metropolitan areas where basic infrastructures like accessible roads and electricity will facilitate the establishments of cyber café within and around colleges of education will go along way in improving the students’ access to computers for their practical drills.
Strategies for improving the administration needs required for the teaching and learning of QBasic programming language in Colleges of Education

The data presented in table 5 provided answers to research question five. The findings revealed that teaching and learning of QBasic programming can be improved upon through the use of the administration needs strategies mainly giving admission to only prospective candidates who are qualified and possess the minimum entry requirement. Thus ensuring that class stream of 35 is not exceeded.

The findings further revealed that school administrators should facilitate the re-training of computer educators through workshops, conferences and seminars. The findings also revealed that school administrators should make teaching conditions commensurate with those of the private sectors through payment of scarce skill allowance to computer educators, payment of special allowance to teachers for SIWES supervision, prompt payment of salaries and awarding of scholarship to students and computer educators who distinguish themselves in areas of programming.

The t-test analysis of the technical skills strategies required for the teaching of QBasic programming in colleges of Education

The analysis of hypothesis one shown in table 6 indicated the comparison of the respondents (B.Sc. Lecturers and HND Instructors) on the technical skills that will improve the teaching of QBasic programming language in colleges of education tested with t-test statistics at 0.05 level of significance and at 58 degree of freedom. The findings indicated that seven of the eight items of technical skills strategies were accepted by the respondents as being capable strategies that can enhance the teaching of QBasic programming language in colleges of education. This is based on the result of the calculated grand mean and p-value of all the items of
the cluster which were greater than the value of 0.05 Therefore, it could be deduced that there was no significant difference between the mean responses of Lecturers and Instructors on the technical skills of teachers required for teaching of QBasic programming language in Colleges of Education in Enugu, Ebonyi and Anambra States. The implication of these findings was that it helped to confirm the findings made in research question one.

The t-test analysis of the teaching methodology strategies required for the teaching and learning of QBasic programming in Colleges of Education

The same t-test significant was used for the second hypothesis formulated on strategies for improving the teaching methodology required for the teaching and learning of QBasic programming language in colleges of education was presented in table 7. The calculated grand mean and p-value for all the 15 items of the cluster were greater than the t-value of 0.05. Therefore, null hypothesis was upheld at 0.05 level of significant. On these items, it is noted that Instructors and Lecturers agreed on the teaching methodology strategies as ways of improving the teaching and learning of QBasic programming language in Colleges of Education in Enugu, Anambra and Ebonyi States.

The t-test analysis of the instructional facilities utilization strategies required for the teaching and learning of QBasic programming in Colleges of Education

A t-test of significance was used to test the third hypothesis on the strategies for the utilization of instructional facilities for the teaching and learning of QBasic programming language was presented in table 8. The calculated grand mean and p-value for all the 10 items of the cluster had their p-values greater than 0.05; hence the null hypothesis was accepted not to have any significant difference between the mean ratings of the two groups of respondents (Instructors and Lecturers) on the instructional facilities utilization strategies that could be employed to improve the teaching and learning of QBasic programming in colleges of
education in Enugu, Ebonyi and Anambra States. The implication of the findings is that it helped to validate the findings of research question 3 that there is agreement between the Lecturers and Instructors on the strategies that can be employed on the utilization of instructional facilities for effective teaching and learning of QBasic programming language in Colleges of Education in Enugu, Ebonyi and Anambra States.
CHAPTER V
SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

This chapter presents the summaries of statement of the problem, procedures adopted in conducting the study and major findings of the study. Also presented in the chapter are conclusion, recommendations based on the findings of the study, implications of the findings and suggestions for further studies.

Re-statement of the problem

Colleges of education in the light of the essence of their establishment are charged with the responsibility of producing teacher educators, technologist as well as other middle level manpower for technological and economical advancement of the nation. QBASIC is therefore one of the numerous courses offered by Nigerian colleges of education packaged with a number of programming courses to enable computer students acquire technological and programming skills needed to live and work in today’s knowledge-based economy. Lack of technical and programming skills by computer educators and students is distressing to the production of quality computer graduates. More distressing to the production of quality computer graduates programmers is the situation where instructional facilities for teaching and learning of QBasic and other programming courses are either unavailable or grossly under utilized. Also worrisome to the teaching and learning of computer programming is that most educators lack the appropriate teaching methodology required for effective transfer of knowledge to the students.

