ZIMBABWE

MINISTRY OF PRIMARY AND SECONDARY EDUCATION

SOFTWARE ENGINEERING SYLLABUS

FORMS 5 - 6

2015 - 2022

Curriculum Development and Technical Services
P. O. Box MP 133
Mount Pleasant
Harare

ACKNOWLEDGEMENTS

The Ministry of Primary and Secondary Education wishes to acknowledge the following for their valued contributions in the development of this syllabus:

- The National Software Engineering Panel comprising representatives from:
  - Computer Studies Secondary Teachers
  - Zimbabwe School Examinations Council (ZIMSEC)
  - Teachers' and Technical Colleges
  - Universities
  - Computer Society of Zimbabwe (CSZ)
- United Nations Children's Fund (UNICEF)
- United Nations Educational, Scientific and Cultural Organisation (UNESCO)

CONTENTS

1.0 PREAMBLE

1.1 Introduction

Software engineering is the branch of computing that creates practical and cost effective solutions to real life problems by applying scientific knowledge. This syllabus is designed to cover the two years of advanced secondary education. It equips learners with software development skills and prepares them for self-sustenance, professional development and lifelong learning.

1.2 Rationale

Software plays a central and underpinning role in aspects of daily life such as communications, governance, manufacturing, banking and finance, education, transportation, entertainment, medicine, agriculture and law.

This course equips the learner with fundamentals of software engineering including understanding system requirements, effective methods of design, coding, testing, teamwork and the application of software engineering tools. The course provides an opportunity for the learner to identify socio-economic problems in a real world setting and develop practical and sustainable solutions.

1.3 Summary of Content

The Software Engineering Syllabus covers software development processes, securing computer systems, adherence to professional and ethical standards and business initiative.

1.4 Assumptions

It is assumed that learners:

- have passed Computer Science and Mathematics at Form 4 level or equivalent
- have acquired skills in system development
- are conversant with at least one programming language

1.5 Cross-Cutting Themes

The teaching and learning of Software Engineering should integrate the following cross cutting themes:

Cross Cutting Themes	Examples
Life skills	Computer based systems
Enterprise skills and financial literacy	Project management
Collaboration	Project teams
Gender equality	Encouraging equal opportunities in the use of ICTs
Environmental issues	Feasibility study during system development
Disaster Risk Management	System security

2.0 PRESENTATION OF SYLLABUS

The Software Engineering syllabus is presented as one document for Forms 5 and 6.

3.0 AIMS

The syllabus aims to enable learners to:

3.1 appreciate the benefit of software design in solving everyday problems
3.2 develop software systems
3.3 appreciate the ever evolving nature of computer systems
3.4 be confident and creative in the development of software projects
3.5 adhere to ethical practices when developing software

4.0 SYLLABUS OBJECTIVES

Learners should be able to:

- identify software development needs and challenges that require various solutions
- formulate solutions to meet desired needs based on user requirements
- justify the need for ethical, legal, security and

social responsibilities in software development
- analyze the impact of computer based solutions on individuals, organizations, and society
- develop software projects using modern software engineering techniques and tools
- apply entrepreneurial skills to commercialize software products
- apply the concepts of system development processes in problem solving
- demonstrate the ability of teamwork during software development
- use knowledge from other disciplines in the development of computer applications
- demonstrate an awareness of the evolving nature of computer systems

5.0 METHODOLOGY AND TIME ALLOCATION

5.1 Methodology

The teaching and learning of Software Engineering is based on a learner-centred approach. The following methods are recommended:

- Problem solving
- e-Learning
- Multi-media
- Simulation and modeling
- Discovery
- Experimentation
- Project-based learning
- Question and answer
- Demonstrations
- Team teaching
- Discussion
- Educational Tours
- Research and Presentations
- Expert guest presentations

5.2 Time Allocation

This learning area should be allocated at least 12 periods of 35 - 40 minutes per week, that is, 4 theory periods and 8 practical periods. Learners should be engaged in each of the following at least once a year: Educational Tour, Exhibition, Seminar and a week of practical orientation.

