Introduction to
Coding
in the
National Curriculum of Pakistan
(2022-23)

National Curriculum Council Secretariat
Ministry of Federal Education
and Professional Training

Introduction

In accordance with the directive of the Prime Minister's Strategic Reforms Initiative, the Ministry of Federal Education & Professional Training has included Coding Skills in the National Curriculum of Pakistan (NCP) 2022-23 for Grades 6-12.

The incorporation of coding into the National Curriculum of Pakistan (2022-23) signifies the importance of digital skills and technological literacy in the modern world. With technology advancing rapidly, coding has become an essential skill for learners to understand, create, and innovate in the digital age. By integrating coding into the curriculum, Pakistani students will be equipped with the knowledge and skills necessary to thrive in an increasingly digital, innovative, and technology-driven society.

Overview of Coding in National Curriculum of Pakistan 6-12

The inclusion of Coding in the NCP for Grades 6-12 underscores its significance as a foundational 21st century skill. The inclusion of coding skills to the NCP (2022-2023) signifies the following areas of learning for all students of Pakistan:

1. Digital Literacy:

Coding education promotes digital literacy by providing students with the tools to understand and interact with technology. It enables them to navigate digital environments, utilize software applications, and comprehend the underlying mechanisms of the digital world.

2. Future Workforce Preparedness:

Coding skills are in high demand in today's job market and are projected to be even more critical in the future. By introducing coding in the curriculum, Pakistani students will be prepared for the evolving job markets in software development, data analysis, artificial intelligence, and other emerging fields.

3. Problem-Solving and Logical Thinking:

Coding fosters problem-solving skills and logical thinking abilities. It trains students to break down complex problems into smaller, manageable components and develop step-by-step solutions. Through coding, students learn computational thinking, algorithmic logic, and analytical reasoning, which are valuable skills applicable beyond programming.

4. Creativity and Innovation:

Coding encourages creativity and innovation by enabling students to transform their ideas into tangible products. It empowers them to think critically, design solutions, and bring their concepts to life through coding projects. By nurturing creativity, coding education promotes entrepreneurship and encourages students to become active creators rather than passive consumers of technology.

5.Technological Empowerment:

n a digital world, understanding coding empowers individuals to have more control over technology. It enables them to customize software, develop applications, and automate tasks, allowing them to leverage technology to address specific needs and create meaningful solutions.

6.Future-Proofing Education:

The integration of coding in the national curriculum reflects the need to adapt education to the changing technological landscape. By equipping students with coding skills, the Government of Pakistan ensures that its education system remains relevant and aligned with global advancements, preparing students for the challenges and opportunities of the digital era.

National Curriculum of Pakistan
(2022-23)

Coding in
Computer Science

Grades 6 to 8

Domain C: Algorithmic Thinking and Problem Solving

Standard: Identify, define, and analyze a problem, and apply algorithmic thinking and problem-solving strategies to develop step-by-step solutions to solve problems

Grade 6	Grade 7	Grade 8
Benchmarks: Students will be able to identify, define and analyze a problem; apply the concepts of computational thinking and problem-solving strategies to solve complex problems; apply basic concepts and concept of nesting in algorithmic design thinking		
Student learning outcomes		
[SLO: CS-06-C-01] Students will be able to identify, define and analyze a problem	[SLO: CS-07-C-01] Students will be able to apply the concept of computational thinking to handle complex problems.	[SLO: CS-08-C-01] Students will be able to apply the concepts of computational thinking and problem-solving strategies to solve complex problems by identifying the most efficient algorithm
[SLO: CS-06-C-02] Students will be able to apply basic algorithmic thinking to solve different types of problems.	[SLO: CS-07-C-02] Students will be able to apply concepts of conditional statements, finite and infinite loops to write different algorithms.	[SLO: CS-08-C-02] Students will be able to apply the concepts of nesting in algorithmic design thinking.

Domain D: Programming

Standard: Understand and apply fundamental programming constructs using visual and textual programming tools

Grade 6	Grade 7	Grade 8
Benchmarks Students will be able to recognize the fundamentals of computer programming; analyze how computers encode and decode information; apply fundamental programming constructs by creating various types of programs using visual programming tools.		
Student learning outcomes		
[SLO: CS-06-D-01] Students will be able to analyze the fundamentals of computer programming.	[SLO: CS-07-D-01] Students will be able to explain how computers encode and decode computer programs (i.e. identification of decimal to binary and vice versa, conversion of texts, images and sounds in binary).	
[SLO: CS-06-D-02]Students will be able to analyze and apply basic programming constructs (e.g. sequence, selection, repetition, variables, inputs/events); by creating simple single-sprite, single-script programs using a visual programming tool. *[SLO: CS-06-D-Add]* **Additional SLO**: *Students will be able to apply basic*	[SLO: CS-07-D-02] Students will be able to apply fundamental programming constructs to create multi-sprite, multi-script programs using visual programming tools. *[SLO: CS-07-D-Add]* **Additional SLO**: *Students will be able to apply fundamental programming constructs to create multi-sprite,*	[SLO: CS-08-D-01] Students will be able to apply intermediate-level programming constructs (e.g. functions, cloning, conditional movement); by creating mini-games using a visual programming tool. *[SLO: CS-08-D-Add]* **Additional SLO**: *Students will be able to apply intermediate-level programming*

programming constructs (e.g. sequence, selection, repetition, variables, inputs/events); by creating simple single-sprite, single-script programs using textual programming tools.	*multi-script programs using textual programming tools.*	*constructs (e.g. functions, cloning, conditional movement); by creating mini-games using a textual programming tool.*
		[SLO: CS-08-D-Add] ***Additional SLO*** *Students will be able to analyze constructs and fundamentals of textual (syntax-based) programming.*

Curriculum Guidelines
Coding in Computer Science 6-8

DOMAIN *C*: Algorithmic Thinking and Problem Solving

STEP 1

Standard*:* Identity, define, and analyze a problem, and apply algorithmic thinking and problem-solving strategies to develop step-by-step solutions to solve problems.

Student Learning Outcome 1: Students will be able to identify, define and analyze a problem; and develop a step-by-step solution to solve simple problems.

Knowledge:
Students will be able to:

- Define and identify a problem.
- Analyze different techniques to deconstruct a problem.
- Differentiate between simple and complex problems.

Skills:
Students will be able to:

- Identify:
  - What is given – facts.
  - Data needed to solve the problem – input.
  - The output of the problem when given a certain input.
  - Specific instructions vs. nonspecific instructions.

Example: If a recipe is given, determine ingredients; given a maze, a robot, and a set of instructions the robot can follow; determine how to perform tasks using the given instructions; determine the task completed using given a set of instructions, etc.

- Deconstruct a problem into sub-problems (e.g. process of making fries, getting ready for school, etc.).
- Design a set of step-by-step instructions to solve a problem (e.g., giving directions using a specific set of words) through logic and reasoning.
- Integrate solutions to sub-problems to solve the main problem.
- Using given data and facts, reason about conclusion (guess the identity of a classmate given a set of qualities/features).

STEP 2

Formative Assessments

Assessment 1: Problem identification, analysis, and definition. Design a written quiz to identify and define the underlying problems in statements/situations etc.
Assessment 2: Problem-solving- Design quiz to apply all the problem-solving steps and choose the best solution out of all the available solutions; ask students to give reasons for the selection of a particular solution.
Assessment 3: Problem decomposition; Design a quiz with some simple and complex problems and ask students to differentiate between them. Ask students to deconstruct a complex problem into smaller parts.
Assessment 4: Step by step solution to a problem (Algorithm Designing by using Logic)- design a quiz to

write complete step-by-step instructions to solve any problem.

In-class or Homework Prompt: Ask students to write 3 problems they see around them and apply all the steps from problem identification to writing step-by-step solutions to the problem.
Students can design solutions to one problem and then can compare them with their classmates to find the best solution.
Student Self-Assessment/Reflection: Ask students to identify problems they are facing in school/ home or while coming to school/real-life/math/science and write them in a statement, find the problem solution and write down step by step solution for it.

Summative Assessments: (Theory & Practical)

- Term Assessment
- Mid-Year Exams
- Final Exams

STEP 3

Learning Activities (The activities below are neither listed in any particular order nor is this an exhaustive list. View them as recommendations).