In colleges of education in the south east states of Nigeria namely Enugu, Anambra and Ebonyi States, there is poor performance students in QBasic programming language. Thus the products of these institutions remain unemployed years after graduation because the lack the
programming skills but nonetheless, there exist a growing demand of computer programmers in our industries. Based on this problem, the following specific objectives were pursued:

1. to determine the strategies for improving the technical skills of the teachers for the teaching of QB in Colleges of Education in Anambra, Ebonyi and Enugu States.
2. to determine the strategies for improving the teaching methodology required for the teaching and learning of QB in Colleges of Education in Anambra Ebonyi and Enugu States.
3. to identify the strategies for improving the utilization of instructional facilities for the teaching and learning of QB programming in Colleges of Education in Enugu, Ebonyi and Anambra States.
4. to identify the strategies for improving learning environment for the teaching and learning of QB programming in colleges of Education in Anambra, Ebonyi and Enugu States.
5. to identify strategies for improving administrative needs for the teaching and learning of QB in Colleges of Education in Anambra, Ebonyi and Enugu States.

**Summary of Procedures used for the study**

The study was a survey research design, questionnaire was the instrument used to collect data for indentifying the strategies for enhancing the teaching and learning of QBasic programming language in colleges of education in Enugu, Anambra and Ebonyi states of south east geopolitical zone of Nigeria. A set of 58 structured questionnaire items was used to gather information from 295 respondents made of 25 Lecturers, 35 Instructors and 235 computer science students. The questionnaire was validated by three expects from the department of
Vocational Teacher Education, University of Nigeria Nsukka and its reliability was established using Cronbach alpha formula for determining the internal consistency. Reliability result of 0.99 was calculated for the instruments after its trial test on 15 computer students and 5 computer lecturers from Alvan Ikoku College of Education Owerri.

The data was collected with the assistance of three research assistance (RA’s) (one RA per state), and analyzed by computer SPSS 12.0 version using mean, standard deviation and t-test. The mean and standard deviation were used to answer the five research questions while t-test statistics was used to test the hypotheses.

**Summary of findings**

Based on the data collected and analyzed the following major findings of the study on strategies for enhancing the teaching and learning of QBasic programming language are:

1. Eight strategies on technical skills that will enhance the teaching of QBasic programming language were in strong agreement with the opinions of the respondents (Lecturers’, Instructors and computer students). This was with means ranging from 3.72 - 4.15.

2. A total of 24 items were considered as what should be the strategies for improving the teaching methodology required for teaching and learning of QBasic programming language. They were in agreement with the opinions of the respondents with means ranging from 3.58 - 4.14.

3. All the ten items on the strategies for improving the utilization of instructional facilities for teaching and learning of QBasic programming language were agreed to as important by the respondents with means ranging from 3.39 - 4.01.
4. Nine of the ten items on the strategies for improving the learning environment required for the teaching and learning of QBasic programming language were agreed to as important by the respondents with means ranging from 3.46 - 4.07.

5. All the 15 items on the administration needs for enhancing the teaching and learning of QBasic programming language were agreed to as important by the respondents with means ranging from 3.21 - 4.05.

6. There was no significant difference between the mean ratings of Lecturers with B.Sc. and Instructors with HND on the technical skills of teachers required for teaching of QBasic programming language.

7. There was no significant difference between the mean ratings of Lecturers and Instructors on the teaching methodology required for enhancing the teaching and learning QBasic programming language.

8. There was no significant difference between the mean ratings of Lecturers and Instructors on the strategies for utilization of instructional facilities for enhancing the teaching and learning of QBasic programming language.

**Implications of the Study**

The findings of this study have implications for the ministries (Education, Labour and Productivity, and Information and Communication Technology), colleges of education, curriculum planners, lecturers, Instructors and computer science students and the society at large.

The study has provided information on the strategies that will improve the teaching and learning of QBasic programming language in colleges of education. The study implies that the strategies identified will improve the performance of students in QBasic programming course
The findings have implications for colleges of education administrators. QBasic programming is a technical and skill oriented programme, thus requires adequate instructional facilities as well as qualified personnel for transfer of knowledge. It is therefore imperative that these school administrators should provide adequate training facilities and create enabling environment for serious academic work to thrive. Computer educators should be encouraged by making sure that they earn commensurate salaries comparable to their counterparts in the private sectors. The findings implied that college administrators should be dynamic in their administrative practices and initiate support and encourage computer educators by retraining them in emerging areas of Information and communication technology.