6.0 TOPICS

The syllabus consists of the following topics:

6.1 Software Process Models
6.2 Software Project Management
6.3 Software Design
6.4 Data Structures and Algorithms
6.5 Programming
6.6 Security and Ethics
6.7 Quality Assurance and Testing
6.8 Enterprise in Software Engineering

7.0 SCOPE AND SEQUENCE CHART

TOPIC	FORM 5	FORM 6
7.1 Software Process Models	• Software Engineering Application Areas • Process Models	
7.2 Software Project Management	• Characteristics of software projects • Project management skills • Software Crisis/Failure • Project Planning • Project Scheduling	
7.3 Software Design	• Software Design strategies • Interface design	• Architectural design • Detailed design
7.4 Data Structures and Algorithms	• Pseudo-code structures • Sorting and searching algorithms • Dynamic and static data structures	
7.5 Programming	• Web based programming • Database systems	• Object Oriented Programming • Advanced Programming
7.6 Security and Ethics	• Common threats and software vulnerabilities • System Security • Risk management techniques • Ethics • Security policy • Cyber crime	
7.7 Quality Assurance and Testing	• Testing Approaches • Testing Levels	• Software Quality Attributes • Software Quality Assurance • Safety and security • ISO standards

7.8 Enterprise in Software Engineering		• Careers in Software Engineering • Business Viability • Marketing Strategies • Intellectual Property Rights

8.0 COMPETENCY MATRIX

FORM 5

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE, SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
8.1 Software Process Models	• identify opportunities for software development • outline the stages of each process model • explain the importance of documentation in software development • develop a project using process models	• application areas for system development such as education, government, banking and mining • Process Models: - SDLC generic model - RAD - Prototyping - Object Oriented • Project Documentation	• Selecting possible application areas as groups • Discussing the stages and activities of each process model • Producing the documentation for their group project • Applying software process models to a project	• Software Development Forums • National ICT Policy • CASE tools • Word processing software • Innovation hub Centres • Software development companies
8.2 Software Project Management	• explain characteristics software projects • identify project management skills • explain common causes of software project failures	• Characteristics of software projects • Project management skills • Software Crisis/Failure	• Describing characteristics of software projects • Discussing project management skills • Applying project management skills in software development • Discussing the issues relating to software crisis • Investigating the causes of software project failure in a given scenario	• Software Crisis articles • Y2K problem Case Study • Internet

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE, SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
	• develop a realistic project plan • evaluate a project plan • identify the relevant skills for a project team	• Project Planning - Software Development Plan - Quality Assurance Plan - Validation Plan - Configuration Management Plan - Maintenance Plan - Staff Development Plan	• Constructing a project plan • Analysing project proposals • Determining the skill sets of project team members	• Project Plan Guideline • PMBOK Guide • Software Extension for Project Management Guide
	• determine the time resource needed for project completion • develop project schedules using appropriate tool(s)	• Project Scheduling • Tools for Schedule Development - Critical Path Method (CPM) - GANTT Chart	• Calculating the amount of time required for project completion • Designing a project schedule using tools :CPM and GANTT charts • Designing GANTT charts	• Project management software • Statistical tools
8.3 Software Design	• outline software design strategies • apply appropriate user interface design principles used in Software Development	• Software Design strategies - Structured design - Function oriented design - Object oriented design • Interface design - User interface - Web based design	• Discussing software design strategies • Designing user interface for their group projects	• Internet • CASE tools • Web design tools e.g HTML, Joomla, Wordpress • Design tools such as MicroSoft Visio, Adobe Photoshop, CorelDraw
8.4 Data Structures and Algorithms	• use pseudocode structures in algorithm design • apply sorting and searching algorithms in problem solving • outline static and dynamic data structures	• Pseudo-code structures • Sorting algorithms - Bubble - Quick • Searching algorithms - Linear - Binary • Dynamic and static data structures - Trees - Arrays	• Formulating trace tables for a given algorithm • Using sorting and searching algorithms to solve problems • Discussing the concept of data structures • Using operations on data structures	• Multi- Media Tutorials • Open source packages

