REPUBLIC OF CAMEROON

PEACE – WORK – FATHERLAND

****

MINISTRY OF SECONDARY EDUCATION

****

INSPECTORATE GENERAL OF EDUCATION

****

INSPECTORATE OF PEDAGOGY IN CHARGE OF THE TEACHING OF COMPUTER SCIENCE

****

Teacher's Guide

to the

High School Computer Science Syllabus

WORKING DRAFT August 2020

TABLE OF CONTENT

1 INTRODUCTION

The High School Teaching Syllabus on Computer Science is designed to equip learners with necessary skills and experience to apply algorithmic and computational thinking practices to solve problems using computer systems. With a unique focus on creative problem-solving techniques, and representation and ethical use of computers, the course offers to learners a broad range of competencies, individual skills and motivation, which are essential for a successful working life.

This Advanced Level Computer Science syllabus is expected to provide a learner with the opportunity to obtain Cameroon national qualifications in Computer Science and pursue university studies or employment.

Learners acquire knowledge and understanding of the academic aspects of Computer Science through theory lessons, laboratory practices and collaborative assignments. The entire syllabus has been designed following the Competence-Based Approach recommended by the Ministry of Secondary Education with specifications from the Inspectorate General of Education and guided by the Inspectorate of Pedagogy in charge of the teaching of Computer Science. This is a task-oriented methodology to give the learners ample user-time practices with the computer to help them bring out feasible solutions to real life situations or problems. Many hands-on tasks have been suggested to enable the learner to practice in a computer laboratory. It is hoped that teachers will follow these suggestions and lead their learners to explore and discover more in a computing environment.

Commented [1]: Good to have a roadmap here, to tell us what to expect from each section of the document! For example, though needed, the section "Teaching Method" was a surprise, especially as syllabuses are more about body-of-knowledge than teaching!

2 ORGANIZATION OF THE SYLLABUS

2.1 COURSE DESCRIPTION AND DURATION

This Computer Science (CS) teaching syllabus for High Schools is a two-year rigorous, University entry-level preparatory curriculum that builds Advanced Level learners with broad foundations in the field of Computer Science. It covers a wide range of fundamental topics including Data Structures and Algorithm Design, Programming, Information Systems, Computer Organization and Networks, Digital Privacy and Security, and the Societal impacts of computing.

The total learning time for all the modules on this High School Computer Science teaching syllabus is 426 periods, with 224 periods for the Lower Sixth (First Year) course work and 192 periods for the Upper Sixth (Second Year) course work. A period here refers to the teaching time scheduled on the school timetable and it ranges from 50 to 60 minutes. The way that this time is spent will reflect the subject matter of each module. In this regard, the average total teaching time is eight periods per school-week, with six periods for theory and two periods for practical work. Also, the judicious

use of internet and laboratory facilities and overhead projectors would greatly enhance the quality and effectiveness of teaching periods.

The following Table summarizes the teaching load of High School Computer Science:

CLASS	WEEKLY TEACHING LOAD (PERIODS)	WEEKLY PRACTICAL LOAD (PERIODS)	ANNUAL TEACHING LOAD (PERIODS)	COEFFI-CIENTS
LOWER SIXTH (First Year)	04	04	224	05
UPPER SIXTH (Second Year)	04	04	192	05

2.2 PREREQUISITE FOR LEARNING HIGH SCHOOL COMPUTER SCIENCE

- Learners willing to study this high school Computer Science course require no specific or formal qualification in Computer Science but must have Ordinary Level Mathematics and should have an understanding of civic education and a disposition for skills acquisition and responsible social behaviour. Basic computer literacy skills are an advantage.
- Learners should have a mastery of English Language at least the Ordinary Level and demonstrate a working knowledge of it.

2.3 MODULES

This High School Computer Science syllabus is divided into 11 modules including a guideline for a mini project suggested for each of the modules. These mini projects enable the learner to carry out hands on activities to implement concepts of the modules. The modules are structured as follows:

CLASS	MODULES	DURATION
LOWER SIXTH (First Year)	**Module 1:** COMPUTER APPLICATIONS AND SOCIO-ECONOMIC IMPLICATIONS	**18**
	Module II: SOFTWARE	**50**
	Module III: INFORMATION SYSTEMS	**7**
	Module IV: DATA STRUCTURES AND ALGORITHMS	**34**
	Module V: PROGRAMMING	**40**
	Module VI: SOFTWARE DEVELOPMENT I	**10**
	Module VII: COMPUTER SCIENCE PROJECT	**18**
UPPER SIXTH (Second Year)	**Module VIII:** COMPUTER ORGANIZATION AND ARCHITECTURE	**29**
	Module IX: COMPUTER NETWORKS, DATA COMMUNICATIONS AND SECURITY	**50**
	Module X: DATABASE SYSTEMS	**28**
	Module XI: SOFTWARE DEVELOPMENT II	**50**

A successful navigation through the 11 modules by the teacher adequately prepares the learner to demonstrate competencies in solving most real-life situations using knowledge and skills gained from Computer Science. Thus, the first six (06) modules are taught in Lower Sixth (First Year) and the last four (04) in Upper Sixth (Second Year) while the Module on Computer Science Project is shared as it is dependent on the other modules.

In effect, the First Year modules provide some background on which the Computer Science Project module could depend on. The Computer Science Project module is biased towards developing Project development and realisation capabilities in the learner, while reinforcing the understanding and application of other module requirements. As such, project skills developed could be applied in subject domains taught in later modules, and the learner should be able to discover and exploit knowledge, skills,

competencies and experiences beyond the project course and even the Computer Science discipline.

