# Module 2 Problem solving in computer programming

2.1 **Problem solving process and concepts**

Content:

* Problem solving
* Application of problem-solving constructs
* Developing solutions

*Learning Outcomes:*

*Students should be able to:*

2.1.1 Define the term problem solving

2.1.2 Define the term computational thinking

2.1.3 Describe the phases of the PLDC (Program Development Life Cycle)

2.1.4 Describe the purpose of problem solving leading to solutions

2.1.5 Explain and apply various problem-solving steps. Polya, G., 1957) (Range:Understand the problem (task/problem description or scenario/user stories) State in own words Clarity on what needs to be done

What is known or given?

What is missing or needed?

Devise a plan/algorithm (storyboard – visual or textual)

Look for patterns

Look at related problems, known solutions

Examine simpler or special cases

Make a table, create diagram, use guess and check, work backwards, identify sub-goal

Carry out the plan/implement the algorithm (write the code)

Look back/test (see if it works)

Check results against original problem. Does it make sense? Is there another solution?)

2.1.6 Use appropriate tools and techniques to present a solution. Range:

User stories (written by the client and provide the requirements)

Noun-verb analysis of user stories

List of nouns provides identification of objects and state

List of verbs provides identification of behaviour

Acceptance tests (does the program meet the requirements?)

2.2 **Construct an algorithm and present a solution to a given problem**

Content:

* Problem solving
* Algorithm design
* Flowcharts

*Learning Outcomes:*

*Students should be able to:*

2.2.1 Define the term algorithm and its purpose in the problem-solving process. (Range: Basic concepts of an algorithm. What is an algorithm? Develop a clear understanding of the problem presented.)

2.2.2 Implement and understand the basic algorithmic constructs used to create a **flowchart.** Range: Input, Output, Processing and Calculations, Selection Iteration

2.2.3 Create a flowchart to present a particular algorithm and its associated tasks

2.2.4 Interpret a basic flow chart and describe its intended operation / function

# Activity 1.1 FORMATIVE ASSESSMENT

1.1.1 Define the term system unit? (2)

1.1.1 A system unit is the part of a computer that houses the primary devices that perform operations and produce results for complex calculations. ✓ ✓ (2)

1.1.2 What is the purpose of CPU? (2)

1.1.2 The purpose of the CPU is to carry out the set of instructions given to the processor from a program. ✓ ✓ (2)

1.1.3 Identify the FIVE stages of information processing cycle. (5)

1.1.3 Information Processing Cycle Stages

1. Fetch- Instruction from Memory (Instruction Fetch, IF) ✓
2. Decode the instructions into binary (Instruction Decode, ID) ✓
3. Execute action and move to next step or calculate address (EXE) ✓
4. Access memory operand (MEM)✓
5. Write back result to register (WB) ✓

1.1.4 The rate at which one operation is completed in a second is measured in \_\_\_\_\_\_\_\_. (1)

1.1.4 The rate at which one operation is completed in a second is measured in hertz. ✓ (1)

1.1.5 Identify THREE main components of the CPU.

1.1.5 THREE main components of the CPU.

* Control Unit✓
* Arithmetic Logic Unit✓
* Registers✓

1.1.6 List **THREE** differences between Reduced Instruction Set Computing (RISC) and Complex Instruction Set Computing (CISC).

1.1.6 Differences between RISC and CISC

* In RISC, the instruction set is reduced, and most of these instructions are very primitive, while in CISC, the instruction set is very large that can be used for complex operations. ✓
* RISC computer’s execution time is very less, whereas CISC computer’s execution time is very high. ✓
* In RISC, the decoding of instructions is simple, whereas, in CISC, the decoding of instructions is complex. ✓

**Activity 1.2 FORMATIVE ASSESSMENT**

**Activity 1.3 FORMATIVE ASSESSMENT**

* + 1. Define memory hierarchy? (2)

1.3.1 The memory hierarchy is an improvement of computer storage into a hierarchy-based modal on response time. ✓✓

1.3.2 Identify the two types of RAM. (2)

1.3.2 Static Random Access Memory and Dynamic Random Access Memory. ✓✓ (2)

1.3.3 What do you understand by the term secondary memory. (2)

1.3.3 Secondary memory is computer memory that is non-volatile and persistent in nature and is not directly accessed by a computer/processor. ✓✓ (2)

1.3.4 List THREE examples each of input, processing and output hardware devices. (3)

1.3.4 Hardware Devices

|  |  |  |
| --- | --- | --- |
| **Input devices** | **Processing devices** | **Output Devices** |
| Mouse✓ | Motherboard✓ | Monitor/Screen✓ |
| Keyboard✓ | Arithmetic Logic Unit✓ | Printer✓ |
| Light pen✓ | Processor✓ | Plotter✓ |

# Activity 1.4 SUMMATIVE ASSESSMENT

1.4.1 Differentiate SRAM and DRAM. (4)

1.4.1SRAM: is a memory chip that is faster and uses less power than DRAM✓✓. DRAM: is a memory chip that can hold more data than an SRAM chip, but it requires more power. ✓✓ (4)