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE, SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
	• perform operations on binary trees and arrays			
8.5 Programming	• develop a web based application • distinguish database systems • develop a database for a web based application • normalize database tables up to 2nd normal form	• Web Application Development - client side development - server side development • Database Systems - File based database systems - Relational database systems - Database Management Systems - Normalization (2nd normal form)	• Creating a web application following appropriate design principles • Creating a database • Comparing and contrasting database systems • Integrating database systems to a web application • Normalizing database tables up to 2nd normal form	• Client side programming languages such as Java Script • Server side programming tools such as PHP, VB.Net • Database packages such as Microsoft Access, SQL and MySQL
8.6 Security and ethics	• analyze common threats and vulnerabilities of software systems • examine user level security measures • identify sources of vulnerability arising at the programming level • explain tools used to eliminate vulnerabilities at programming level	• Common threats and software vulnerabilities such as: - Malware - Botnets - Phishing • System Security - User level security measures o Antispyware/ antivirus o Firewalls - Programming and Security	• Evaluating major counter measures to software and system attacks • Applying security techniques in designed solutions • Conducting case studies on different attack scenarios • listing tools used to eliminate vulnerabilities at programming level • applying security features found in programming languages	• Antivirus software • Firewalls • Anti-spyware • Video Clips • Library functions • Programming packages such as Visual Basic, Java and C++

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE, SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
	• explore techniques and practices of risk management	• Risk management techniques	• Conducting case studies on Risk analysis and management • Formulating a disaster recovery plan in case of system failure: • system back-ups • System restore	• Tools such as Logic Manager
	• identify code of ethics and professional practices in Software Engineering (SE) • demonstrate ethical practices in SE • identify relevant legislative and regulatory frameworks in systems security • analyse various types of Cybercrime	• Ethics • People and security • Data protection legislation • Security Policies • Cybercrime	• Applying code of ethics and professional practices in SE • Discussing ethical issues in SE • Describing relevant legislative and regulatory frameworks in SE • Analyzing the effects of cybercrime from case studies	• Data Protection Act • Second Science Technology and Innovation Policy of Zimbabwe (2012) • Print and Electronic Media • Case Studies on Data Protection and Legislation issues • National ICT Policy
8.7 Quality Assurance and Testing	• use different testing approaches in software development • outline various testing levels in system development	• Testing Approaches - Black box testing - White box testing • Testing Levels - Unit testing - Module testing - System testing - Acceptance testing	• Applying testing approaches to all the levels of the group project • Validating software through the various testing levels • Demonstrating the use of automated testing tools	• Automated testing tools such as HTML unit, Selenium • CASE Tools • Multimedia tutorials • Expert Guest

FORM 6

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE,SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
8.8 Software Design	• explain the architectural design process and architectural models	• Architectural design - Organisation - Control modelling - Modular decomposition	• Discussing the design process for given systems such as embedded, database, banking, mining, education and web based	• Multimedia tutorials • CASE tools • Design Tools • Such as Microsoft Visio, Corel Draw, Adobe Photoshop • Expert Guest
	• outline the importance of architectural styles in Software Design	• Architectural models - Static - Dynamic - Interface • Architectural styles - Client server - Service oriented - Layered	• Discussing the importance of architectural styles in software design • Researching on architectural styles	
	• demonstrate how detailed design is used to further decompose analysis and design models	• Detailed Design - Analysis models - Design models	• Applying detailed design in software development • Conducting educational tours to IT companies and manufacturing industries	