2.4 MODULE PRESENTATION STRUCTURE

Each module is presented as a table having three main columns with nine sub-headings among them. This structure is intended to give teachers an orientation on how to exploit the module entries and subsequently prepare lessons. The terminology used for the column headings is contextual and non-standard and so the following definitions and clarifications are significant.

I. CONTEXTUAL FRAMEWORK: This gives a global picture of the life situations from which lesson inspirations are drawn. This column is further broken down into two sub-headings:

- ***Family of Real-life situations***: This term is an umbrella statement that groups related real life situations.
- ***Examples of Real-life situations:*** This column situates the lesson by bringing examples from life situations. This can be an activity within a task. It is expected that the teacher can use the types of examples listed to coin a life situation for a given lesson.

II. COMPETENCIES: Competencies refer to the ability to do something successfully and efficiently. It is understood here to be a process(es) evident in an action(s). This main column has two sub-headings:

- ***Categories of Actions:*** These categories group examples of related activities that learners are expected to carry out in the course of the lesson, as facilitated by the teacher. This may serve as topics or sub topics from where lessons are derived.
- ***Examples of Actions****:* These refer to the actions or activities the learners are expected to do successfully and efficiently during a lesson, with or without the teacher. They constitute indicators that certain specific abilities have been built into the learner in the course of the lesson. These examples of actions may also serve as lessons.

III. RESOURCES: Resources refer to the cognitive and material requirements that ensure a successful lesson. This column has five sub-headings:

- ***Core knowledge:*** These are expressions and concepts that learners should be able to define and comprehend in an effective lesson.
- ***Skills:*** These are indicators of competencies/skills that the learner should demonstrate in class/team during a lesson, or out of class as the context may demand.

- ***Attitudes:*** These are the behavioural responses the learner develops and demonstrates competence in during and after the lesson. It is inherent in the learner so that the learner attains performance requisite levels in the learning process.
- ***Other Resources****:* These are material resources to be used by the teacher or learner and are required to facilitate the teaching/learning process. They are mostly standard teaching aids black-/white- board, overhead projectors and computers.
- ***Duration****:* This is the expected duration, typically divided into periods, in which the associated material should be formally taught or learned.

3 OBJECTIVES ENVISAGED BY THE SYLLABUS

3.1 HIGH SCHOOL LEAVER PROFILE IN COMPUTER SCIENCE

Mindful that society is fast developing with rapid changes in knowledge, it is pivotal to study the fundamentals of how such knowledge is produced, acquired, used and propagated in this information age.

At the completion of this high school Computer Science syllabus, learners will have developed skills that let them build and exploit computer systems that address problems in their community. Specifically, inter alia, the learners use the knowledge and skills acquired to:

- Develop computer software;
- Design databases;
- Setup and configure data communication networks;
- Carry out a project successfully;
- Propose IT solutions to tackle real life issues;
- Develop websites;
- Assure the quality of developed software;
- Analyse computer systems and architectures.

3.2 THE AIM OF TEACHING COMPUTER SCIENCE IN HIGH SCHOOLS

This High School Computer Science teaching syllabus is intended to build in the learners:

- an understanding and proper use of basic but fundamental concepts in Computer Science;

- positive attitudes towards contemporary developments and evolution of computer systems;
- problem solving skills underpinned by algorithmic and computational thinking;
- confidence to analyse and apply theory and practice in real world situations; and
- foundation for higher studies or training.

4 METHODOLOGY

4.1 METHODS OF TEACHING COMPUTER SCIENCE

A teaching method describes the means used by the teacher to facilitate learning and to attain set pedagogic objectives. The fundamental teaching approach adopted in this syllabus is the Competency Based Approach (CBA) which, helpfully, allows the teacher to include diverse teaching and learning methods. The variant of CBA used here favors learner centered teaching and learning, with entries and/or exits through real life situations.

The following subsections discuss some of the pedagogic methods most adopted to teach Computer Science, and could be exploited within CBA.

4.1.1 PROBLEM BASED LEARNING

The Problem Based Learning is one where, given a problem, the learners play the role of active problem solvers while the teacher assumes the role of a facilitator.

The learners are presented with a real-life situation which could be an ill structured problem, they study the problem or research on it, organize their ideas, discuss their relative knowledge concerning the problem, and attempt to define its scope. In the course of their discussions, the learners ask questions concerning aspects of the problem which they do not understand. The learners are continually encouraged by the teacher to bring forth what they know and what they do not know. Problem-solving reckons with individual and group interactions, classification of problem-types and acquisition of their resource needs, and the integration of possibly researched solutions.

4.1.2 PROJECT BASED LEARNING

The High School Computer Science teaching syllabus exploits project-based learning to strongly engage learners through practical experience and to challenge them to solving real life situations. The strategy helps learners to improve their retention of learned experiences and to develop stronger problem solving, critical thinking, and communication skills.

In the second year of high school, each learner presents a report of a project(s) undertaken. The learners are expected to go through the project lifecycle of

conception, planning, execution, presentation and evaluation within the two years of the high school course work. Mini projects and laboratory practice may be based on any parts of the syllabus, but are structured and presented in a way that meets the objectives of the School-Based Assessment. These mini projects may take 3 to 4 weeks with the guidance of the teacher and collaboration of peers. In accordance with the guidelines of the Advanced Level Computer Science and Information and Communication Technology examinations as specified by the General Certificate of Education Board, projects and practical work are part of the student's normal schooling activities, and so these durations are effectively turn-around time.