Students can solve the following problems by breaking down the problem into smaller parts, and then suggesting a sequence of steps to reach a solution. Activities should be in groups of 4-6 students, and the output document should include a problem statement and a step-by-step solution.

- **Activity 1:** (Maze/map) to reach the target where none of the paths should be repeated.
- **Activity 2**: Use different blocks to make shapes.
- **Activity 3**: Create different objects by using Origami (Boat, house, doll, robot, airplane, etc.)
- **Activity 4:** Identify numbers of shapes in an image.
- **Activity 5:** Solve a Rubik's cube.
- **Activity 6:** Find the odd one out in the given objects.
- **Activity 7:** Arrange a birthday party/ family dinner/find the best path to school, etc.
- **Activity 8:** Solve a mathematical equation by deconstructing it into various smaller parts. (BODMAS, Finding LCM/HCF, solving algebraic equation)
- **Activity 9:** Design an activity to solve a big scientific problem by deconstructing it into various smaller parts. (Life cycles of living things, change in the state of matter, etc.)
- **Activity 10:** Design an activity like (draw the smiley/ draw the shape, etc.) to understand the concept of Logic; every student can have different logic and can follow different steps to solve the same problem.

STEP 1

Standard*:* Identity, define, and analyze a problem, and apply algorithmic thinking and problem-solving strategies to develop step-by-step solutions to solve problems.

Student Learning Outcome:2 Students will be able to analyze and apply basic algorithmic thinking to solve different types of problems

Knowledge: Students will be able to: • List benefits of algorithmic thinking. • recognize that algorithms are a sequence of precisely described instructions. • Examine that Algorithmic thinking is breaking down a problem, identifying important information, logical thinking, and confidence in decision making. • Define conditional statements/ selection statements that decide whether certain instructions should run (e.g. if there is rain take an umbrella)? • Identify loops, and analyze how they allow instructions to be repeated. • Analyze ways to solve a problem by using a combination of sequence, selection, and repetition.	**Skills:** Students will be able to: • Identify and differentiate between simple and complex problems. • Create a sequence of steps to solve a problem. • Relate sequence, selection, and repetition to daily life tasks. • Create solutions to problems using sequencing, loops, and conditions.

STEP 2

Formative Assessments

Assessment 1: Understand the algorithm as a step-by-step procedure. Design a written quiz to identify steps of a simple mathematical algorithm for adding two 3-digits integers.
Assessment 2: Design a worksheet in which they give multiple scenarios to determine which problem-solving technique (sequence, loops, and conditions) can be applied.
In-class or Homework Prompt: Ask students to write 3 problems around them and apply all the steps from problem identification to writing step-by-step solutions to the problem.
Students can design solutions to one problem and then can compare them with their classmates to find the best solution.
Student Self-Assessment/Reflection: Ask students to identify problems in their daily routines where they can apply sequencing, conditions, and loops to determine their solution (Eg. problem of traffic can be solved through traffic lights that run in loops).

Summative Assessments: (Theory & Practical)

- Term Assessment
- Mid-Year Exams
- Final Exams

STEP 3

Learning Activities (The activities below are neither listed in any particular order nor is this an exhaustive list. View them as recommendations. The learning activities below have been adopted from https://www.kodable.com/learn/learn-to-code-sequence/***)***

In-class activity 1: ALGORITHMS
The key learning is that activities should use algorithms or step-by-step processes to perform a task.
Students can work in groups:

- **Prompt 1**: Write an algorithm that would allow a person to create a pizza with 4 toppings.
- **Prompt 2**: Write an algorithm that gives the steps involved with brushing your teeth.

- **Prompt 3:** Write an algorithm to find a path in an 8x8 box with one starting point and ending point.

In class activity 2: SEQUENCE

The key learning is that activities should use a specific sequence to perform a task. Students can work in groups:

- **Prompt 1:** Apply sequencing using BODMAS to solve a mathematical problem.
- **Prompt 2:** 'Divide students into pairs such that one partner is the "coder" and one partner is the "robot". The coder decides on a simple task for their partner "robot" to do; the simpler the task, the better, like "walk across the room." Next, the coder gives their partner "robot" step-by-step instructions, also known as an algorithm, to complete the task. "Robots" need to remember that they can only do exactly what their coder tells them to do. If one of the steps is incorrect or not specific enough, this will result in a bug in their algorithm, and it will need to be redone. Switch places when finished' (Learning Activity, 2022).

In-class activity 3: REPETITION or LOOPS

The key takeaway is that activities should apply repetition or loops to solve a problem.

- **Prompt 1:** Ask the students to pick between creating a fun dance or exercise routine. Ask students to come up with three actions to perform physically e.g. for a workout routine, this could be a jumping jack, pushup, and sit-up, and for a dance routine, this could be a clap, spin, and jump. 'Have the students decide how many times each action should be repeated, or "looped". For example, "clap 2 times, spin 1 time, and jump 3 times". Perform the routine altogether, looping each action the designated number of times. Then, try looping the entire routine at least twice! As an extension, invite a partner to execute the same commands and perform the routine together!' (Flex Your Loop Skills, 2022).

In-class activity 4: CONDITIONS

The key takeaway is that activities should use conditions to solve a problem.

- **Prompt 1:** 'Come up with a rule – something that always happens in the same way. For example, "I go to school on Mondays". Then, come up with an exception, also known as a condition, to this rule. For example, "If it is a holiday, then I don't go to school on Monday." Write the conditional statement at the top of a piece of paper. Use the rest of the paper space to illustrate the conditional statement. In this example, you could draw a picture of your favourite holiday. Repeat this process 3 times so you have three conditional art pieces! If working in pairs, one partner can come up with the rules, and the other can come up with the condition and draw the image. Then, switch roles' ("Learn with Conditions", 2022).

- **Prompt 2:** 'Students will line up at one end of the classroom to reach the other side of the classroom. The teacher, and then the students themselves will call out conditionals and all the students will advance or not depending on the specific conditional statement. Have the students line up at one side of the room.
  Explain the rules:
  1. The object of the game is to get across the room first.
  2. For if...then conditionals: If the condition called out is true for you, then perform the action described in the then. If the condition called out is false for you, then do nothing.
  3. For if...then...else conditionals, listen carefully to the whole condition, as the else may apply to

you' ("Learn with Conditions", 2022).

DOMAIN D: Programming

Standard: Understand and apply fundamental programming constructs using visual and textual programming tools.	
Student Learning Outcome 1: Students will be able to analyze the fundamentals of computer programming.	
Knowledge: Students will be able to: • Determine the need for a programming language. • list ways in which programming is important in today's world. • Define 'Program'. • List different applications of computer programming. • Identify programming languages and their uses.	**Skills:** Students will be able to: • Differentiate between an algorithm and a program. • Convert an algorithm into a program.
Formative Assessments ***Student Reflection*** *– how many daily items use some sort of programming to work? What could be the benefits of learning programming?* **Summative Assessments: (Theory & Practical)** • Term Assessment • Mid-Year Exams • Final Exams	
Learning Activities (The activities below are neither listed in any particular order nor is this an exhaustive list. View them as recommendations) • **In-class or Homework Prompt** - Describe how various industries use programming (e.g. gaming industry, automobile industry, textile manufacturing industry, farming industry, etc.) • **Research project or Homework Prompt** – Students can work in groups in class or as a homework research assignment. The prompt "What kind of programming languages exist and what are their uses?". The takeaway is to understand that programming language exists for specific purposes (e.g. HTML for web development, python for data science, Unity for game design, etc.)	

Standard: Understand and apply fundamental programming constructs using visual and textual programming tools.

Student Learning Outcome 2: Students will be able to analyze and apply basic programming constructs (e.g. sequence, selection, repetition, variables, inputs/events); by creating simple single-sprite, single-script programs using visual programming tools.

Knowledge:
Students will be able to:

- List the fundamental programming constructs that enable a computer to interpret a computer program. The constructs are:
    - An event is an action from the user (or something outside the computer)
    - A sequence is a series of actions that are completed in a specific order. Computers follow all instructions in the sequence in which they are written.
    - A loop repeats instructions until a specific stopping condition is met.
    - A variable is used to store information called a value.
    - A conditional statement asks a question to figure out which path to take next.
- Analyze ways to debug a computer program.