TOPIC	OBJECTIVES Learners should be able to:	CONTENT (KNOWLEDGE, SKILLS, ATTITUDES)	SUGGESTED LEARNING ACTIVITIES AND NOTES	SUGGESTED LEARNING RESOURCES
8.9 Programming	• outline features of Object Oriented Programming (OOP) • design games and mobile applications	• Object Oriented Programming - Classes - Encapsulation - Polymorphism - Inheritance • Advanced Programming - Mobile applications - Gamming and Animations	• Implementing classes • Demonstrating the use of Singleton and Model View Controller(MVC)design patterns • Designing mobile device applications • Designing games with indigenous orientation	• Object Oriented Programming tools such as Java, Python, VB.NET, PHP • Animation software such as Scratch, Adobe Creative Suite
8.10 Quality Assurance and Testing	• explain software quality attributes and their importance in quality assurance process • demonstrate the use of quality assurance tools and techniques in quality management • outline the importance of safety and security management in quality assurance • recognize appropriate ISO Standards	• Software Quality Attributes • Software Quality Assurance - Quality policy - Tools and techniques o Inspection o Audit trials • Safety and Security Management • ISO Standards (ISO 9000)	• Discussing the importance of quality assurance attributes in software quality management • Discussing the importance of software tools and techniques • Applying quality assurance tools and techniques in their software projects • Discussing the importance of ISO standards in software development • Applying ISO Standards relevant to their project areas • Conducting case studies related to quality assurance and testing in software management	• Field trips • ISO Standards Documents

8.11 Enterprise in Software Engineering	• identify careers in SE • describe the duties done by SE professionals • evaluate cost effective solutions in SE • outline ways of marketing software • evaluate marketing strategies • recognize intellectual property rights	• Careers in SE • Business viability: - Benefits realization - Cost analysis - Trade off analysis • Return on investment • Marketing strategies • Intellectual Property Rights	• Identifying careers in SE • Conducting educational tours to software development companies • Participating in IT expos/fairs/events • Discussing on cost effective solutions in SE • Researching on viability of software development project • Designing marketing strategies • Discussing intellectual property issues	• Expert guests presentation • Print and Electronic Media • Internet • Business journals • Statistical tools • Print and Electronic Media • Expert Guest

9.0 ASSESSMENT

In order to have a holistic assessment of the learner from form 5 to 6, learners will be assessed in the aspects of continuous and summative assessment with each contributing to the learner's final grade.

9.1 Assessment Objectives

Learners will be assessed in the following areas:

(i) Knowledge and Understanding

Learners should be able to:

- describe the use of software engineering in a range of information processing systems
- explain the systematic development of solutions to problems and the appropriate techniques for implementing such solutions
- describe and explain the need for the use of SDLC in software development

(ii) Problem Solving

Learners should be able to:

- analyze the situation and identify the parts which are appropriate for a computer based solution
- design, implement and document an effective solution using appropriate software and programming languages
- implement the system development life cycle to produce effective and documented and tested systems

(iii) Communication Skills

Learners should be able to:

- develop an understanding of the component parts of computer systems and how they inter-relate including software, data, hardware, communications and people
- interpret and organize information
- recognize and present information in a variety of forms
- use ICT tools to disseminate information about applications of computers, problems and their solutions