Furthermore, the Inspectorate of Pedagogy in charge of the teaching of Computer Science in the Ministry of Secondary Education runs capacity building workshops for teachers on the implementation of this teaching syllabus, including specific components such as project-based learning and the production of remedial lessons.

4.1.3 EXPOSITORY

In this method, the teacher presents the contents of a structured lesson to the learners in the form of an exposé. In the context of CBA, this method is only recommended when the teacher is presenting external resources necessary for the acquisition of a competence, such as when explaining or defining a new concept.

4.1.4 DEMONSTRATIVE

In the demonstrative method, the teacher leads learners to do (demonstrate, experiment or formulate) something by demonstrating it, and then he/she guides the learner to do same, while evaluating the degree of learner comprehension.

4.1.5 INTERROGATIVE

In this method, the learner is assumed to have some knowledge on what is to be acquired, or an idea of its contents. Through appropriate questioning by the teacher, the learner constructs his own knowledge based on his/her own understanding of the situation and his/her attempts to link ideas and make some sense out of them. This method is often used when the teacher seeks to determine the level of knowledge in learners.

4.1.6 DISCOVERY

With the discovery method, the teacher creates a pedagogic scenario and uses appropriate materials to allow the learners use trial and error methods for learning. The teacher makes use of his personal experience or those of the learners to solve a given problem, using the means available. Intra-cognitive and collaborative work here are highly utilised. This method must be used with care given that it is expensive and time consuming.

4.2 PHASES IN LESSON DELIVERY

All teaching and learning activities are done in three essential phases: planning, execution and follow-up/evaluation.

4.2.1 THE PLANING PHASE

The planning phase consists of putting in place preparatory materials for teaching and learning activities, acquiring didactic resources, and identifying how evaluation would be done. The planning phase provides foresight into how the lesson would look like. Planning also helps him/her to save time and energy, avoid redundant work, and enhance the presentation quality of the lesson. During the planning phase, all the actions of the teaching and learning processes are previewed to ensure that they develop competencies in the learner. To achieve this, we focus on three elements:

- Essential components of the CBA: knowledge, skill and attitude;
- Integration activities; and
- Evaluation (which is usually formative).

4.2.2 THE EXECUTION PHASE

This is the lesson delivery phase. During a lesson presentation, the teacher may apply one or more of the methods stated above, that help realize the purpose of his/her lesson. The teacher may allow individual or group learning. A group situation could be one in which group members communicate, organize themselves and learn from each other. Whether the teacher allows individual or group learning, the essence should always be to solve a real-life problem in a disciplined manner.

4.2.3 THE FOLLOW UP AND EVALUATION PHASE

This is the phase during which the teacher carries out an analysis of his teaching practice or activities in order to continuously improve. Keeping a diary of remarks and suggestions from each teaching session, can be a useful practice that helps the teacher to correct him- or herself. This practice also helps the teacher develop creative ideas that enable him to reach out effectively to all categories of learners.

Also, the teacher evaluates the learner, usually through a formative evaluation. We can evaluate a competence by proposing to the learners a problem new to them, which problem need not reveal the resources needed to solve it. Alternatively, the learner may actually acquire and use the resources need. In any case, errors committed by the learner during the teaching and learning process should be used constructively, not negatively, to get to the teaching and learning right.

4.3 LESSON PREPARATION

A prepared lesson is what should be used by a teacher to facilitate learning. Hence, a good lesson delivery follows from a good lesson preparation.

To select and teach a lesson from the syllabus, a teacher could follow the steps outline below:

- Express and explore a ***Family of Life Situation (problem)***;
- Pick out an ***Example of Life Situation***. This may guide the teacher to develop a real-life situation. A number of them could be grouped to form a ***Category of Actions***. (Teachers may use a category of actions as a **Topic**.);
- Under each ***Category of Actions***, select an ***Example of Action***. (Teachers may use an Example of Action as a **Lesson**.);
- An ***Example of Action*** could be selected in any order but it is advisable to start with the first and move chronologically to the last in that group of actions provided in the teaching syllabus. Be aware that the list of examples is representative, not exhaustive, and available resources may dictate how best to present and develop the teaching and learning;
- Put in place the necessary didactic material;
- Produce your lesson plan, gauging the lesson within the specified duration;
- Guide the learner into carrying out the actions stated in the Example of Action. A successful completion of each action is a mastery or an achievement of a skill;
- Select the Examples of Actions identified with a project and carry it out using any of the teaching methods given earlier.

In the course of teaching, you can start your lesson through a real-life situation or leave through a real-life situation.

4.3.1 TOOLS FOR LESSON PREPARATION OR TEACHING

The required materials to teach this course are the following:

- Availability of one or more computer laboratories. We recommend a ratio of 2 pupils per computer;
- Availability and possession of up-to-date and authoritative Computer Science textbooks and laboratory manuals;
- Essential classroom and laboratory supplies;
- Availability of relevant software specified as didactic materials;
- Relevant hardware accessories and platforms. In particular, overhead projectors and demo machines should be standard to every Computer Science course;
- Internet connection for documentation, research and teaching;
- Links to online resources within the content's learning environment;

- Schemes of work and lesson notes;
- Students should master and be able to work on MS Windows-based and Unix (Linux)-based machines, even if they are proficient in just one of them;
- C or Pascal language should be used for programming or any other language(s) that revised versions of the syllabuses may introduce;
- Latest popular and stable versions of HTML, CSS, Java and PHP should be used for web authoring and database programming.

