The New Senior Secondary Curriculum for Sierra Leone

Subject Syllabus for Computer Science
Subject stream: Core Subject

This subject syllabus is based on the National Curriculum Framework for Senior Secondary Education. It was prepared by national curriculum specialists and subject experts.

Curriculum Elements for Computer Science – a core subject

Coding has become such an essential skill that it has become necessary to make it part of our grade school curriculum. Coding called computer programming is how we communicate with computer. It is a practical subject where students can apply the concept learned in the classroom to real-world systems.

Rationale for the Inclusion of Computer Science in the Senior Secondary School Curriculum

This specification incorporates established methodologies, computing, technological advances that would prepare students that are college bound as well as students that would prefer to enter the job market after high school. Instructors can choose from a range of programming languages for the programming project to tailor to the strengths and preferences class objectives and needs.

a) The study of computer science in senior secondary school will deepen critical thinking and empower students to grasp complex or abstract ideas and transform them into beliefs or direct actions.
b) High school students who study computer science have familiarity with technology with deep appreciation for cognitive computation which give them a distinct advantage over their peers since.
c) Cognitive computing enables students to have the ability of higher-order thinking and problem-solving skills.
d) Computer science programs for high school students deepen critical thinking as they empower students to grasp complex or abstract ideas and transform them into beliefs or direct actions.
e) relay bits of information from multiple sources.
f) Which is in line with so many jobs today requiring teamwork.
g) Computer science collaborative learning sharpens interpersonal skills, teaches active listening and how to draft eloquent reports that have been thoroughly researched.
h) Computer science incorporates a precise, mathematical language to relay bits of information from multiple sources
i) High school computer science programs ignite intellectual inquiry, build confidence in problem-solving, creativity, collaboration, and enhances communication skills that students need to innovate, invent, and lead in college and beyond.
j) Innovative computer science classes teaches students how to navigate the ever-changing world of technology:
k) A high-quality computer science course in high school can help students develop mental dexterity and adaptable skills that help them to succeed in life.
l) The escalating interest that spoors many governments to demand that computer science be taught in high schools is its fluence in 21st industrial growth, its pragmatic expertise for tomorrows jobs and its benefits to futuristic technologies.
m) To prepare all students with the creative, collaborative and digital problem-solving skills of the future, schools must teach computer science as part of the core curriculum.
n) Computer science is not just about coding. It is also about computational thinking, interface design, data analysis, machine learning, cybersecurity, networking and robotics.

General Learning Outcomes/Broad Goals

Students will acquire the ability to apply fundamental principles and concepts of computer science including abstraction, decomposition, logic, algorithms, and date representation. Upon completion of this course, students should be able to solve problems by writing programs to do so.

The course is separated into two parts. SS1 & SS2 cover the same material. In SS3, students will branch on to either digital development qualification or digital authoring. Digital development focusses on programming. Digital development qualification focusing on programming. Digital authoring concepts branch into digital authoring practice which includes Multimedia authoring and database development concepts.

a) Expected outcome would be that these students have spent most of their time learning to think critically and solve problems and would fit squally as abled participant in the global marketplace.
b) Students as expected would have achieved mental dexterity and adaptable skills, traits that would in a world inundated with so many forms of technology.
c) Students who enrolled in computer science program develop skills to compare and verify details, apply logic to diagnose results and persevere to find workable solutions.
d) Develop logical thinking and problem-solving are all a part of the computer science curriculum.
e) High school students who successfully completed computer science programs would have enhanced their ability to imagine an object or service, communicate its value, and execute the design.
f) From learning computer science, more and more girls are becoming curious, agile thinkers.
g) According to a recent College Board reports across countries, the number of girls taking Advanced Computer Science Principles has skyrocketed, suggesting that these students value logic, computation, organization, and collaboration that the discipline teaches.
h) The "language" of computer science crosses international and cultural boundaries. Therefore, global citizens of the 21st century must acquire the appropriate "language skills" to exist in a technological age.
i) It has been acknowledged that learning computer science encourages creativity, problem-solving, ethics and collaboration – skills which aren't just important for technical careers in the developed world, but valuable for every career in all economies.
j) The information science and technology driven by computer science is making handwriting increasingly obsolete, complex arithmetic is no longer done by hand, and the internet has replaced the need to memorize many basic facts.
k) Learning computer science encourages creativity, problem-solving, ethics and collaboration – skills which aren't just important for technical careers.

Subject Content Outline by Broad Themes & Specific Topics

Students will acquire the ability to apply fundamental principles and concepts of computer science including abstraction, decomposition, logic, algorithms, and date representation. Upon completion of this course, students should be able to solve problems by writing programs to do so.