1.4.2 List and describe **FIVE** factors to consider when choosing an input device. (10)

1.4.2 FIVE factors to consider when choosing an input device.

1. **User Needs**- this is the urgency of use of the device in the computer room by users also whether it will satisfy the needs of the user. ✓✓
2. **Initial cost-**the amount it can cost when buying/purchasing the devices✓✓
3. **Maintenance Cost**-the amount that can be used to maintain the servicing of these devises should be considered. ✓✓
4. **Mode Of Transmission**- how will you transport them to the computer room should be considered✓✓
5. **Compatibility With Available Hardware**- will the devices fit to other devices in the room already. A device that fits your needs but doesn't work with your computer is useless. ✓✓
6. **User- Friendliness**- whether the devices will be used to solve problems and easy to be used by users✓✓

**ANY FIVE CORRECT**

1.4.3 Define the term software. (2)

1.4.3 Software – set of instructions, data or programs used to operate computers and execute specific tasks. ✓✓ (2)

1.4.4 Identify THREE types of language translators. (3)

1.4.4 THREE types of language translators (3)

Translators✓

Compilers✓

Assemblers✓

1.4.5 Differentiate a compiler from an interpreter. (4)

1.4.5 Compiler-A compiler is a computer program that transforms whole code written in a high-level programming language at once into the machine code✓✓ whereas an interpreter is a computer program, which converts each high-level program statement into the machine code line by line. ✓✓ (4)

1.4.6 Define the following terms:

* Freeware
* Middleware
* Shareware
* Open Source (8)

1.4.6

**Freeware-** Freeware is the software that is available to use for free of cost without any limitations. ✓✓

**Middleware**- is software that is used to bridge the gap between applications and other tools or databases. Some examples of middleware activities include handling data and [API](https://www.techtarget.com/searchapparchitecture/definition/application-program-interface-API) management, authentication and messaging services. ✓✓

**Shareware**- The software is copyrighted and distributed for free only for testing purposes. After the trial period ends, you must pay. ✓✓

**Open source-** This is provided for use, modification, and redistribution. Open-source software is downloaded from the internet at no cost. ✓✓

# Activity 2.1 FORMATIVE ASSESSMENT

2.1.1 What is meant by software testing? (2)

2.1.1 *Software Testing* is a method to check whether the actual software product matches expected requirements and to ensure that software product is[Defect](https://www.guru99.com/defect-management-process.html)free. ✓ ✓ (2)

2.1.2 Identify and explain 4 software testing phases in software development. (8)

2.1.2 **Software Testing Phases**

* **Integration testing**- This brings together two or more application modules to make sure they work together. .✓ ✓
* **Unit testing**- Unit testing is typically done throughout the application development process and its goal is to make sure that every single unit or component works as planned. .✓ ✓
* **Regression testing**- This determines if adding additional features results in a decrease in an application's functionality. .✓ ✓
* **Acceptance testing**- a testing technique performed to determine whether or not the software system has met the requirement specification.✓ ✓

# Activity 2.2 FORMATIVE ASSESSMENT

2.2.1 Define the term problem solving as applied in software development. (2)

2.2.1 Problem Solving is the sequential process of analysing information related to a given situation and generating appropriate response options.✓ ✓ (2)

2.2.2 What is meant by the term computational thinking? (2)

2.2.2 Computational thinking is an interrelated set of skills and practices for solving complex problems, a way to learn topics in many disciplines, and a necessity for fully participating in a computational world. .✓ ✓ (2)

2.2.3 Identify FOUR cornerstones of computational thinking. (4)

2.2.3 **FOUR** cornerstones of computational thinking are:

* **decomposition**
* **pattern recognition**
* **abstraction**
* **algorithms**

2.2.4 List SIX phases of program development life cycle? (6)

2.2.4 Six phases of program development life cycle. (6)

1. Problem Identification✓
2. Designing the solution✓
3. Coding the program✓
4. Testing and debugging✓
5. Implementation✓
6. Review and Maintenance✓

2.2.5 Write an algorithm that reads three numbers and prints the value of the largest number. (10)

2.2.5 Algorithm

Step 1: Input N1, N2, N3

Step 2: if (N1>N2) then

if (N1>N3) then

MAX =N1 [N1>N2, N1>N3]

else

MAX N3 [N3>N1>N2]

endif

else

if (N2>N3) then

MAX =N2 [N2>N1, N2>N3]

else

MAX =N3 [N3>N2>N1]

endif

endif

Step 3: Print “The largest number is”, MAX

**Activity 2.3 SUMMATIVE ACTIVITY**

2.3 Explain FOUR cornerstones of computational thinking. (8)

**2.3 FOUR** cornerstones of computational thinking are:

* **decomposition** - breaking down a complex problem or system into smaller, more manageable parts
* **pattern recognition** – looking for similarities among and within problems
* **abstraction** – focusing on the important information only, ignoring irrelevant detail
* **algorithms** - developing a step-by-step solution to the problem, or the rules to follow to solve the problem (8)

2.4 Consider the following scenario.

Tickets are sold for a concert at $20 each, if 10 tickets are bought then the discount is 10%, if 20 tickets are bought the discount is 20%. No more than 25 tickets can be bought in a single transaction. Design a flowchart to depict the above scenario. (10)

2.4 Flowchart



2.5 What is a conditional flowchart? (2)

2.5 . A conditional flowchart is a design technique used when a condition is imposed on a problem. (2)