5 ACTORS AND THEIR ROLES

5.1 THE TEACHER'S ROLE

The role of the teacher is to create the learning requirements, situations and environments that favor the development of the competency in view. These activities take into consideration the individual characteristics of the learner such as fast learner, slow learner, physically challenged, impaired, etc. The teacher should prepare lessons following the recommendations in the CBA concept.

The teacher follows up learner activities, provides guidance and corrects work-in-progress. The teacher is also required to end every lesson or a group of lessons with assignments that consist of tasks or mini-projects that provide the opportunity for learners to develop their skills in problem solving. Requirements for mini-projects may span modules, and so their correct teaching order and methods must be ascertained.

The teacher is expected to obtain and use appropriate material from sources which will facilitate the development of the desired competency in the learner. The use of projectors and multimedia platforms in the display of information is strongly recommended. These materials help secure learner attention as well as facilitate teaching.

5.2 THE LEARNER'S ROLE

Learners are introduced problems from real life whose solutions require that they exhibit creative, innovative and entrepreneurial abilities. With the teacher's guidance, the learner acquires the necessary ingredients needed to exercise the expected competency. The learner spends time carrying out research on projects given out by the teacher.

6 ASSESSMENT

The primary purpose of assessment and evaluation is to improve learning. Information gathered through assessment helps teachers to determine the learners' strengths and weaknesses in the achievement of the curriculum expectations in each module. This teaching syllabus uses three types of assessments: Formative, Summative and School-Based Assessments.

6.1 THE FORMATIVE ASSESSMENT

Formative assessment is to check for understanding after teaching each sub-topic, topic or module. It usually takes the form of assignments, quizzes, multiple choices questions, fill-in-the-blanks, or simple structural questions.

6.2 THE SUMMATIVE ASSESSMENT

This is a summative examination at the end of a sequence, a term, a year or at the end of a course to find out if the objectives set in them are attained. These examinations could take the form of multiple-choice questions, structural questions, case study questions and practical examinations. They are used to assess students' ability to apply the concepts covered in the course.

6.3 THE SCHOOL–BASED ASSESSMENT

School-Based Assessments (SBAs) are mini-projects carried out by learners under the guidance of the teacher within or after each module. Guidelines for mini projects are outlined for each module. Teachers should follow these guidelines to prepare tasks for their learners. Each mini-project report is evaluated by the teacher using the project evaluation template provided by this guide. These mini-projects are assessed based on submitted formal project reports typically in the form of a presentation or demonstration. The submission provides the teacher with an opportunity to appreciate the student's knowledge and understanding of the subject matter. An Inter alia, the assessment may consider the oral and written communication skills of the learner, as well as his/her ability to think critically and to reflect on the material learned.

As part of assessments, teachers are required to provide learners with descriptive feedback that guides their efforts towards improvement. Classroom teachers are advised to focus on setting practical tests that relate to scientific and daily life-- situations in order to test learners' reasoning and technical skills.

Learners are required to carry out a final project. At the end of some modules, are mini projects to be done by the learners under the guidance of their

teacher. However, at the end of all the modules, the learner should be able to carry out a final project which will help to materialize the module on project management

The section of the syllabus also suggests framework and assessment grids for the final projects.

6.4 GRADING POLICY

Bearing in mind that the subject Computer Science is intended to give students opportunities to use computers and gain experience in solving problems, teachers will be well aware that written examinations need not be the most suitable means of assessment. As such, written, summative and similar examinations should be naturally easier for learners better-exposed to real-life, practical and laboratory experiences. In addition, it is recommended that continuous assessment be used and letter grades, not just marks, be awarded.

As such, the seven-point scale (A through F plus U) that follows provides a record of the learning skills demonstrated by the learner in every module of the course through independent work, teamwork, organisation, work habits, and individual or group initiative.

Remark	Honour Roll	Excellent	Good	Satisfactory	Needs improvement	Failed	Ungraded
% Score	90 to 100	80 to 89	70 to 79	60 to 69	50 to 59	40 to 49	< 40
Grade	A	B	C	D	E	F	U

7 SYLLABUS OUTLINES

7.1 TABLE OF MAIN COMPONENTS OF MODULE 1: COMPUTER APPLICATION AND SOCIO-ECONOMIC IMPLICATIONS

Module 1: COMPUTER APPLICATIONS AND THE SOCIO-ECONOMIC IMPLICATIONS OF THE USE OF COMPUTERS			
Class: Lower Sixth	Theory:	Practical:	Duration of Period:

Specific objectives: On completion of this module, students should have the opportunity to:

a) Describe ways in which computing enables innovation.
b) Explain some areas or domain that the computer could be used to facilitate work.
c) List examples of input and output devices.
d) State clearly the various stages of information processing cycle and give examples.
e) Know the various software that can be used in data processing to accomplish tasks. Discuss the ways in which innovations enabled by computing affect communication and problem solving.
f) Analyze how social and economic values influence the design and development of computing innovations.
g) Discuss issues of equity, access and power in the context of computing resources.
h) Communicate the legal and ethical concerns raised by computational innovations.
i) Discuss privacy and security concerns related to computational innovations.
j) Explain positive and negative effects of technological innovation on human culture.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Description of Computer system	Describe types of computers; Identify input & output devices; Describe how a computer system functions; Manipulate input and output devices.	Input devices; Processing devices; Output devices; Storage devices; Peripheral devices	Compare characteristics: size, processing capabilities, price. of computers; Connect correctly input, processing and output devices; Power a computer (soft and warm booting); Transfer data/information to peripheral devices; transfer data/information from Peripheral devices; Enter information into computer system; Print a document	Learners should be taught how to effectively connect a computer system and the computing devices should be taught equally to them
Utilization of General purpose and other computing applications	• Identify domains of use of general purpose and other computing applications. • Exploit productivity tools (e.g., word processor; Desktop publisher)	• Monitoring and control system; • Simulation and modelling systems; • Batch and online processing systems.	• Determine use of general purpose applications in commerce, industry, science, education, arts and media (essay); • Accomplish tasks via productivity tools (spreadsheets, word processors, database, presentation software).	• Learners should visit appropriate sites: Shops Pharmacies, Banks/IT firms, Insurance Companies, …)