Skills:
Students will be able to:

- Recognize that a program executes in a sequence – write computer programs accordingly.
- Write a program using the following constructs:
    - Event - take input from the user e.g. when the user presses an arrow key on the keyboard, or when the user clicks the mouse button – a certain action is executed.
    - Loop - repeats an instruction(s) a finite number of times or forever.
    - Motion - That involves the motion of a sprite/object.
    - Variable- creates and assigns values to a variable(s) (as per the event e.g. create an integer variable such as a score or a counter, and increase or decrease the value when certain events take place)
    - Conditional statement (if-else block) – running a block of code only if a specific condition is true
- Debug a computer program.

Formative Assessments

Quiz 1 – Definitions of fundamental constructs
Quiz 2 – Explaining a computer program (i.e. code) in natural language (i.e. human language) (e.g. When the green flag is clicked, change x by 100 can be explained as when the program starts the sprite will move right by 100), and converting natural language (i.e. human language) to a computer program (i.e. code).

Student Self-Assessment/Reflection:

- What are some programming constructs that we use in our daily lives?
- The importance of following a certain **sequence of instructions**. The sequence of instructions to perform a certain task correctly. (e.g. breaking the eggshell and then frying the egg will work, but frying the egg and then breaking the eggshell will not work).
- The use of **loops**. Which instructions do we repeat in our daily lives (e.g. pour water till the glass is full)
- The use of **conditional statements**. Which instructions do we follow under certain occasions (e.g. take an umbrella if it is raining or sunny)

Summative Assessments:

- Term Assessment
- Mid-Year Exams
- Final Exams

Learning Activities (The activities below are neither listed in any particular order nor is this an exhaustive list. View them as recommendations)

Students should code the following activities, and be encouraged to test their code, and debug accordingly if the program does not meet the required outcomes.

Lab Activity or In-class Coding Task #1 – EVENTS: When the program starts (e.g. in block-coding; using the event block "When green flag clicked"), make an action happen (e.g. make a sprite do something like change color when the sprite is clicked, or display some text like “hello world”)

Lab Activity or In-class Coding Task # 2 – SEQUENCE: When the program starts, two or three instructions should be executed in sequence (e.g. in block-coding; when the player clicks a sprite, make it go to a random position, play a sound, and switch costume in that order). Reflection question – what happens when the order is changed? Add a wait block (e.g. a wait one-second block or a print statement) between each instruction to make the changes more noticeable in a sequence.

Lab Activity or In-class Coding Task # 3 (continuation of task # 2) – LOOPS: When the program starts; give an instruction that will repeat forever (e.g. take the instructions given in task # 2 and set them to repeat)

Lab Activity or In-class Coding Task # 4: VARIABLES: Create an integer variable, assign a value to the variable at the start of the program, and then change the value in response to a certain input (e.g. create a score variable. When the program starts, the score should be set to zero, and when the player presses the up-arrow key the score should increase by 1, and the score should decrease by 1 when the down arrow key is pressed)

Lab Activity or In-class Coding Task # 5 (continuation of task # 4): Run an instruction only if a specific condition is true (for example when the variable score from task # 4 reaches a certain value, a sound should play)

National Curriculum of Pakistan
(2022-23)

Coding in Computer Science & Entrepreneurship

Grades 9 to 12

Domain B: Computational Thinking & Algorithms

Standard: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems*

Grade 9	Grade 10	Grade 11	Grade 12
Benchmark I: *Students will understand and apply computational thinking techniques to solve complex, real-world problems.*		**Benchmark I**: *Students have core concepts of basic data structures and algorithms used extensively in computer science and knowledge of how to apply these techniques toward solving more complex and real-life problems.*	
Student Learning Outcomes			
[SLO CS-09-B-01] Understand and apply techniques to decompose problems	*[SLO CS-10-B-01] Students will identify common algorithms used to develop software, store, search, or sort information*	*[SLO CS-11-B-01] Plan, develop, systematically test, and refine computational artifacts for problem-solving such as pseudocode, etc.*	*[SLO CS-12-B-01] Understand and evaluate the computational solutions in terms of efficiency, clarity, and correctness*
[SLO CS-09-B-02] Solve simple and complex problems computationally	*[SLO CS-10-B-02] Develop and apply abstractions to create generalized, modular solutions*	*[SLO CS-11-B-02] Apply common search, and sort algorithms*	*[SLO CS-12-B-02] Understand and apply complex algorithms on data structures such as trees and binary search*

Domain C: Programming Fundamentals

Standard: Students will create and debug projects in programming languages Python, HTML, and JavaScript, learning how to translate algorithms into code and define & apply fundamental programming constructs such as sequence, selection, and iteration

Grade 9	Grade 10	Grade 11	Grade 12
Benchmark I: *Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)*		**Benchmark I**: *Students will develop, test, debug command-line interface (CLI) applications in Python*	
Student Learning Outcomes			
[SLO CS-09-C-01] Students will understand web development and differentiate between a website and a web application	*[SLO CS-10-C-01] Students should be able to differentiate between front-end development, and back-end development of a website*	*[SLO CS-11-C-01] Students should understand the importance of computer programming and applications*	*[SLO CS-12-C-01] Students should be able to understand and evaluate applications of various programming paradigms.*
[SLO CS-09-C-02] Students should be able to create a static website using HTML/CSS in an appropriate environment *[SLO CS-09-C-03] Students should be able to create dynamic websites using JavaScript as the frontend scripting*	*[SLO CS-10-C-02] Students should be able to use more advanced HTML/CSS features in an appropriate environment* *[SLO CS-10-C-03] Students should be able to use more advanced programming constructs (lists, etc.) to create dynamic websites using JavaScript as backend scripting*	*[SLO CS-11-C-02] Students should be able to write and execute simple programs in Python.* *[SLO CS-11-C-03] Students should be able to draw shapes using Turtle Graphics functions in Python* *[SLO CS-11-C-04] Students should be able to understand the need for libraries and learn the use of some simple libraries in Python.*	*[SLO CS-12-C-02] Students should be able to use more advanced programming constructs such as data structures (lists etc.), file handling (disk IO to write to storage), and databases in Python.*
[SLO CS-09-C-04] Students should be able to implement common algorithms that use sequence, selection, and repetition in JavaScript	*[SLO CS-10-C-04] Students should be able to implement complex algorithms that use more complex data structures (lists, etc.) in JavaScript*	*[SLO CS-11-C-05] Students should be able to translate simple algorithms that use sequence and repetition in Python.* *[SLO CS-11-C-06] Students should be able*	*[SLO CS-12-C-04] Students should be able to implement complex algorithms that use lists etc. in Python*

		to decompose a problem into sub-problems and implement those sub-problems using functions in Python.	
[SLO CS-09-C-05] Students will determine ways of debugging their code in JavaScript	*[SLO CS-10-C-05] Students will determine more advanced techniques (unit tests, breakpoints, watches) for testing and debugging their code in JavaScript*	*[SLO CS-11-C-07] Students will determine ways of debugging their code in Python*	*[SLO CS-12-C-05] Students will determine more advanced techniques (unit tests, breakpoints, watches) for testing and debugging their code in Python*

Curriculum Guidelines

Coding in Computer Science & Entrepreneurship 9 - 12

Grade 9

Domain B: Computational Thinking & Algorithms

[SLO CS-09-B-01] & [SLO CS-09-B-02]

Standard 1: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Student Learning Outcomes
[SLO CS-09-B-01] *Understand and apply techniques to decompose problems*
[SLO CS-09-B-02] *Solve simple and complex problems computationally*

Knowledge:
Students will understand

1. The importance of computational thinking and problem-solving in computer science.
2. Principles of computational thinking:

a. Logical thinking
b. Algorithmic thinking

3. How to identify steps in identifying a computing problem
4. How to identify the inputs, processes, and outputs of a problem
5. Different methods to design and construct a solution to a simple problem, such as flow charts, and/or concept maps.
6. Steps to produce simple diagrams to show:
   . The structure of a problem
   a. Subsections and their links to other subsections

Skills:
Students will be able to

1. Explain the role of computational thinking in computer science.
2. Read and interpret simple computational problems
3. Apply computational thinking principles to define and refine problems
4. Identify the procedure appropriate to solving a problem.
5. Evaluate whether the order in which activities are undertaken will result in the required outcome.
6. Identify the inputs and outputs required in a solution.