The course is separated into two parts. SS1 & SS2 cover the same material. In SS3, students will branch on to either digital development qualification or digital authoring. Digital development focusses on programming. Digital development qualification focusing on programming. Digital authoring concepts branch into digital authoring practice which includes Multimedia authoring and database development concepts.

Learning Basic computational thinking

- Problem solving
- Pseudo code
- Codes tracing
- Flowcharts

The Basic of Computer Science

- Types of identifiers
- Operators
- Control structures

Object Classes and Inheritance

- Specification
- Programming concepts
- Structured Programming
- Relations
- Operations

Lists and Arrays

- One-dimensional array
- Array Lists

Two Dimensional Arrays

- Two dimensional arrays
- Row-column traversal
- Fir each-loop traversal
- Row-by row array processing

Sorting and Searching

- Selection sort
- Insertion sort
- Merge Sort
- Binary Search

Structure of the Syllabus Over the Three Year Senior Secondary Cycle

	SSS 1	SSS 2	SSS 3
Term 1	Software Development • Language (Java) • A software developer • Software engineering: Prototyping, incremental development, rapid application development, agile software development, waterfall model Software and Our World • The software developer • Soft effect on the modern world • Choosing your integrated development environment (IDE) • Hello world Software Life Cycle • The software development circle • Developing class hierarchy • Testing	Writing Classes • The class declaration • Instance variables • Constructors • Methods Putting it all Together • The Circle and Circle Runner Classes • Understanding the key word new when constructing an object The Reference Variable • The reference variable versus the actual object • The null reference	Sorting • Background on sorting data • Insertion sort • Selection sort • Review 2D Array • The 2D Array • More algorithms • The accumulate algorithms • The find-highest algorithm • The connect - four advanced algorithm • Review
Term 2	Primitive Types • Introduction • Syntax • The console screens • Primitive variables • Mathematical operations • Arithmetic overflow • Type of errors Classes and Objects	Continuation of Classes • Parameters • Overloaded constructors • Overloaded methods • Static, static, static Encapsulation • Data encapsulation • Scope • The keyword this	Inheritance • Inheritance • Polymorphism • The object class • Projects Recursion • Recursion versus looping • The base class • Projects

	• Relation between classes and objects • The java app Strings • The string variable • The string object • Virtual representation of a string variable • String concatenation • Comparing string objects • Important string methods • A string is immutable • Escape sequences	• Illegal augment exception • Review Data Structure • What is data structure • The array	• Steve
Term 3	Classes • The math class • The integer class • The double class • Autoboxing and unboxing • Summary of the integer class and double class Boolean Expression and If Statement • Introduction • Relational operators • Logical operations • Precedence of java operators • Conditional statements Iteration • Introduction • Looping statements • Standard algorithms	Algorithms • How we use algorithms • Why we use algorithms • Why algorithms are important • Algorithms versus pseudo code versus java code Algorithms Continued • The swap algorithm • The copy algorithm for the array • The accumulate algorithm • The find highest algorithm Array List • The Array List • Important array list methods • The copy algorithm for the array list • The sequential (or linear) search algorithm • The accumulate algorithm • The find-highest algorithm • The accumulate advanced algorithm	Using Recursion Method • Merge Sort • Binary sort • Projects • Review

		• The find highest advanced algorithm • The twitter sentiment advanced algorithm	

Teaching Syllabus

Topic/Theme/Unit	Expected learning outcomes	Recommended teaching methods	Suggested resources	Assessment of learning outcomes
Software Development: goal of the course • Program design and algorithm development • Code logic • Code implementation • Code testing • Documentation	The course reflects what computer science computer science teachers, professors, and researchers have indicated should be a preparatory course for those who want to study computer science or related subjected at the university level. The expectation is to learn how to solve different kinds of problems using a particular computer language (in our case, java).	• The course will take the form of lecture format. • It will include a minimum of 20 hours of hands on and structure lab experience. • Students will be given ample chance to work with big, complex set of related classes that would help them to understand concepts like inheritance and polymorphism.	• Textbook • Picture and chart • Computers • Overhead projector with white board+.	Students would understand the expectation of the course, which is to become proficient at solving different type of problems using a particular computer science language, (java programming language).