7.2 TABLE OF MAIN COMPONENTS OF MODULE 2: SOFTWARE

Module 2: SOFTWARE			
Class: Lower Sixth	Theory:	Practical:	Duration of Period : 50 minutes
Specific objectives: On completion of this module, students should have the opportunity to: a) Demonstrate an understanding of software Requirements. b) Differentiate between application and system software. c) Know the different types of operating systems. d) Understand the structure, functions, and philosophy of operating systems. e) Understand scheduling, dispatch and deadlocks simulation computing.			

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Management of processes in a computer.	Explain the role of OS in process management Explain concepts in process management Describe scheduling strategies used by the OS to manage processes.	Process Sharing of processor: Multi-tasking Multi-programming Process creation and termination Concurrent processes; Race condition; Mutual exclusion; Deadlock; Deadlock detection and resolution strategies Context switching. Scheduling strategies: Notion of burst time pre-emptive notion of quantum time, Round Robin Priority Shortest Remaining Time Next. non pre-emptive First Come First Served, Shortest Job First, Gantt chart representation of process scheduling.	• Describe how the file directory is organised (single level, two level, tree structure directories) • Determine ratings of file access methods • Outline file attributes • Outline OS operations on a file • Differentiate Sequential Access and direct access Compare file systems.	The learners are supposed to know just the concepts
Management of file by OS	• Describe how file are organised and stored in the computer • Determine ratings of file			Emphasis should be made on the comparison of the

	access methods • Create and managing files in the computer Explain file system (eg FAT16, FAT32, NTFS, ext in unix environment).			various file systems. That is FAT16, FAT32, NTFS Also for the GUI environment, the interface or window should be mentioned to the learner.

Commented [elyse ple2]: replace that is with :

7.3 TABLE OF MAIN COMPONENTS OF MODULE 3: COMPUTER NETWORK, DATA COMMUNICATIONS AND SECURITY

Module 3: COMPUTER NETWORKS, DATA COMMUNICATIONS AND SECURITY			
Class: Lower Sixth	Theory:	Practical:	Duration of Period: 50 minutes
Specific objectives: On completion of this module, students should have the opportunity to: a) Appreciate the need for data communication networks. b) Identify the different equipment and components used. c) Know all transmission modes, and media. d) Describe the various network communications standards e) Recognize the need for communication Protocols. f) Understand modulation and multiplexing. g) Explain the concepts of computer Networks and Topology. h) Understand network implementation and security.			

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Exploration of computer network platform	• recall types of network (LAN, MAN, WAN) • Describe types of network connections (point to point, multipoint, …) • Explain features of network operating systems (multitasking, multi-user) • Describe network architectures.	• Types of network (LAN, MAN, WAN); • Components of a network; • Network topology; • Network operating system; Network architecture.	• Choose appropriate type of network for a given context; • outline network software; • Compare, network topologies based on characteristics like: robustness, scalability … • Select suitable physical and logical topologies;	Better ways or best route of connecting network should be examined. Compare physical topology apart from logical topology

Commented [elyse ple3]: compare physical topology with logical....

			Differentiate types of network architectures.	
Setting up a computer network	• Explain how network devices function (MODEM, repeaters, switches, bridges, routers, and gateways …); • Describe transmission mediums (cables and wireless); • Configure the operating system for network; Explain how mobile communication technology work.	• Network devices; • Transmission mediums; • network configuration; • Mobile technology; Network troubleshooting.	• Select and connect the hardware components of a network; cables, switch, router, modem; • Compare transmission mediums based on characteristics (data rate, transmission distance, ease of installation …); • Explain the difference between wired and wireless transmission; • Outline errors that may occur in a network; • Set up a network; • Configure an operating system for networking; Troubleshoot a network.	You should have an open mind when teaching this section as technology evolved frequently. Explanation should be given how equipment function individually as well as collectively Also insist on the specification of addresses, as well as some errors that may occurred.
Data security, privacy and integrity	• Explain data security; • Describe safety principles in protecting data and network from malware; • Use antivirus to protect computer network from virus, Trojan horse, worm ; • describe different types of error detecting code (parity bits, hamming codes, cyclic redundancy checks/check sum); • Explain measures used to protect computers and networks from intruders and natural disaster (username and password, firewall, data encryption, backup, …); Recognize ownership of digital information and guard against digital theft and plagiarism.	• data security, privacy and integrity; • Protect network, computer and data from viruses, spyware unauthorized access ... • Data backup; • Firewall; • Data encryption; • Malware; Plagiarism.	• Explain concepts related to data security (privacy, integrity, …); • Explain how backup ensures data security; • Explain in a report technique used to fight plagiarism; • Apply safety principles in protecting data and network from malware (scan every incoming document/program before opening or running, …); • Scan a computer system using an antivirus; • Set up a firewall and web filtering using an antivirus; • Save data on cloud storage systems (Google drive, …).	The importance of plagiarism should be mentioned to the learner. Practical lessons should be conducted at this level to illustrate security measures to learners.
Networking standards and protocols.	• Describe different network standards, and protocols; Explain the OSI	• Standards and protocols; • The OSI reference model;	• Produce a report comparing the OSI and TCP reference models;	The OSI model should be well elaborated Ask learners to produce a report here to show the