Assessments
Formative Assessments
Quiz questions on the following topics:

- Defining & describing problem solving methods
- Solving computational problems using flow charts or concept maps

Summative Assessments
Exam question on the following topics:

- Explaining the role of computational thinking in computer science.
- Reading and interpreting simple computational problems
- Applying computational thinking principles to define and refine problems
- Identifying the procedure appropriate to solving a problem.
- Evaluate whether the order in which activities are undertaken will result in the required outcome.
- Identify the inputs and outputs required in a solution.

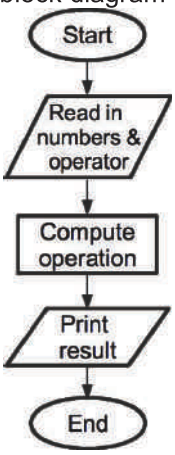
Standard 1: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Sample questions can include:

- Write logical solution steps & create concept map and flow chart solvng a word search puzzle, calculator progrma, or tic tac toe game

Learning Activities:

1. Describing computational thinking with examples and linking it with logical thinking.
2. Let students play simple games on the paper, for example, tic tac toe, connecting dots game or traditional such as connecting dots within a group. https://kidpillar.com/games-kids-think-critically-critical-thinking/
3.
4. Write Word search problem https://www.thesprucecrafts.com/free-word-searches-for-kids-1357174
5. Apply simple steps to solve problems such as tic tac toe.
6. Write the logical steps to find the treasure, and create concept map to find the solution https://www.javatpoint.com/the-wumpus-world-in-artificial-intelligence
7. Draw flowchart for calculator program using software like LARP
8. Activity: Spotting patterns helps programmers write better programs. https://abitofcs4fn.org/puzzles/pattern-matching-puzzles/caterpillar-puzzles-1/
1. Activity 2: Subjective questions with solving practical examples (ref Fig. 11. BCTt v.2 item example (item number 3) from https://www.semanticscholar.org/paper/Computational-Thinking-Test-for-Beginners%3A-Design-Zapata-C%C3%A1ceres-Mart%C3%ADn-Barroso/edf77fbe51d12bc76dcd5d4e7612bb047e676858
2. Make a cleaning robot: http://aimaterials.blogspot.com/p/blog-page_17.html
3. Ask students to make an algorithm for playing Tic Tac Toe
4. Critical thinking essay: https://www.studypool.com/discuss/26722703/critical-thinking-essasy
5. Importance of critical thinking & how to boost it: https://study.gov.pl/news/how-boost-your-critical-thinking
6. Computational thinking definition: https://digitalpromise.org/initiative/computational-thinking/computational-thinking-for-next-generation-science/what-is-computational-thinking/
7. Problem definition and Steps to solve an algorithm: https://ramahanishagunda.medium.com/a-search-algorithm-8233683c5d60
8. Draw a block diagram to solve calculator problem (simple example given below)



Domain C: Programming Fundamentals

[SLO CS-09-C-01]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-09-C-01] *Students will understand web development and differentiate between a website and a web application*

Knowledge:
Students will

- Grasp the basic concepts and principles of web development, including the technologies, languages, and frameworks commonly used in building websites and web applications.
- Understand the distinctive features of web applications. They should learn that web applications are dynamic in nature, offering interactive functionalities, data processing, user authentication, and real-time updates.
- Comprehend that websites primarily provide information and content to users, while web applications offer more complex functionality, such as user input, data manipulation, and task execution

Skills:
Students will be able to…

- Distinguish between front end and back end developement
- Determine which technology is appropriate for making static and dynamic web pages and web applications
- Identify and describe the features and characteristics of a website. This may include static content, informational pages, navigation menus, multimedia elements, and contact forms.
- Distinguish between websites and web applications based on specific criteria.

Assessments

Formative Assessments
Quiz / in class assignments on the following topics:

- **Differences between a website and a web application**
- **Role of different technologies like HTML, Javascript, web browser in web development**
- **Different between a static and dynamic web site**

Summative Assessments
A case study where students determine the best solution in terms of static/dynamic/interactive website and associated technologies for real world problems

Learning Activities

- Website vs. Web Application Examples: Divide students into small groups and provide them with a list of websites and web applications. Ask each group to analyze the features and functionalities of each example and categorize them as either a website or a web application. Afterward, have groups present their findings and discuss their reasoning behind the categorizations.
- Website vs. Web Application Showcase: Assign each student or group with the task of researching and finding a real-life example of a website and a web application. In class, have students showcase their findings, explaining the key characteristics and functionalities that differentiate them. Encourage discussions on the distinguishing factors and potential use cases for each

example.

- Design and Development Comparison: Provide students with a design mockup or wireframe of a website and a web application. In small groups, ask them to identify the elements and features that are unique to each type. Have them discuss and present their observations, emphasizing the user interactions, data processing, and dynamic functionality present in the web application design.

- Case Study Analysis: Provide students with a case study that involves a real-world website and web application. Ask them to examine the purpose, functionalities, and user experience of each. Have them compare and contrast the design choices, features, and development approaches used in the website and web application. Encourage critical thinking and discussions on the advantages and disadvantages of each approach.

- Potential textbook resources:
  - Learn to Program with Python 3, A Step-by-Step Guide to Programming, By Irv Kalb · 2018
  - Python Coding (Intermediate Level) For Kids
    Learn To Code Quickly With This Beginner's Guide To Computer Programming. Coding Projects in Python with Awesome Coding Activities, Games And More...
    By Tommy Harry Johnson · 2020
  - Python by Example
    Learning to Program in 150 Challenges
    By Nichola Lacey · 2019

[SLO CS-09-C-02]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)	
Student Learning Outcome(s): [SLO CS-09-C-02] Students should be able to create a static website using HTML/CSS in an appropriate environmen	
Knowledge: Students will know… • What is HTML • What is CSS • What is JavaScript Students will understand… • Characteristics, examples, similarities & differences between of static and dynamic website • The purpose of HTML, which is to display text and image content over the Internet • The capability of HTML documents to hyperlink to other documents • The structure of an HTML document including the underlying DOM tree (Document Object Model) • How JavaScript is used to modify a website	**Skills:** Students will be able to… • Differentiate between a static and dynamic website • Create a static website using HTML & CSS in an IDE like Visual Studio, Netbeans etc.

Assessments

Formative Assessments
Practical / in class exercises on the following topics:

- Share code snippets in class with bugs and ask students to point out the bug and how to fix it. Give students an assignment to make a webpage look a certain way (just share visual) and have them use CSS and HTML to create.
- What is the difference between static and dynamic websites? Illustrate with an example
- What does HTML mean and what is it used for?
- What is CSS and what is it used for?
- What is a web browser? Explain what happens when you type a website address in a browser?
- What are the main benefits of using Javascript?
- Inside which HTML element do we put the JavaScript?
- What is the correct syntax for referring to an external script called "xxx.js"?
- What will be the output of this code snippet? **console.log(String.raw`HelloTwitter\nworld`);**

Summative Assessments-
Exam / quiz on following topics:

- What is the correct CSS syntax for making all the elements bold?
- How do you add a comment in a CSS file?
- The # symbol specifies that the selector is?
- How can you open the link in a new window with HTML?
- Which of the tags below must be located in the <head> section of your page?

<title>
<form>
<link>
<meta>

- Which tag is used to create an ordered list?

