Unit no 07: Computational Thinking

Short Answer Questions:

1. **Define computational thinking.**
   Computational thinking (CT) is a problem-solving process that involves a set of skills and techniques to solve complex problems in a way that can be executed by a computer. This approach can be used in various fields beyond computer science, such as biology, mathematics, and even daily life.

2. **What is decomposition in computational thinking?**
   Decomposition is an important step in computational thinking. It involves dividing a complex problem into smaller, manageable tasks. Let's take an example of building a birdhouse. This task might look tough at first, but if we break it down, we can handle each part one at a time.

3. **Explain pattern recognition with an example.**
   Pattern recognition involves looking for similarities or patterns among and within problems. For instance, if you notice that you always forget your homework on Mondays, you might recognize a pattern and set a reminder specifically for Sundays. Pattern recognition is an essential aspect of computational thinking.

4. **Describe abstraction and its importance in problem-solving.**
   Abstraction is a fundamental concept in problem solving, especially in computer science. It involves simplifying complex problems by breaking them down into smaller, more manageable parts, and focusing only on the essential details while ignoring the unnecessary ones.

5. **What is an algorithm?**
   An algorithm is a step-by-step collection of instructions to solve a problem or complete a task similar to following a recipe to bake a cake.

6. **How does problem understanding help in computational thinking?**
   Understanding the problem involves identifying the core issue, defining the requirements, and setting the objectives. Understanding the problem is the first and most important step in problem-solving especially in computational thinking.

7. **What are flowcharts and how are they used?**
   Flowchart is a diagrammatic representation of an algorithm. It describes what operations are required to solve a given problem. Flowcharts illustrate the sequence of operations to be performed to solve a problem in the form of a diagram.

8. **Explain the purpose of pseudocode.**
   Pseudocode is a method of representing an algorithm using simple and informal language that is easy to understand. It combines the structure of programming clarity with the readability of plain English, making it a useful tool for planning and explaining algorithms.

9. **How do you differentiate between flowcharts and pseudocode?**
   Flowcharts and pseudocode are both tools used to describe algorithms, but they do so in different ways.

Pseudocode	Flowcharts
● Pseudocode uses plain language and structured format to describe the steps of an algorithm. ● It is read like a story, with each step is written out sequentially. ● Pseudocode communicates the steps in a detailed, narrative -like format. ● It is particularly useful for documenting algorithms in a way that can be easily converted into actual code in any programming language.	● Flowcharts use graphical symbols and arrows to represent the flow of an algorithm. ● It is like watching a movie, where each symbol (such as rectangles, diamonds, and ovals) represents a different type of action or decision, and arrows indicate the connection and direction of the flow. ● Flowchart communicates the process in a visual format, which can be more intuitive for understanding the overall flow and structure. ● They are useful for identifying the steps and decisions in an algorithm at a glance.

10. What is a dry run and why is it important?

A **dry run** is the process of manually going through the steps of a program, algorithm, or process without actually executing the code on a computer. It involves simulating the execution, usually on paper or a whiteboard, to see how the logic works and what outputs are produced for given inputs. Helps identify logical or runtime errors before actual execution.

11. Describe LARP and its significance in learning algorithms.

LARP stands for Logic of Algorithms for resolution of Problems. It is a fun and interactive way to learn how algorithms work by actually running them and seeing the results. Think of it as a playground where you can experiment with different algorithms and understand how they process data.

12. List and explain two debugging techniques.

Debugging is the process of finding and fixing errors in an algorithm or flowchart. Here are some common debugging techniques:

- **Trace the steps:**

Go through each step of your algorithm or flowchart to see identify where it goes wrong.

- **Use Comments:**

Write comments or notes in your algorithm to explain what each part is supported to do. This can help you spot mistakes.