	reference model.	Internet protocols.	Discuss internet protocols (TCP, UDP, IP, FTP …) (essay).	other aspects of the various protocols used in the OSI model
Using the Internet	• Describe the history of the internet; • Explain the concepts intranet, extranet and Internet; • Describe services available on the Internet (e-commerce, e-learning, e-banking …); • Exchange information using email; • Use search engines; • Doing business online; • Change privacy settings.	• history of the Internet; • Notions of: Internet Intranet and extranet; • Internet services; • Safety and security risks in participating online;	• Explain in a report major events in the history of the internet; • Select suitable hardware and software needed for access to internet connectivity; • Access a website using a browser; • Select appropriate internet service for a given context; • Manage bookmarks using a browser; • Send and receive e-mail; • Apply for a job online; • Apply for scholarship; • ensure privacy; • Search information on the internet using a search engine; • assess online information for relevance, bias, validity, reliability and sufficiency; Download from the internet (images, files, software, and drivers).	Emphasis should be made known to the learners that everything that is done or work carried out online is businesses

7.4 TABLE OF MAIN COMPONENTS OF MODULE 4: DATA STRUCTURES AND ALGORITHMS

MODULE 4: DATA STRUCTURES AND ALGORITHMS
Specific objectives: On completion of this module, students should have the opportunity to: a) Demonstrate how data is efficiently organized in a computer b) Practice and expand their ability to analyze and solve problems using a computer. c) Analyse more complex problems and the development of solutions using Algorithmic tools.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Simple data types	Distinguish the various data types.	• Character; • Integer; • Real or float; • Boolean; Representation of each data type in a programming language.	• Outline examples of a given data type; • Calculate memory allocated to each data type; Declare variables using a programming language syntax.	Specify the types of data type.
Overview of Algorithms	• Describe forms of algorithms; • Describe characteristics of a good algorithm; Write steps to solve a problem.	• Algorithm; • Forms of algorithm (pseudocode, flowchart); • Characteristics of a good algorithm.	• Model the solution to a complex problem in a series of precise and finite set of steps; Distinguish good and poor algorithms with respect to the spelled-out characteristics.	State good characteristics for a good algorithm which is already in examples of actions
Algorithmic Design strategies	• Break down a complex problem into simpler solvable forms; Build solution for a problem using the different modules of the problem.	Design strategies: • Top-down design; • Bottom-up design; • Step-wise design; Modular design.	• Choose a design strategy to solve a problem; • Split complex problems into simpler problems until they are easily solvable; • solve the simple problems and combine solutions to build the complex solution; • Split a problem into different modules which can run independently; Create an interface to coordinate the different modules.	Each design strategy should be well explained

7.5 TABLE OF MAIN COMPONENTS OF MODULE 5: PROGRAMMING

MODULE 5: PROGRAMMING			
Class:	Theory:	Practical:	Period:

Specific objectives: On completion of this module, students should have the opportunity to:

1. Demonstrate the ability to use different data types in computer programs;
2. Demonstrate the ability to use control structures and simple algorithms in computer programs;
3. Describe fundamental programming concepts and constructs;
4. Plan and write simple programs using fundamental programming concepts;
5. Apply basic code maintenance techniques and conventions when writing programs.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Classification of programming languages.	• *Present programming as a tool to solve problems using the computer;* • Describe types of programming languages (low level and high level language); • Explain the advantages and disadvantages of using various categories of programming languages; • Identify areas of application of various programming languages; Explain why some programming languages are preferred to others.	• Programming; Program..	• Justify in writing the choice of a programming language for a given context; • Differentiate between Program and algorithm; • Classify a given programming language; Differentiate machine code and human understandable code.	Programming generations (G1, G2, G3, …) should be explained for the types of programming language

Syntax and semantics.	• Identify the main elements of a program and give examples;
	• Practice how to declare and use various program elements in a code;
	• Explain various program structure;
	• Explain the importance of documentation in
		• syntax ;
		• semantics;
		• Elements of a program
		  ○ Identifier,
		  ○ Variable/identifiers,
		  ○ Constant,
		  ○ Reserved word,
		  ○ Character sets,
		  ○ Simple data types;
		• Scope of variables in a program: local and global; programming language;
			• Describe the main elements of a program;
			• Outline examples of main elements of a program;
			• Declare and use various program elements in a code;
			• Document a programming process;
			• Write global and local
				Apply the concepts to a programming language. For example, in C-programming

	programming; Explain the role of subroutine in a program.		variables in a given programing language; • Declare a variable in a chosen programming language; Write a subroutines.	
Program structure	• State elements of standard program structure (Program header or pre-processor directives, Variable declaration, Constant declaration, Body of the program, Begin/end notations); • Assignment notation; • Improve on the structure of a standard program.	• I/O functions e.g. Pascal: writeln; C: printf, scanf; Variables, expressions, and assignment statements.	• Explain the use of I/O functions; • Write simple programs e.g. compute areas, list of statements ...; • Store and manipulate numbers and text in a program using variables, expressions, and assignment statements; • Use I/O functions e.g. o Pascal: writeln; C: printf, scanf.	Make emphasis on some standard codes like the Fibonacci sequence

7.6 TABLE OF MAIN COMPONENTS OF MODULE 6: SOFTWARE DEVELOPMENT

MODULE 6: SOFTWARE DEVELOPMENT			
Class:	Theory:	Practical:	Period:
Specific objectives: On completion of this module, students should have the opportunity to: a) Be familiar with the fundamental concepts of software development. b) Identify software requirements. c) Appreciate the design processes in software development. d) Understand verification and validation Process. e) Apply Software Management.			