Learning Activities

1. Stages of website development: https://www.newperspectivestudio.co.za/website-design-process-steps/
   https://medium.com/@rachelheo/p4-case-study-9bc802f51a98
2. Create breakout groups and assign a problem to solve using HTML, CSS and Javascript.
3. Student groups present their solutions back to class
4. Demo of various debugging and testing tools for websites and share their pros and cons (some examples: https://developers.google.com/search/help/debug)
5. Begins with showing the following two websites
   i. https://funtech.co.uk/latest/why-kids-should-learn-computer-science
   ii. Search some keyword on www.google.com
   Ask students what they can observe and then explain that (i) no change in data whenever you refresh/call the page, while, (ii) search results can vary with different keywords.
6. Assign students a website project to showcase their resume and portfolio of projects (coding projects or art etc) and have them build it using html, css, javascript. This can be a group or individual activity
7. In class assignment to use forms to build a survey website
8. How HTML is used to display content such as dividing content into divisions, different heading styles. [https://www.w3schools.com/html/]
9. The capability of HTML documents to hyperlink to other documents
   https://www.w3.org/TR/html401/struct/links.html
10. The structure of an HTML document including the underlying DOM tree (Document Object Model)
    https://www.w3.org/TR/WD-DOM/introduction.html
11. Adding pictures

https://www.w3schools.com/html/html_images.asp
12. How CSS is used to style displayed content using fonts, alignment and colors https://www.w3schools.com/css/
13. Start with defining website and share the story of first website https://www.home.cern/science/computing/birth-web
14. What Can Kids Do With HTML? https://www.codewizardshq.com/html-for-kids/]
15. code in javascript within html https://www.w3schools.com/html/html_scripts.asp
16. The structure of an HTML document including the underlying DOM tree (Document Object Model) https://www.w3.org/TR/WD-DOM/introduction.html
17. Explain the difference between static website and dynamic website https://blog.hubspot.com/website/static-vs-dynamic-website

[SLO CS-09-C-03]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-09-C-03] Students should be able to create dynamic websites using JavaScript as the frontend scripting

Knowledge:
Students will understand…

- Variables, Data Types and Arrays in JavaScript
- Variable assignment in JavaScript
- What aspects of HTML can be changed with JavaScript
- What aspects of CSS can be changed with JavaScript

Skills:
Students will be able to…

- Use JavaScript to modify an HTML website to create a dynamic website

Assessments

Formative Assessments
Quiz / in class assignments / practicals on the following topics:

- Variables, data types and arrays in JavaScript
- HTML aspects that can be changed with JavaScript
- CSS aspects that can be changed with JavaScript

Sample projects can include:

- Create a magic 8 ball game in JavaScript
- Create a todo list using an appropriate programming language
- Create a drum kit using JavaScript

Summative Assessments
Project submission / practical / Exam questions related to:

-
- Using JavaScript to modify the HTML to create a dynamic website

Sample questions can include;
- What is dynamic typing?
- What are some basic data types in Javascript? Given an example usage of each
- Write statement to create an array of 3 elements of different data types
- Build a clock using Javascript and HTML

Learning Activities

1. Students can watch a video on how to make a website dynamic using JavaScript: https://www.youtube.com/watch?v=MkV_x-x79UI
2. In-class project build a dynamic website using JavaScript:

a. Sample 1: https://medium.com/@pearlmcphee/build-a-dynamic-app-using-javascript-html-and-css-f0dfc136007a
b. Sample 2: https://www.htmlgoodies.com/javascript/creating-dynamic-websites-using-javascript/

3. Additional reading on concepts:

. https://www.udacity.com/blog/2021/06/javascript-hub.html
a. Variables, Data Types and Arrays in JavaScript and printing values in console, https://www.edureka.co/blog/data-types-in-javascript/
b. Variable assignment in JavaScript https://www.w3schools.com/js/js_variables.asp
c. What aspects of HTML can be changed with JavaScript: https://www.w3schools.com/js/js_htmldom_html.asp
d. What aspects of CSS can be changed with JavaScript: https://www.w3schools.com/js/js_htmldom_css.asp

[SLO CS-09-C-04]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-09-C-04] Students should be able to implement common algorithms that use sequence, selection, and repetition in JavaScript

Knowledge:
Students will understand…
- An algorithm is a set of instructions
- How the sequence of instructions affects the result
- Sequence, Selection and Repetition in JavaScript
- Arrays in JavaScript

Skills:
Students will be able to…
- Construct an array and populate its values using a loop in JavaScript

Assessments

Formative Assessments
In-class assignments on computing problems that should be solved using JavaScript

Summative Assessments

Practical / lab exercise programming small algorithms in JavaScript
Practical / lab project creating a programming project in JavaScript

Learning Activities

1. Fun mini-applications that are great for beginners and use all the programming fundamentals:
   - https://hackr.io/blog/javascript-projects
   - https://skillcrush.com/blog/projects-you-can-do-with-javascript/
2. Start with using JavaScript in IDE (VS Code or Visual Studio or online interpreter) without HTML https://linuxhint.com/javascript-visual-studio-code/
3. Sequence, Selection and Repetition in JavaScript http://students.cs.ucl.ac.uk/schoolslab/projects/HT6/cooking/HT6/JavaScript_Seq_Sel_Itr.htmlLink algorithm thinking in here from previous SLOs
4. Arrays in JavaScript https://www.w3schools.com/js/js_arrays.asp
5. Loops in javascript https://www.w3schools.com/js/js_loop_for.asp

[SLO CS-09-C-05]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-09-C-05] *Students will determine ways of debugging their code in JavaScript*

Knowledge:
Students will understand…

- Code written outside of a function is hard to test
- Code written inside a function can be tested
- That they can write code that calls functions to ensure the results are correct
- Using a debugger allows programmers to set a breakpoint to stop execution of their code to see the state of variables mid-execution for the purpose of discovering errors in their code

Skills:
Students will be able to…

- Write code to invoke functions and check their return values for correctness
- Set a breakpoint to debug mistakes in their code

Assessments

Formative Assessments
Practical / lab exercises on identifying & resolving errors in computing problems using test functions and breakpoints.

Summative Assessments
Quiz / exam / practical on computing problems with errors and students to identify errors using test functions and breakpoints.

Learning Activities

- Basic workflow debugging for JavaScript programming (instructors can follow video and steps on this page: https://developer.chrome.com/docs/devtools/javascript/) **to use breakpoints**

Grade 10
Domain B: Computational Thinking & Algorithms
[SLO CS-10-B-01]

Standard 1: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Student Learning Outcomes [SLO CS-10-B-01] Students will identify common algorithms used to develop software, store, search, or sort information

Knowledge:
Students will understand

1. How to solve the counting problems

a. Basics of a counting problem
b. Basic counting principles
   i. Multiplication
   ii. Addition
   iii. Permutation
   iv. Combination
   v. The pigeonhole principle
   vi. Inclusion and exclusion principle.

2.
3. Properties of Algorithm
   i) Input
   ii) Output
   iii) Definiteness
   iv) Finiteness
   v) Effectiveness
   v) Generality

4. Logical reasoning and will be able to solve

a. Boolean logic
b. Verbal logical reasoning
c. Nonverbal logical reasoning

5. Understand data search and sort, and briefly describe standard algorithms on linear arrays such as linear search, binary search, insertion sort, bubble sortetc.

Skills:
Students will be able to

1. Apply logical reasoning to refine and solve problems
2. Apply algorithmic thinking to refine and solve problems
3. Identify when & where to use key search & sort algorithms
4. Discuss an algorithm to solve a specific problem.

Assessments

Formative & **Summative Assessments**
Assessment questions on:

- Boolean logic, verbal logical reasoning, nonverbal logical reasoning
- Algorithmic thinking and will be able to solve problems by abstraction, decomposition, pattern recognition, and algorithms

Subjective questions / / inclass discussions on

- Applying logical reasoning to refine and solve problems
- Applying algorithmic thinking to refine and solve problems

Learning Activities

1. Homework: Watch video and summaries one of the algorithms mentioned here in your own words: https://www.youtube.com/watch?v=d7iGniWrRng
2. Course activities in https://www.khanacademy.org/computing/computer-science/algorithms

a. Guessing game: https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/a-guessing-game
b. Route finding: https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/route-finding
c. Algorithms in your life: https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/discuss-algorithms-in-your-life

3. Consider a grading system where numbers are turned into letters. Activity details are here: http://ndrdmath.weebly.com/lesson-21.html
4. Verbal reasoning questions: https://irp-cdn.multiscreensite.com/16712cd9/files/uploaded/CGP%20NVR%20GL%20ALL.pdf
5. Write algorithm to solve word search problem and Sudoku game: https://medium.com/@george.seif94/solving-sudoku-using-a-simple-search-algorithm-3ac44857fee8
6. Common algorithms: https://u.osu.edu/cstutorials/2016/11/21/7-algorithms-and-data-structures-every-programmer-must-know/
7. Add boolean expressions to conditional statements (e.g. exercise 5.1: https://www.sccollege.edu/Departments/upwardboundms/SiteAssets/Pages/Computer-Science-/chapter5and6.pdf)
8. Searching & Sorting Algorithms Practice: https://www.101computing.net/searching-sorting-algorithms-practice/

[SLO CS-10-B-02]

Standard 1:*Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems*

Student Learning Outcomes [SLO CS-10-B-02] Develop and apply abstractions to create generalized, modular solutions

Knowledge:
Students will understand
1. Steps in an algorithm to solve computational problems
2. Dry running or Trace Table to run algorithm
3. Identify logical and syntax errors
4. Abstractions to create generalized, modular solutions