Software Life Circle • Program specification • Software engineering • Prototyping • Rapid application development	• Students would learn how to write code from a given specification. • Students would now how to use given program specification to design a program to do what it is supposed to do. • Students would learn the different software models for developing and maintains quality software.	• The course will take the form of lecture format.	• Textbook (to be determined). • Picture and chart. • Computer and internet availability.	• Students will be conversant with top-down design versus bottom-up design. • Students would be familiar with different testing methods that would ensure that the program works.
Primitive Types	• Students would have learned the fundamental building blocks needed to write software. • The syntax is the technical way of writing the code.	• The methodology will use a computer language (ex. java language) to write code. • All structures learned applies to any computer language.	• Computers • Overhead projector	Students would be required to analyse codes in a multiple-choice section and find hidden errors.
Using Objects	• Students are introduced to the most fundamental futures of object-oriented language (java). • Students to understand the relationship between object and class. • Students get introduced to the java api (application programming interface)	• The methodology will use the java api to teach a language whose foundation is built in object and classes. • All structures learned applies to any computer language	• Textbook • Picture and chart • Computers • Overhead projector	• Students should be able to show clear understanding of why a class, what is an object and what is the meaning of method. • What is the java api and how it is used. • Student should be conversant with autoboxing and unboxing.

Boolean Expression and If Statement	• Student understands using conditional statements for program flow in one or many directions based on outcome of a Boolean Expression. • Students the use of relational operators and rational operators for decision making.	• The method will take the form of executing programs that includes relational and logical operators. • Students will write programs that include conditional statement.	• Textbook • Picture and chart • Computers • Overhead projector	• Student should have a good grasp of scope of a variable. • Students should be able to evaluate compound conditional statements. • Use de Morgan to decipher complex statement complex conditional statements according to some given criteria. • Students would be able to use the negation of a relational operator. Lab work: ex. design a magpie lab – an exploration of some of the basics of natural language. you will work with strings and parse them for reconstruction.
Iteration and Standard Algorithm	At the end of the lessons, students should be able to execute iteration causes statements. Statements that can be repeated more than once using looping structures. • A for loop • A while loop • An infinite loop • Boundary testing	• The teaching methods will include executing standard algorithms using loop structures.	• Textbook • Picture and chart • Computers • Overhead projector	Students are expected to be able to read and or write any tasks that involves • Variables • Conditional statements • Loop structures (for and while loops) • Overhead projector Lab work: ex. design a consumer lab – explore ways online reviews are created.

Writing Classes	In this unit, students will learn how to design their own classes including • class declaration • parameterized constructors and parameters • virtual attributes called instance variable • have grasp of public versus private visibility	the students will be taught the keyword new to be used when constructing an object incorporating • Data type • Object reference variable • Call to the parameterized construct	• Textbook • Picture and chart • Computers • Overhead projector	Students should be able to demonstrate full understanding od • Designing classes • Writing constructors • Writing methods • The dot operator • The null reference • Data encapsulation and information hiding
Array How to Use Algorithm Array List	Students will learn that solving certain problems require need a list of variables and objects. • The array • The two-dimensional array • The Array List	The methodology used will get students to understand the use of data structures in computer science. • Simple data structures • Complex data structures. • Teachers will explain algorithm versus pseudocode versus real java code	• Textbook • Picture and chart • Computers • Overhead projector	Students would be required to know the advantages and disadvantages of the three data structures • The array • The two-dimensional array • The Array List • The array index out of bounds exception Lab work: ex. design a steganography – how to conceal messages within images. Lab work: ex. design a data lab – that searches for a dataset to find answers to questions.
Inheritance	Students will acquire key ideas on • A class hierarchy • Inheritance • A subclass • A child class • Polymorphism • The object class	The instructor will show how • Super classes and subclasses are related with a class hierarchy • An object within a class hierarchy may act independently of	• Textbook • Picture and chart • Computers • Overhead projector	Students will show an ability to handle inheritance in object-oriented programming. • They should show clear understanding of data encapsulation within class hierarchy. • Dynamic binding

		each other in explaining polymorphism		Lab work: ex. design a steganography – how to conceal messages within images. Lab work: ex. design a picture lab – uses 2-d arrays to modify digital pictures.
Recursion versus Looping Search and Sort Algorithms	Students will acquire key ideas on • What is a recursive method? • Recursion versus nested loops • Tracing recursive methods • Binary search • Merge sort	The instructor will explain recursion and self-similarity versus the for loop and while loop. • The base case • Factorial recursive method	• Textbook • Picture and chart • Computers • Overhead projector	Students should be able to picture and chart • Hand-trace factorial recursive method • Hand-trace Fibonacci recursive method Lab work: ex. design a guessing game Lab work: ex. design a solitaire card game