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Developing Software requirements	• Explain software requirement analysis; Illustrate requirements of specific software.	• Software requirement and specification; • Technical requirements; User requirement.	• Outline activities in software requirement analysis (elicitation, validation, specification and verification); • Write a requirement specification document; Compare technical and	Software requirement is not a stage, so the **Nature** of software should be described clearly when designing requirement specifications. The word **Nature** is the whole idea

			User requirement.	behind for what a software is to do. Technical requirements should be emphasis as the elements needed to develop software
Design process in software	• Identify components of a software to be designed; • Explain design elements; Specify design elements.	• Data types and data structures design; • Architectural design; • Interface design; • test data; Algorithm design;	• Identify stages in software design (interface, architectural and detailed design); Produce design document.	Implementation strategies were replaced with specify design elements. This concept should be explained well
Verification and validation Process	• Explain software verification and validation; Explain testing mechanisms of a developed software.	• Verification; • Validation; • Testing methods: ○ Unit testing; ○ Integration testing; ○ Smoke testing; ○ Regression testing ; Acceptance testing.	• Describe methods to test a software (code review, static code analysis, unit testing, …); • *Apply testing methods;* Differentiate between testing methods.	You are not supposed to talk of data validation and data verification concepts as both do not have common characteristics but instead as a stage in the designing of a software Learners should be described the various testing strateties practically which are the methods applied for testing
Management of software development process	Explain project management activities.	• Outsourcing; • Management activities: ○ Proposal writing, ○ Project planning and scheduling, ○ Project monitoring and reviews, ○ Personnel selection, Evaluation report writing and Presentation.	• Produce a report: ○ Software development stages; ○ software outsourcing; software management activities;	Emphasis should be made on the elaboration of the path taken to realise a software

Commented [elyse ple4]: this concept should be well explained

Commented [elyse ple5]: Replace "you" with " the teacher"

7.7 TABLE OF MAIN COMPONENTS OF MODULE 7: COMPUTER SCIENCE PROJECT

MODULE 7: COMPUTER SCIENCE PROJECT			
Class:	Theory:	Practical:	Period:

Specific objectives: On completion of this module, students should have the opportunity to:

a) Work in a team to integrate and apply the learning outcomes from the modules to the later stages of a sustained project.
b) Develop a small computer related business from the point of starting-up to running it.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Starting a business	• Identify businesses related to basic skills Explain customer needs	• Business world • E-commerce	• Study IT success stories (Nji Collins, Arthur Zang, Mark Zuckerberg, Bill Gates, Steve Jobs, Jack Mah,) • Study IT success stories (Google, Apple, Amazon) Detect community needs that can be solved with IT.	Success stories of prominent IT personalities should be presented to learners like Bill Gates and the rest. Learners should be asked to investigate about the success stories of this prominent IT personalities

7.8 TABLE OF MAIN COMPONENTS OF MODULE 8: COMPUTER ORGANISATION AND ARCHITECTURE

Module 8: COMPUTER ORGANIZATION AND ARCHITECTURE			
Class: **Lower Sixth**	Theory:	Practical:	Duration of Period: 50 minutes

Specific objectives: On completion of this module, students should have the opportunity to:

a) Demonstrate an awareness of the nature of the hardware involved in computer systems.
b) Appreciate the choice of a combination of particular types of peripheral devices, the operating system and the processor.
c) Understand and analyze computer systems architecture.
d) Explain the structure and functioning of computer instruction set.
e) Describe the organization of different bus systems and their characteristics in a computer system.
f) Understand low-level parallelism and its implementation in a processor.
g) Know basic logic gates.
h) Carry out arithmetic operations with basic digital circuits.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Polling and Interrupts	Illustrate how devices interact.	• Polling • Interrupt Detection of interrupt	• Difference between Interrupt and Polling; Describe interrupt detection strategy.	Learners are supposed to know the clear distinction between polling and interrupt
Binary Arithmetic	• Carry out arithmetic operations (Addition, Subtraction, division, multiplication) in base 2, 8, 10 and 16. • Convert from one base to another. Represent numbers using sign magnitude, one's and two's complement.	• Number systems • Data representation Sign magnitude, one's and two's complement.	• Distinguish between the use of kibi and kilo, mebi and mega, gibi and giga, tebi and tera; • represent binary numbers in one's and two's complement ; • Convert an integer value from one number base representation to another; • Perform binary addition and subtraction; Describe practical applications of Binary Coded Decimal (BCD) and Hexadecimal.	Emphasis should be made on arithmetic operations SI unit used to represent absolute values should be highlighted
Logic gates and Boolean arithmetic	• Explain combinational and sequential circuits; • Identify logic components; • Simplify Boolean expressions.	• Logic gates; • Boolean expression; • Boolean Algebra; • Truth tables; Logic circuits	• Differentiate between combinational circuits and sequential circuits. • Sketch the NOT, AND, OR, NAND,	Learners should know that inputs used for Logic gates ranges from 1, 2 till 3 Teachers should be able to build AND, OR, and NOT gates using