Skills:
Students will be able to
- Use algorithmic approach to solve the computational simple problems
- Apply abstractions to create generalized, modular solutions
- Create and use dry runs/trace tables to follow an algorithm
- Identify syntax/logic errors in code and solve logical errors

Assessments

Formative Assessments
Quiz assessment questions / inclass discussions on:
- Steps in an algorithm to solve computational problems
- Dry running or Trace Table to run algorithm
- Identification of logical and syntax errors
- Abstractions to create generalized, modular solutions

Sample question could be:
- Find the smallest integer in the list of 4 1 -4 0 9 9 3 5 8
- Write an algorithm to find and display the square and cube of a positive number. The execution must be terminated, if a negative number is entered.
- Evaluate answers for some simple programs
- Find logical error in the some simple list of steps

Summative Assessments
Exam questions/project submissions on topics such as:

- Algorithmic approach to solve the computational simple problems
- Abstractions to create generalized, modular solutions
- Creating and using dry runs/trace tables to follow an algorithm
- Identifying syntax/logic errors in code and solve logical errors

Learning Activities

1. Activity 3 Develop a high-level algorithm from this link: http://sofia.cs.vt.edu/cs1114-ebooklet/chapter4.html

a. Students should write steps for an activity, e.g. “send a birthday card to Mark”. Steps could be:
Go to a store that sells greeting cards
Select a card
Purchase a card
Mail the card

b. The instructor can then drill down questions that show that the instructions could be more specific:
"Which store will you visit?"
"How will I get there: walk, drive, ride my bicycle, take the bus?"
"What kind of card does Mark like: humorous, sentimental, risqué?"

2. Video on specific instructions and developing algorithms. Class can watch this video and try to create their own set of instructions
https://www.youtube.com/watch?v=Ct-lOOUqmyY

Domain C: Programming Fundamentals
[SLO CS-10-C-01]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-10-C-01] Students should be able to differentiate between front-end development, and back-end development of a website

Knowledge:
Students will understand…

- Back-end development allows writing code that emits HTML/CSS/JavaScript
- Front-end development deals with HTML/CSS/JavaScriptin the browser

Skills:
Students will be able to…

- Differentiate between back-end and front-end development

Assessments

Formative Assessments

- In Class activities / practical / quiz on differentiating between back-end and front-end development

Summative Assessments

- Exam / practical

Learning Activities

- Inspect webpage to discover code of HTML/CSS/JavaScript
- Write a webpage using HTML/CSS/JavaScript
- Differentiate between back-end and front-end development

[SLO CS-10-C-02]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-10-C-02] Students should be able to use more advanced HTML/CSS features in an appropriate environment

Knowledge:
Students will understand…

- How HTML tags can be used to show tabular date
- How HTML can be used to retrieve inputs from users
- How to apply animation movements to HTML components

Skills:
Students will be able to…

- Create forms in HTML using an IDE like Visual Studio, Netbeans etc.
- Create tables in HTML using an IDE like Visual Studio, Netbeans etc.
- Create animations in CSS using an IDE like Visual Studio, Netbeans etc.

Assessments

Formative Assessments

In class activities / practicals on the following topics:

- Create forms in HTML
- Create tables in HTML
- Create animations in CSS
    - Draw a circle with CSS

Summative Assessments
Exam/quiz on

- How HTML tags can be used to show tabular date
- How HTML can be used to retrieve inputs from users
- How to apply animation movements to HTML components

Learning Activities

- In class / practical HTML Forms activity on this dynamic website: https://www.w3schools.com/html/html_forms.asp
- Populate form in HTML
- Populate tables in HTML
- Simple animation using CSS, i.e, change colour of button, translate a text horizontally from left to write.
- Develop three web pages, i.e., main_page.html, animation.html, and forms.html. Link all these pages and run it in the browser
- Write logical steps (algorithm) of translating text diagonally from top left corner to the bottom right

corner
- In class activity / practical: Students can test out and try to recreate various CSS animation listed here: https://blog.hubspot.com/website/css-animation-examples

[SLO CS-10-C-03]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-10-C-03] Students should be able to use more advanced programming constructs (lists, etc.) to create dynamic websites using JavaScript as backend scripting

Knowledge:
Students will understand…
- How to create and use arrays in JavaScript
- How to create and use bullet points in HTML

Skills:
Students will be able to…
- Create bullet points in HTML that are generated from an array in JavaScript

Assessments
-
Formative Assessments
In class activities / practicals on the following topics
- How to create and use Arrays in JavaScript
- How to create and use bullet points in HTML

Summative Assessments
In class activities / practicals / project which includes the following component:
- Create bullet points in HTML that are generated from an array in JavaScript

Learning Activities
1. Make ordered and unordered lists in HTML, sample activities and code here: https://www.freecodecamp.org/news/html-list-how-to-use-bullet-points-ordered-and-unordered-lists/

[SLO CS-10-C-04]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-10-C-04] Students should be able to implement complex algorithms that use more complex data structures (lists, etc.) in JavaScript

Knowledge:
Students will understand…
- The array data structure is similar to a list
- Finding an element in a list requires iterating through entire list till the element is found

Skills:
Students will be able to…
- Write an algorithm that finds an element in a list
- Implement an algorithm that finds an element in a list using JavaScript

Assessments

-

Formative Assessments

Quiz on the following topics:

- The array data structure is similar to a list
- Finding an element in a list requires iterating through entire list till the element is found

Summative Assessments

Quiz/exam / practical on the following topics:
Identify algorithms that finds an element in a list
Implement an algorithm that finds an element in a list using JavaScript

Learning Activities

1. In class activity / practical: Students can test out code for search algorithms here https://www.freecodecamp.org/news/4-methods-to-search-an-array/
2. JavaScript drills: https://jsbeginners.com/javascript-projects-for-beginners/
3. Starter projects in JavaScript that use lists: https://hackr.io/blog/javascript-projects
4. Calculator: https://github.com/harsh98trivedi/Simple-JavaScript-Calculator
5. Alarm clock in JS: https://github.com/swasti98/JS-Clock

[SLO CS-10-C-05]

Standard: Students will develop, test, and debug static website (using HTML and CSS) and a dynamic website (using JavaScript)

Student Learning Outcome(s):
[SLO CS-10-C-05] Students will determine more advanced techniques (unit tests, breakpoints, watches) for testing and debugging their code in JavaScript

Knowledge:
Students will understand…

- The purpose of a unit test
- Debugging allows them to analyze code as it runs

Skills:
Students will be able to…

- Write simple unit tests for the functions in their code
- Set a breakpoint and use it to analyze intermediate values of variables in JavaScript

Assessments

Formative Assessments

Practical / lab exercises on identifying & resolving errors in computing problems using test functions and breakpoints.

Summative Assessments

Quiz / exam / practical on computing problems with errors and students to identify errors using test functions and breakpoints.

Learning Activities

- Basic workflow debugging for JavaScript programming (instructors can follow video and steps on this page: https://developer.chrome.com/docs/devtools/javascript/) **to use breakpoints and watch expressions.**

Grade 11

Domain B: Computational Thinking & Algorithms

[SLO CS-11-B-01]

Standard 2: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Student Learning Outcomes [SLO CS-11-B-01] Plan, develop, systematically test, and refine computational artifacts for problem-solving such as pseudocode, etc.

Knowledge:
Students will understand

1. How to use different methods to design and construct a solution to a computational problem

Skills:
Students will be able to

1. Create psuedocode to address computational problems in the correct font, size, style, indentation, case, line numbers, comments, data type key words, variable assignments & declarations, common operators, and key commands
2.
3. Systematically test computational artefacts
4. Analyse an algorithm presented as a flow chart in terms of include tracing an algorithm as well as assessing its correctness.
5. Evaluate algorithms in terms of their efficiency, correctness, and clarity

Assessments

Formative Assessments
Quiz questions on the following topics:

1. What is the difference between pseudocode and algorithm?
2. Write a pseudocode to output odd integers from 0 to 100?
3. Write an algorithm to “rock, paper, scissor” game

Summative Assessments
Practical activities on topics such as:

1. Write down the algorithm to solve tic tac toe game

Learning Activities

1. Interpreting word problems in code: https://docs.google.com/document/d/1_Hu-ZJz2p4dyYzyYgjZeORWuLX8PVfcFn7i8MZpPFHM/edit
2. Write a Pseudo-code?
   - Arrange the sequence of tasks and write the pseudocode accordingly.
   - Start with the statement of a pseudo code which establishes the main goal or the aim. Example: **If case is 1 then print “I am case 1” else print “I am not case 1”**
3. Reinforcement on the previous topics i.e., algorithmic approaches with new examples

[SLO CS-11-B-02]

Standard 2: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Student Learning Outcomes [SLO CS-11-B-02] Apply common search, and sort algorithms

Knowledge:
Students will understand

1. Problem solving methods using simple example of

a. Abstraction
b. Decomposition
c. Pattern recognition

2. Algorithmic approaches to solve practical exercises of algorithms
3.
4. When to use various search and sort algorithms such as linear search, binary search, insertion sort, bubble sort, etc.