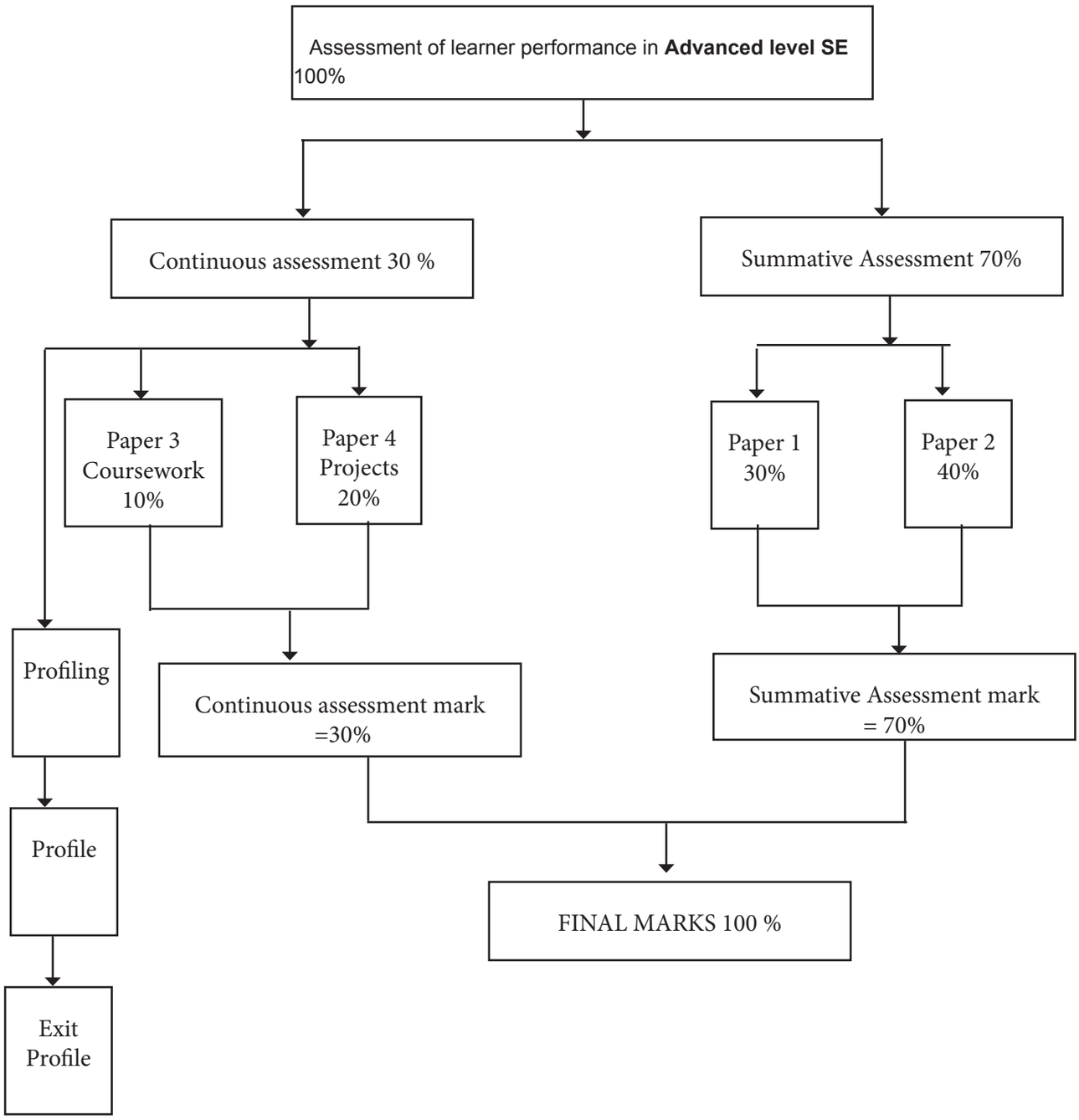
(iv) Practical Skills

Learners should be able to:

- design and develop a solution to a problem using any programming language
- demonstrate proficiency in the creation, design and implementation of computer solutions using prescribed programming packages.
- conduct research using the internet

9. 2 Scheme of Assessment

Software Engineering learning areas will be examined through continuous and summative assessment as follows.



The Scheme of Assessment is intended to encourage positive achievement by all learners.
The subject will be examined in 4 papers as shown in the table below.

Paper	Form of Assessment	Type of Paper	Duration	Total marks	Weighting
1	Summative Assessment	Free Response	3 Hours	100	30
2		Practical	3 Hours	100	40
3	Continuous assessment	Coursework	5 terms	100	10
4		Project	4 terms	100	20
TOTAL				**400**	**100%**

9.3 Specification Grid

(i) Content distribution

PAPER 1

TOPIC	WEIGHTING (30%)
Software Process Models	20
Software Project Management	15
Software Design	15
Data Structures and Algorithms	10
System security and Ethics	15
Quality Assurance and Testing	15
Enterprising in Software Engineering	10
TOTAL	**100**