The findings of this study have some implications to computer educators (Lecturers and Instructors). Methods of instructional delivery are activities that will enhance the understanding of students in QBasic programming language. The findings implied that teachers should expand their research activities on modern instructional techniques that are capable of aiding teaching and learning of knowledge and skills. This implies that teachers should keep abreast with recent developments in information and communication technology in order to equip the students with requisite skills. They should make personal effort in acquiring new skills and knowledge that their career demands rather than waiting on the government and their employers for their retraining needs.

Conclusions

The study concludes that there are some strategies that could be adopted in enhancing the teaching and learning of QBasic programming language in colleges of education in Enugu, Anambra and Ebonyi States in the south east zone of Nigeria. The study is also of the view that
QBasic programming language should provide sound basis on which other programming language should be relied upon thus enabling students; develop appropriate programming skills needed in the workplace.

Evidence from the study also revealed that technical skills strategies of teachers could be used in improving the teaching of QBasic programming language. It is also found that teaching methodology strategies as well as the utilization of instructional facilities strategies could be used to enhance the teaching and learning of QBasic programming language. It is evident in the study that when computer students of colleges of education are adequately trained under an improved learning environment there performance in programming course are bound to improve considerably, as such they are bound to develop a remarkable interest in computer related courses thus enabling them to acquire the necessary programming skills required to secure and excel in the workplace.

**Recommendations**

The following recommendations were made based on the findings of the study and the implications:

1. School administrators should ensure that computer educators are sponsored on retraining programmes at least once every year by organizing workshops, seminars and conferences to enable them learn the modern technological skills in their chosen career.

2. In order to make the art of programming responsive to the needs of the society, computer students should be made to spend at least six months in computer and ICT programming firms during their industrial work experience to enable them acquire the relevant skills required to excel in the world of work.
3. School administrators should ensure that adequate instructional facilities are provided to every computer laboratory to enable students learn and master sound programming skills.

4. Parents, host communities and other major stakeholders of our education should assist our higher institutions by donating instructional facilities like laptops, desktop multimedia, generating plants, Internets etc to enable students practice and acquire the skills of programming languages.

**Suggestions for further studies**

1. The study can be replicated in other geopolitical zones of our country to identify the strategies to enhance the teaching and learning of QBasic programming language in colleges of education.

2. A similar study should be carried out to identify the strategies for improving the teaching and learning of programming languages in the Nigerian universities.

3. A study should be carried out to determine the training need of computer programming students in our higher institutions.
REFERENCES


Bezanson, W.R. (1975), Teaching structured programming in Fortran with IFTRAN. SIGCSE 7(1), 196 - 199.


APPENDIX 1.

Table 1: Performance of first-year students in QBASIC Programming courses according to their colleges of Education between 2003-2007 academic sessions

<table>
<thead>
<tr>
<th>Name of school</th>
<th>Session</th>
<th>Performance</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No. Pass</td>
<td>No. Fail</td>
</tr>
<tr>
<td>1 FCE EHA AMUFU ENUGU STATE.</td>
<td>2003/2004</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>2004/2005</td>
<td>28</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>2005/2006</td>
<td>67</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>2006/2007</td>
<td>54</td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>Total (%)</td>
<td>170 (48%)</td>
<td>183 (52%)</td>
</tr>
<tr>
<td>2 College of Education IKWO EBONYI STATE.</td>
<td>2003/2004</td>
<td>20</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>2004/2005</td>
<td>25</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>2005/2006</td>
<td>32</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>2006/2007</td>
<td>24</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>Total (%)</td>
<td>101 (40%)</td>
<td>151 (60%)</td>
</tr>
<tr>
<td>3 College of Education NSUGBE ANABRA STATE.</td>
<td>2003/2004</td>
<td>30</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>2004/2005</td>
<td>38</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>2005/2006</td>
<td>36</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>2006/2007</td>
<td>51</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Total (%)</td>
<td>155 (51%)</td>
<td>150 (49%)</td>
</tr>
<tr>
<td>4 College of Education (Tech.) UMUNZE ANAMBRA STATE</td>
<td>2003/2004</td>
<td>27</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>2004/2005</td>
<td>36</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>2005/2006</td>
<td>38</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>2006/2007</td>
<td>40</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>Total (%)</td>
<td>141 (44%)</td>
<td>177 (56%)</td>
</tr>
<tr>
<td>5 College of Education (Tech.) ENUGU State.</td>
<td>2003/2004</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>2004/2005</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>2005/2006</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>2006/2007</td>
<td>42</td>
<td>61</td>
</tr>
<tr>
<td></td>
<td>Total (%)</td>
<td>41 (%)</td>
<td>59 (%)</td>
</tr>
<tr>
<td>Overall Entry</td>
<td>609 (46%)</td>
<td>722 (54%)</td>
<td>1331</td>
</tr>
</tbody>
</table>