Skills:
Students will be able to

1. Use and adapt classic algorithms to solve computational problems (e.g. sorting and searching algorithms such as linear search, binary search, insertion sort, bubble sort, etc.)

Assessments

Formative Assessments
Quiz questions on the following topics:

- Define problem solving methods?
- What are the definitions and applications of various searching & sorting algorithms?

Practical exercises such as:

- Apply algorithmic thinking to use algorithm approaches (Abstraction, Decomposition, Pattern recognition) to handle complex problems such as:
- Sort following array {1, 5, 3, 2}
- Search for the number 5 in {2, 1, 5, 3, 2, 4, 5}
- Word search problems

Summative Assessments
Practical exercises for applying algorithmic thinking to create search & sort algorithms

Learning Activities:

1. Algorithmic approaches
   https://junilearning.com/blog/guide/how-to-introduce-computational-thinking-to-kids/
   https://www.teachingexpertise.com/classroom-ideas/algorithmic-games/
2. Write pseudocode for solving simple word problems, sample exercises here:
https://olevelcomputerscience.files.wordpress.com/2015/09/pseducode.pdf

Domain C: Programming Fundamentals
[SLO CS-11-C-01]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python	
Student Learning Outcome(s): [SLO CS-11-C-01] Students should understand the importance of computer programming and applications	
Knowledge: Students will understand… • Programs use the basic components of a computer to take inputs, process the input, and produce output • The Agile and Waterfall are models of the Software Development Lifecycle and are used to gather requirements and implement software	**Skills:** Students will be able to… • Take a real world problem, propose a software solution and implement it.
Assessments - **Formative Assessments** Quiz questions on the following topics: • Basic components of a computer program • Agile and Waterfall processes in gathering software requirements **Summative Assessments** Practical exercises such as: • Apply an agile / waterfall models to gather software requirements • Take a real world problem, propose a software solution and implement it.	
Learning Activities 1. Summarize or present in groups differences between Agile & Waterfall processes. Sample analysis is here: https://www.forbes.com/advisor/business/agile-vs-waterfall-methodology/ 2. Watch a video on gathering requirements in the software development process, example: https://www.pearson.com/channels/product-management/learn/Mariya/4-requirements-gathering-and-maintenance/42-scope-management-in-agile-vs-waterfall	

[SLO CS-11-C-02]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python	
Student Learning Outcome(s): [SLO CS-11-C-02] Students should be able to write and execute simple programs in Python.	
Knowledge: Students will understand… • What is Python, why is it used • What type of problems can be solved using Python • Input/ Output handling • Variables in Python • Operators in Python • Sequence, Selection, Repetition in Python	**Skills:** Students will be able to… • Write and execute a program in Python using an IDE like replit.com (online) VS Code (offline) that uses variables, sequence, selection and repetition

Assessments
-Quiz questions on the following topics:

- Variables in Python
- Sequence, Selection, Repetition in Python

Summative Assessments
Practical exercises such as:

- Write and execute a program in Python that uses variables, sequence, selection and repetition

Learning Activities

- Instructor can request students to code geginner projects in Python, examples here: https://www.dataquest.io/blog/python-projects-for-beginners/
  - https://www.freecodecamp.org/news/python-projects-for-beginners/
  - https://www.upgrad.com/blog/python-projects-ideas-topics-beginners/
- Add suggested books
- Add links to suggested compilers

[SLO CS-11-C-03]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python

Student Learning Outcome(s):
[SLO CS-11-C-03] Students should be able to draw shapes using Turtle Graphics functions in Python

Knowledge:
Students will understand…

- How to use the Python Turtle Library
  - Turtle methods
  - Methods of screen
  - Turtle motion
  - Use of events
  - Create/ draw shapes
  - Compound Shapes
- How to create shapes by means of instructions to a "turtle" to move in a given direction
- How to create more complex shapes by allowing the "turtle" to lift the pen while moving

Skills:
Students will be able to…

- Write and execute a program in Python to create complex shapes using the Turtle library

Assessments
-
Formative Assessments
Quiz questions on the following topics:

- How to create shapes by means of instructions to a "turtle" to move in a given direction
- How to create more complex shapes by allowing the "turtle" to lift the pen while moving

Summative Assessments
Practical exercises such as:

- Write and execute a program in Python to create complex shapes using the Turtle library

Learning Activities

1. In-class Exercises using Turtle library, example exercises:

a. https://realpython.com/beginners-guide-python-turtle/
b. https://www.vivaxsolutions.com/web/python-turtle.aspx

[SLO CS-11-C-04]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python	
Student Learning Outcome(s): [SLO CS-11-C-04] Students should be able to understand the need for libraries and learn the use of some simple libraries in Python.	
Knowledge: Students will understand… • The concept of abstraction allows the use of complex libraries without knowing their internal implementation	**Skills:** Students will be able to… • Find and use a third party Python library that is simple to use but has a complex implementation
Assessments - **Formative Assessments** Practicals / exercises on: • Importing & using libraries • Find and use a third party Python library that is simple to use but has a complex implementation **Summative Assessments** • Practical exercises that have an element of using a library or designing your own library	
Learning Activities 1. In-class research project, what is a library and why are they used? https://careerfoundry.com/en/blog/web-development/programming-library-guide/ 2. Top 30 Python libraries: https://www.mygreatlearning.com/blog/open-source-python-libraries/ 3. In-class video on how libraries work: https://www.youtube.com/watch?v=4oXc3EpUN4E	

[SLO CS-11-C-05]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python	
Student Learning Outcome(s): [SLO CS-11-C-05] Students should be able to translate simple algorithms that use sequence and repetition in Python.	
Knowledge: Students will understand… • What are variables, sequence, repetition, and lists in Python • How to use sequence and repetition to manipulate lists in Python	**Skills:** Students will be able to… • Write and execute a Python program that uses variables, sequence and repetition to populate a list • Write and execute a Python program that uses variables, sequence and repetition to find an element in a list
Assessments -	

Formative Assessments
Practical exercises such as:

- Use sequence and repetition to manipulate lists in Python
- Write and execute a Python program that uses variables, sequence and repetition to populate a list

Summative Assessments
Practical exercises such as: Write and execute a program in Python that uses variables, sequence, selection and repetition

Learning Activities

1. Writing python script (https://www.w3schools.com/python/)
2. Sequence, Selection, Repetition in Python (https://austincode.com/itse1359/sequence-selection-repetition.php)

[SLO CS-11-C-06]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python

Student Learning Outcome(s):
[SLO CS-11-C-06] Students should be able to decompose a problem into sub-problems and implement those sub-problems using functions in Python

Knowledge:
Students will understand…

- Why we need functions
- How to decompose a large problem into sub-problems
- How to identify duplication in their code
- How to move duplicated code into a function
- How to create/ define /invoke a function
- Types of Functions
- Function parameters/ arguments
- Scope of variables
- Returning value from a function
- Pass by value

Skills:
Students will be able to…

- Write and execute a Python program that solves a large problem by decomposing into sub problems.
- Write a Python program that invokes functions within loops
- Write a Python program that performs some mathematical operation on a value passed to it, and returns the updated value (for example celsius to Farienheit conversion etc.)