	• Build logic circuits Implement other logic gates using universal logic gates (NAND and NOR)		NOR logic gates; • Construct the truth tables for AND, OR, NAND, NOR gates; • Construct a logic circuit from a logic expression or a truth table; • Construct a truth table from a problem statement, a logic expression or a logic circuit; • Derive a logic expression table from a problem statement, a truth table or a logic circuit; Implement logic gates using universal logic gates (NAND, NOR)	the universal gates like NAND and NOR gates Emphasis should be made on sequential circuit which amplifies and combinational circuit which does not amplify

7.9 TABLE OF MAIN COMPONENTS OF MODULE 9: INFORMATION SYSTEMS

Module 9: INFORMATION SYSTEMS			
Class: Lower Sixth	Theory:	Practical:	Duration of Period: 50 minutes
Specific objectives: On completion of this module, students should have the opportunity to: a) Demonstrate knowledge and understanding of main aspects of Information Systems. b) Demonstrate an understanding of the components of an information system and the links between them. c) Introducing data flow diagrams and their use in the description of an information system. d) Understand the need for designing user interfaces, and becoming familiar with design principles.			

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Exploring Information Systems	• Identify the components of an Information system • Describe the roles of each component in an IS • Classify data capture methods Identify application areas for information systems	• System • Information system • Data • Information Data capture	• Describe the components of an information system • Describe the different types of IS • Explain the purpose of IS in and organisation • Define a problem in your community for which an IS can be developed (Project) Discuss (essay) the need for an IS in an organization.	Exploring information systems means a detail analysis of this important concept. Learners should carry out research on information systems and put it in writings in a form of an essay for presentation in class The list for information system is not exhaustive, so teachers should talk of other types. Setting up an IS in an institution like a school should be specific. Learners could be asked to put in place an IS for a school library only, or for processing marks. It should just be an aspect and not a complete IS for the whole school. Teachers should download **"ARCMAP"**

7.10 TABLE OF MAIN COMPONENTS OF MODULE 10: DATABASE SYSTEMS

MODULE 10: DATABASE SYSTEMS

Class:	Theory:	Practical	Duration of Period:

Specific objectives: On completion of this module, students should have the opportunity to:

a) Describe the nature and purpose of database models and how they are used;
b) Describe the functions of the tools readily available in database packages;
c) Appreciate the advantages of relational database systems over traditional file systems;
d) Understand how a relational database is designed, created, used, and maintained;
e) Describe the components of a database management system;
f) Describe the different types of database organizations;
g) Distinguish between shared and distributed databases;
h) Describe some ways databases are used on the Web.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Understanding Relational database management systems	• Explain relational database concepts; • Identify links between related tables; Use appropriate data types in relational tables – text, numeric, date, Boolean and memo.	• Relational database; • Entity, field (attribute), key field (primary key), secondary key, record (tuple), foreign key.	• Describe the significance of each relational database concept; • Organize data into related tables; Select the appropriate data type for each field.	The concept of relational database should be well explained
	• Explain relational database relationships	RDBMS relationships: • One-to-one; • One-many; • Many-many;	• Determine various relationships between tables or entities;	
	• Define and implement joints; • Explain constraints and constraints enforcement; • Distinguish between Referential integrity and check constraints;	• Inner joins and Outer joins; • Notion of constraints and constraints enforcement; Referential integrity Vs check constraints.	• Set up joins between tables in a RDBMS; • Enforce constraints using features of your RDBMS and SQL; • Work with modification anomalies on a database.	Relational database concept should be well illustrated in the computer laboratory using SQL.

7.11 TABLE OF MAIN COMPONENTS OF MODULE 11: SOFTWARE DEVELOPMENT II

MODULE 11:			
Class:	Theory:	Practical:	Period:

Specific objectives:
On completion of this module, students should have the opportunity to:

a) Use a variety of problem-solving strategies to solve different types of problems independently and as part of a team;
b) Develop Algorithmic thinking skills, and Design algorithms according to specifications;
c) Design software solutions to meet a variety of challenges;
d) Apply a software development life-cycle model to a software development project.
e) Show understanding of how testing can expose possible errors (syntax errors, logic errors and run-time errors) and error detection in the development of a system.

Category of actions	Examples of actions	Core knowledge	Skills	Explanatory Notes
Problem-solving strategies	• Solve a given problem in well-defined steps; • Demonstrate mastery in solving a problem by solving its constituent parts; • Explain models used in solving computing problems (divide and conquer, stepwise refinement, incremental, …); • Use models to solve a problem.	• Algorithm. • Algorithms Development Techniques (Divide and Conquer or Stepwise refinement, Incremental, Parallelism).	• Explain the concept of algorithm; • Enumerate examples of problems solved by divide conquer; • Write simple algorithms; • Describe approaches to design algorithms (top-down and bottom-up design); • Resolve a problem using problem solving models.	This module is supposed to be taught in a way that learners carry out projects.
Designing software solutions	• Use a programming language; • Use an IDE; • Expose faults in programs and ways of avoiding faults; Describe features found in a typical Integrated Development	• Features found in a typical Integrated Development Environment (IDE); • IDE; • Modify functioning of existing codes to sort or search variables in arrays.	• Transform an algorithm into a program;. • Write a program; • Translate, test, run a high-level language using an IDE; • Identify features of an IDE; • Write programming codes using a given	The use of IDE implies translate, test and run

	Environment (IDE): (Coding, including context-sensitive prompts, initial error detection, including dynamic syntax checks).		compiler ; • Identify errors in a program; • Correct errors in a program; • Write codes with correct syntax.; Write programs using arrays.	