Source: The data above were retrieved from the schools as indicated.
APPENDIX II

LETTER TO THE RESPONDENTS

Department of Vocational Teacher Education,
(Computer Education Unit),
University of Nigeria Nsukka,
9th July 2008.

Dear Respondent,

QUESTIONNAIRE ON: Strategies for Enhancing the Teaching and Learning of QBASIC Programming language in Colleges of Education in Nigeria

I am a post graduate student of the above named institution, conducting a research work on the stated title. The study is being carried out with the aim of identifying: ways of improving technical skills of Lecturers required for teaching of QBASIC Programming, ways of improving the teaching methodology for the teaching and learning of QBasic programming, ways of improving the utilization of Instructional facilities for the teaching and learning of QBasic programming, ways of improving the learning environment needed for teaching and learning of QBasic programming and ways of enhancing administration needs for the teaching and learning of QBasic programming language in Colleges of Education.

Yours anonymity and confidentiality will be strictly preserved. This all information supplied by you will strictly be used for academic purpose.

Thanks for your anticipated cooperation.

Yours faithfully,

Olelewe, C.J
APPENDIX 111

Department of Vocational Teacher Education,
(Computer Education Unit),
University of Nigeria Nsukka,
9th July 2008.

Dear Sir/Madam,

REQUEST FOR VALIDATION OF RESEARCH INSTRUMENT

I am a post graduate student in the Department of Vocational Teacher Education (Computer Education), University of Nigeria Nsukka, currently undertaking a research project on the strategies for improving the teaching and learning of QBASIC programming language in Colleges of Education in old Anambra State.

Attached here-with is a drafted copy of the questionnaire. I sincerely request for your assistance in validating the questionnaire items for the study. Your comments will help to improve the final instrument.

Thanks for your co-operation.

Yours faithfully,

Olelewe, Chijioke Jonathan
PG/M.ED/05/40314
APPENDIX IV

QUESTIONNAIRE

STRATEGIES FOR IMPROVING THE TEACHING AND LEARNING OF QBASIC PROGRAMMING IN COLLEGES OF EDUCATION IN ANAMBRA, EBONYI AND ENUGU STATES

SECTION A

PERSONAL DATA

Read the following statements carefully and write down your responses in the blanks spaces provided. Where there are alternatives put a check ( ) against the response that is best applicable to you.

1. Teaching Staff
   (i) Instructor…………………………………………
   (ii) Lecturer…………………………………………

2. Qualification
   (i) Ph. D. / M. Ed. / M.Sc…………………………
   (ii) H.N.D/ O.N.D…………………………………

3. How long have you been working?
   (i) 0 – 10 years……………………………………..
   (ii) Over 10 years ………………………………..

Instruction: Please indicate by a check ( ) the degree to which you agree or disagree with each of the items.

Note:  
Strongly Agree  SA  (5) 
Agree A (4) 
Undecided UD (3) 
Disagree D (2) 
Strongly Disagree SD (1)
SECTION 1:

The strategies for improving the technical skills of the teachers for the teaching of QB in Colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Providing ample opportunities for in service training for computer educators.</td>
</tr>
<tr>
<td>2</td>
<td>Certifying computer educators in troubleshooting in computer maintenance skills.</td>
</tr>
<tr>
<td>3</td>
<td>Making web design and networking skills a prerequisite for all computer educators.</td>
</tr>
<tr>
<td>4</td>
<td>Making debugging skill essential in teaching computer programming.</td>
</tr>
<tr>
<td>5</td>
<td>Making keyboarding skills a prerequisite for computer educators.</td>
</tr>
<tr>
<td>6</td>
<td>Making identification of program bug task, such as syntax and semantic error mandatory for computer educators.</td>
</tr>
<tr>
<td>7</td>
<td>Making participation in at least three workshops, conferences and seminars compulsory for computer educators.</td>
</tr>
<tr>
<td>8</td>
<td>Making computer educators professionally certified.</td>
</tr>
</tbody>
</table>
SECTION 2