Assessments
-
Formative Assessments
Quiz questions / Practical exercises such as:

- How to write a function?
- Why do we need a function?
- How a function is called in the program?
- What is the scope of function?
- What is the scope of variable

Summative Assessments
Practical exercises such as: Write a functions add, subtract, multiply and divide and call them in the script to perform calculations

Learning Activities

1. Revise functions
2. Establish importance of functions
3. Different terminologies to the functions
4. Define scope of variables and functions with examples
5. Define “naming conventions”, such as CamalCascadeing, camelCascading and so on.
6. Writing a function in python
7. Converting a duplicate code into a function
8. Calling a function
9. Function types
10. Passing parameters in function
11. Return type of a function
12. Passing by value
13. Practical examples

[SLO CS-11-C-07]

Standard: Students will develop, test and debug command-line interface (CLI) applications in Python

Student Learning Outcome(s):
[SLO CS-11-C-07] Students will determine ways of debugging their code in Python

Knowledge:
Students will understand…

- Code written outside of a function is hard to test
- Code written inside a function can be tested
- That they can write code that calls functions to ensure the results are correct
- Using a debugger allows programmers to set a breakpoint to stop execution of their code to see the state of variables mid-execution for the purpose of discovering errors in their code

Skills:
Students will be able to…

- Write code to invoke functions and check their return values for correctness
- Read through code and dry-run by hand to find bugs

Assessments

-

Formative Assessments
Practical / lab exercises on identifying & resolving errors in computing problems using test functions and dry-run.

Summative Assessments
Quiz / exam / practical on computing problems with errors and students to identify errors using test functions and breakpoints.

Learning Activities

- Class activity for variable dry-run in Python, lesson plan here: https://teachinglondoncomputing.files.wordpress.com/2014/05/activity-assignmentdryrunpython.pdf

Grade 12
Domain B: Computational Thinking & Algorithms
[SLO CS-12-B-01]

Standard 2: *Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.*

Student Learning Outcomes [SLO CS-12-B-01] *Identify and apply complex algorithms on data structures such as trees and binary search*

Knowledge:
Students will know that

1. Define data structures such as lists, arrays, trees stack, and queue
2. How to traverse a tree

a. In-order Traversal
b. Pre-order Traversal
c. Post-order Traversal

3. How to conduct a binary search
4. Application of tree data structure
5. Application of binary search algorithm

Skills:
Students will be able to

1. Identify and apply tree algorithm
2. Traverse trees
3. Explain binary search method and predict results of a binary search algorithm

Assessments

Formative Assessments

- Quiz questions on topics related to case studies of trees and students to solve/traverse them

Summative Assessments

- Examination questions on identifying and applying trees algorithm

Learning Activities

-
- Tree traversal: https://www.tutorialspoint.com/data_structures_algorithms/tree_traversal.htm
- Binary search videos & online exercises: https://www.khanacademy.org/computing/computer-science/algorithms/binary-search/a/binary-search
- Inclass activity / practical on binary search: https://classroom.thenational.academy/lessons/binary-search-chjked
- Sample lesson plan on binary search including activities like Raffle Tickets, Ping Pong Ball and Guess My Number to learn about binary trees: http://csunplugged.mines.edu/Activities/BinarySearch/BinarySearch.pdf

Domain C: Programming Fundamentals
[SLO CS-12-C-01]

Standard: Students will develop, test, debug, and document command-line interface (CLI) applications in Python

Student Learning Outcome(s):
[SLO CS-12-C-01] Students should be able to understand and evaluate applications of various programming paradigms.

Knowledge: Students will understand… • The purpose of programming language paradigms is to reduce complexity and make code easy to understand for programmers • A high level view and pros/cons of Object Oriented Programming • A high level view and pros/cons of functional programming	**Skills:** Students will be able to… • Write a Python program using Object Oriented Programming to define a class with instance attributes to manage states
Assessments - **Formative Assessments** Projects / practicals / in-class activities on developing an artefact in the Python programming language that will use object oriented programming to manage states **Summative Assessments** Exam questions should include explanations of purpose of programming, function programming, object oriented programming and it's advantages	
Learning Activities 1. In-class activity: Watch a video on OOP & Python and discuss key points, e.g. https://youtu.be/E40NqsDgYa4 2. Python OOP exercises: https://pynative.com/python-object-oriented-programming-oop-exercise/	

[SLO CS-12-C-02]

Standard: Students will develop, test, debug, and document command-line interface (CLI) applications in Python	
Student Learning Outcome(s): [SLO CS-12-C-02] Students should be able to use more advanced programming constructs such as data structures (lists etc.), file handling (disk IO to write to storage), and databases in Python.	
Knowledge: Students will understand… 1. The purpose of a list is to store an ordered list of elements 2. What is importance of disk I/O? 3. File handling methods and file operations (Create, Read and Write, etc) 4. The purpose of a dictionary is to store key-value pairs 5. Given a key, finding a value in a dictionary is faster than in a list 6. What is the use of database 7. Introduction to database (MS Access, MySQL, etc) 8. Data normalization up to third form 9. Primary key, secondary key, etc 10. How to connect databases with python programming	**Skills:** Students will be able to… • Write and execute programs to create and add/remove items in a list in Python • Construct and retrieve values from a dictionary in Python • (Advanced) Write and execute a Python program that can construct a dictionary based on user input, and can print a value whose key is input from the user. • Write and execute programs that create and writes to the file • Read existing files • Create and manage database in Python • Create & update tables • Data normalization up to third form • Select from tables (How to add, delete and edit records) • Select with filter (where statement) • Sort results (Order by statement) • (Advanced) Create GUI based programs using Pygame or Tkinter

Assessments

-

Formative Assessments

Quiz questions on topics such as:

- The purpose of a list
- The purpose of a dictionary
- Finding values in a dictionary vs. list

Summative Assessments

Exam questions on following topics:

- Constructing dictionaries in Python
- Retrieving values from a dictionary in Python
- Writing and executing a Python program that can construct a dictionary based on user input, and can print a value whose key is input from the user.

Learning Activities

1. Mini-drills to learn about dictionaries in Python: https://towardsdatascience.com/12-examples-to-master-python-dictionaries-5a8bcd688c6d
2. In-class activity / practicals: Python projects that use dictionaries, e.g. https://favtutor.com/blog-details/7-Python-Projects-For-Beginners
3. Create a mini-game using dictionaries, e.g. Hangman: https://favtutor.com/blog-details/7-Python-Projects-For-Beginners

[SLO CS-12-C-03]

Standard: Students will develop, test, debug, and document command-line interface (CLI) applications in Python

Student Learning Outcome(s):

[SLO CS-12-C-03] *Students should be able to implement complex algorithms that use lists etc. in Python*

Knowledge:	**Skills:**
Students will understand… • The concept of a nested list (list within a list) • A list as a value within a dictionary	Students will be able to… • Write, execute and debug a Python program that reads a text file from disk and prints the number of occurrences of each letter of the alphabet

Assessments

-

Formative Assessments

Quiz questions on topics such as:

- Nested lists (list within a list)
- A list as a value within a dictionary

Summative Assessments

Examination questions / practicals on writing, executing and debugging Python programs that read a text file from disk and prints the number of occurrences of each letter of the alphabet

Learning Activities

1. Count the number of times a letter appears in a text file in Python https://www.geeksforgeeks.org/count-the-number-of-times-a-letter-appears-in-a-text-file-in-python/
2. https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/Python_FileText.html

[SLO CS-12-C-04]

<table>
<tr><td colspan="2">Standard: Students will develop, test, and debug Python code</td></tr>
<tr><td colspan="2">Student Learning Outcome(s):
[SLO CS-12-C-04] Students will determine more advanced techniques (unit tests, breakpoints, watches) for testing and debugging their code in Python</td></tr>
<tr><td>Knowledge:
Students will understand…
The purpose of a unit test
Debugging allows programmers to analyze code as it runs</td><td>Skills:
Students will be able to…
Write a unit tests for the functions in their code
Use a print statement to help debug bugs in their code</td></tr>
<tr><td colspan="2">Assessments

Formative Assessments
Practical / lab exercises on identifying & resolving errors in computing problems using test functions and breakpoints.

Summative Assessments
Quiz / exam / practical on computing problems with errors and students to identify errors using test functions and breakpoints.</td></tr>
<tr><td colspan="2">Learning Activities
Basic workflow debugging for Python programming using print statements (example step by step guide here: https://www.codementor.io/@allisonf/how-to-debug-python-code-beginners-print-line-du107ltvx)
Instructors can use this YouTube video as a reference: https://www.youtube.com/watch?v=r0JvqH6OWKQ</td></tr>
</table>

National Curriculum Council Secretariat
Ministry of Federal Education and Professional Training, Islamabad
Government of Pakistan