PAPER 2

SECTION	WEIGHTING (40%)
Software Design	15
Data Structures and Algorithms	10
Programming	60
System Security and Ethics	5
Quality Assurance and Testing	10
TOTAL	**100**

(ii) Skills distribution

All internal and external tests on cognitive skills (theory) shall be 40% knowledge and understanding 50% problem solving and 10% Practical Skills.
All internal and external practical tests shall be 100% practical skills. Thus the weighting of questions based on the skills for the subject, Software Engineering, will be as follows:

SKILL	Paper 1(%)	PAPER 2(%)
Knowledge and Understanding	40	10
Problem Solving	50	30
Practical Skills	10	60
TOTAL	**100**	**100**

9.4 paper Descriptions

Paper 1: Theory (100 Marks)
The paper consists of 10 to 12 compulsory questions.

Paper 2: Practical (100 Marks)
The paper consists of 7 practical questions each worth 20 marks and the candidate is to choose 5.

Paper 3: Coursework (100 marks)
Coursework is made up of 7 items which comprises of five practical assignments and two theory tests. Practical assignments and tests are set, marked and recorded internally by teachers. Each of these assignments and tests should match the skills distribution table given above. The internal practical assignments are spaced equitably from the beginning of term one in Form 5 up to the end of term two in Form 6. The internal theory coursework tests should be written one in Form 5 and one in Form 6 however the tests should be within the range term two in Form 5 and end of term two in Form 6. All marked practical and theory scripts including the compiled mark schedule will be submitted to ZIMSEC.

Paper 4: Project (100 marks)
Examination Centres are advised to encourage their candidates to identify real life problems within their environment and develop solutions in line with the project guide lines. The project will require candidates to have practical programming experience including writing their own programs, executing (running), testing and debugging them. Knowledge of programming language syntax will be examined in the project report. The higher ability candidates are to be encouraged to extend their practical programming beyond the scope of these tasks. The project work should be carried out from term two in Form 5 to end of term two in Form 6.The marked project reports including the compiled mark schedule will be submitted to ZIMSEC.

10.0 GLOSSARY/APPENDICES

APPENDIX I: GLOSSARY OF TERMS USED IN QUESTION PAPERS

It is hoped that the glossary will be helpful to learners as a guide. The glossary has been deliberately kept brief not only with respect to the number of terms included but also to the descriptions of their meanings. Learners should appreciate that the meaning of a term must depend in part on its context.

1	**Define**	is intended literally for only a formal statement or equivalent paraphrases being required.
2	**State**	implies a concise answer with little or no supporting argument e.g. numerical answer that can readily be obtained by inspection.
3	**List**	requires a number of points generally each of one word with no elaboration, where a number of points is specified this should not be exceeded.
4	**Explain**	implies reasoning or some reference to theory depending on the context.
5	**Describe**	expected to state in words (using diagrams where appropriate) the main points of the concept.
6	**Outline**	implies brevity that is restricting the answer to given essentials.
7	**Predict/deduce**	required to produce the expected answer by making a logical connection betwee other pieces of information.

8	**Suggest**	it is used in two main contexts that is either to imply that there is no unique answer or to imply that learners are expected to apply their general knowledge.
9	**Find**	is a general term that may alternatively be interpreted as calculate, measure, determine etc.
10	**Determine**	often implies that the quantity concerned cannot be measured directly but is obtained by calculation.

APPENDIX II: ACRONYMS

SDLC	System Development Life Cycle
SE	Software Engineering
VB	Visual Basic
PHP	Hypertext PreProcessor
HTML	Hypertext Markup Language
MySQL	Structured Query Language
OOP	Object Oriented Programming
ICT	Information and Communication Technology
Internet	International Network
ISO	International Standard Organisation
Y2K	Millennium bug Year 2000
CASE	Computer Aided Software Engineering
CPM	Critical Path Method
PMBOK	Project Management Body of knowledge
RAD	Rapid Application Development
DFD	Data Flow Diagram
MVC	Model View Controller

APPENDIX III: PROJECT GUIDE

The project must not exceed 60 pages excluding appendices. The project must include the following layout:

1. Cover Page
2. Table of Contents
3. Project Content (Sections)
4. Appendices

The Appendices include any two of the following

- Sample of completed questionnaires
- Sample of interview questions with respondent answers
- Sample documents

NB the project must be spiral bound.