The strategies for improving the teaching methodology required for the teaching and learning of QB in Colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>SA</th>
<th>A</th>
<th>UD</th>
<th>D</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Making teaching of good programming principles a topmost priority.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Providing students with proved program and documentations on how to use them.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Teaching basic steps on how to load or invoke QB compilers.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Teaching with use of canned programs that students can compare and modify.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Teaching first the methodologies of problem-solving before the translation of algorithm into a well-structured program.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Teaching algorithm development before implementation of the program code.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Making top-down step-wise refinement approach compulsory in program development.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Making automated teaching tools essential requirement for teaching programming courses.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Making the use of Automated learning tools to enhance reliability of the programming.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Identification of summary of students problems in programming lessons</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Making use of individual and group projects in teaching QB programming.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Using collaborative participation in on-line tutorial lessons in programming.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Giving assignments after every QB programming lessons</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Making use of Continuous Assessment Test (CAT) as appropriate means of evaluation of students.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Making assessment of computer practical 60% and theory 40%.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SECTION 3: The strategies for improving the utilization of instructional facilities for the teaching and learning of QB programming in Colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>Making students have free access to Internet</td>
</tr>
<tr>
<td>25</td>
<td>Provision of standard library for every computer department by donor agencies.</td>
</tr>
<tr>
<td>26</td>
<td>Use of computer for practical in ratios of one computer is to one student</td>
</tr>
<tr>
<td>27</td>
<td>Making host communities contribute in donating computers and other teaching aids to computer department.</td>
</tr>
<tr>
<td>28</td>
<td>Providing every computer educator with desktop and/laptop for classroom demonstration.</td>
</tr>
<tr>
<td>29</td>
<td>Provision of VSAT for internet connectivity for every computer department.</td>
</tr>
<tr>
<td>30</td>
<td>Making the conduct of practical a sole function of computer Instructors.</td>
</tr>
<tr>
<td>31</td>
<td>Provision of standby generators as a criterion for accreditation of every computer department.</td>
</tr>
<tr>
<td>32</td>
<td>Making attendance to pratical class mandatory for computer students.</td>
</tr>
<tr>
<td>33</td>
<td>Providing every classroom with desktop computer for classroom demonstration by the teachers.</td>
</tr>
</tbody>
</table>
SECTION 4: The strategies for improving learning environment for the teaching and learning of QB programming in colleges of Education

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>SA</th>
<th>A</th>
<th>UD</th>
<th>D</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>Limiting each computer class stream to not more than 35 students.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>Using computers for practical in the ratio of one student is to one computer.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>Providing students with free access to computer laboratory in batches.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>Launching nation-wide program on computer oriented programmes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>Making computer students to participate in SIWES in computer related establishments only.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>Providing every computer department with standard library.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>Citing computer schools in metropolitan areas.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>Making access to electricity and basic infrastructures a pre-requisite for computer programmes.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>Provision of computer centers and cyber cafes within and around collages of education.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>Provision of access to virtual libraries by students and teacher.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**SECTION 5: The strategies for improving administrative needs for the teaching and learning of QB in Colleges of Education**

<table>
<thead>
<tr>
<th>S/N</th>
<th>ITEMS</th>
<th>SA</th>
<th>A</th>
<th>UD</th>
<th>D</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>Given admission only to prospective candidates who possess minimum entry requirement.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>Payment of enhance SIWES allowance to participating students of computer studies.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>Making school administrators collaborate with private sectors in provision of teaching aids.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>Making computer studies a top priority both at the national and state level.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>Using criterion reference test for evaluation of computer programme.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>Involving parents in funding and provision of instructional materials</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>Making Post UME/PCE screening test a criterion for admission into computer science program.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>Making school management sponsor computer educators in seminars and workshops.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>Payment of scarce skill allowance to computer Instructors.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>Making teaching conditions of service for computer educators commensurate with those of private sectors.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>Providing special fund for retraining of computer educators.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>Payment of special allowance to teachers for SIWES supervision.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>Prompt payment of salaries of computer educators.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>Awarding of scholarship to outstanding students in programming courses.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>Providing students who distinguished themselves in programming courses with automatic employment.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>