SECTION A (25 marks)

Selection, Investigation and Analysis

- Define a problem
  - Choice of problem area and background analysis.[3]
- Investigation of the current system
  - Data analysis using DFDs, flow charts and ERDs
  - Research instruments e.g. questionnaire, record inspection, interviews and observation.
  - Identify problems with the current system.[5]
- Feasibility study[5]

- Requirements specification
  - User
  - Software
  - Hardware.[4]
- Aims and objectives. [5]
- Evidence that the research has been carried out.[3]
  - Examples are filled in questionnaires, interviews with respondent answers, sample documents and write up on observation.

SECTION B (25 marks)

Design

- Consideration of alternative method.[3]
  - Justification of proposed solution [2]
- Input design
  - Appropriate data capture forms and screen layouts[4]
- Data Structures/File design[5]

OR

- Object Oriented Design
  - Class diagrams
  - Use Case diagrams
  - Sequence diagrams
- Overall Plan [3]
- Output Design
  - Specification and design of the required output
  - Interface design (on screen commands)[5]
- Test Strategy/Test Plan
  - Select test plan and justify[3]

SECTION C (25 marks)

Software Development

- Techniques that improve the structure, appearance and clarity of the code that is:
  - Procedures
  - Functions
  - Scope of variables (local and global)
  - Use of comments
  - Blank lines
  - Indentation[10]
- Technical documentation
  - Algorithms
    - pseudo codes
    - flowcharts for modules
  - codes/program listing [7]

- User documentation
  - Installation
  - Starting the systems
  - Navigation of the system
  - Exiting the system [8]

SECTION D (15 marks)

Testing and Evaluation

- User Testing
  - Design and select test data
  - Test for standard, extreme and abnormal/invalid data
  - Evidence of testing to be shown through sample runs and error messages [5]

- System Testing
  - Ease of use
    - clarity of instruction to the user
  - Reliability
    - produce reliable results, there should be no bugs
  - Effectiveness
    - The system should work efficiently
- Produce results with minimum del[5]
- Evaluation Limitations of the system
  - Extent of success in meeting the system objectives as stated in the system requirement specification
  - Achievements
  - Limitations
  - Evaluate results against the system objectives – achievements and limitations
  - Opportunities for future development [5]

SECTION E (10 marks)

General Expectations

- Depth of Knowledge and Understanding
  - Reflects the degree of computing in the project
  - Is the code fairly standard?
  - Different techniques implemented[2]

- Degree of Originality
  - Imagination and innovation
  - Has an attempt been made to do something different/unique? [2]

- Overall conduct of the project
  - Is the work carefully organized? The degree of help to be reflected[1]
- Quality of the completed report
  - Written report should be easy to follow
  - Defined sections, page numbers and an index. [5]

APPENDIX IV: RESOURCES AND EQUIPMENT

Infrastructure and Equipment

For a school to run the Software Engineering Syllabus for examination purposes, the under listed infrastructure and equipment need to be in place

Computer Laboratory

Personal Computers to accommodate 1 learner per computer
NB it is encouraged that the learner be in possession of Laptop or tablet.

A printer
Air conditioned laboratory
Open source and /or Licensed software
Computer Desks and Chairs to accommodate the number of students
Dustless Displays for the Teacher (securely-mounted Whiteboard, LCD projector)
Computer Repair Toolkit
Internet connectivity
Alternative power source such as generator, solar, ups

Theory Classroom

Classroom furniture to accommodate the learners
Writing Surface for the Teacher (e.g. securely-mounted Whiteboard, LCD projector)

In both the above cases, there should be adequate lighting and ventilation.