Lesson 1: Computational Thinking Techniques

In this lesson, students will learn about computational thinking techniques, including decomposition, abstraction, and algorithmic thinking. They will understand how these techniques are used to solve complex problems in computer science and other fields. Through guided and independent practice activities, students will apply these techniques to real-world scenarios, developing their problem-solving skills. The lesson will culminate with an exit ticket and quiz to assess students' understanding of the concepts. By the end of the lesson, students will be able to effectively decompose problems, identify essential details, and develop step-by-step plans to solve problems using computational thinking techniques.

## **Objectives:**

- Define the terms decomposition, abstraction, and algorithmic thinking.

- Recognize scenarios where each of these computational thinking techniques are applied.

- Apply decomposition, abstraction, and algorithmic thinking to solve a problem.

## **Materials:**

- Whiteboard or blackboard

- Markers or chalk

- Handouts with problem-solving scenarios

- Pen/pencil and paper for each student

## **Bell-Ringer Activity:**

1. Display the following question on the board: "What do you understand by the term 'computational thinking'?"

2. Give students 5 minutes to write down their responses individually.

3. After 5 minutes, ask a few students to share their answers with the class.

## **Introduction:**

1. Explain to the students that computational thinking is a problem-solving approach used in computer science and other fields.

2. Define the terms decomposition, abstraction, and algorithmic thinking:

- Decomposition: Breaking down a complex problem into smaller, more manageable parts.

- Abstraction: Identifying the essential details and ignoring unnecessary details to focus on the problem's core.

- Algorithmic Thinking: Developing a step-by-step plan or set of instructions to solve a problem.

3. Discuss the importance of these techniques in problem-solving and their applications in various fields, including computer science, mathematics, and everyday life.

## **Direct Instruction:**

1. Explain decomposition in detail:

- Provide examples of how decomposition can be used to solve complex problems.

- Discuss how breaking down a problem into smaller parts makes it easier to understand and solve.

- Emphasize the importance of identifying the relationships between the smaller parts and the whole problem.

2. Explain abstraction in detail:

- Give examples of how abstraction can be used to simplify complex problems.

- Discuss how focusing on the essential details helps in understanding and solving the problem efficiently.

- Explain the concept of layers of abstraction and how they can be used to manage complexity.

3. Explain algorithmic thinking in detail:

- Define an algorithm as a step-by-step plan or set of instructions to solve a problem.

- Discuss the importance of developing algorithms in problem-solving.

- Provide examples of how algorithms can be used in different scenarios.

## **Guided Practice:**

1. Divide the students into small groups.

2. Distribute handouts with problem-solving scenarios to each group.

3. Instruct the groups to apply decomposition, abstraction, and algorithmic thinking to solve the given problems.

4. Circulate among the groups, providing guidance and support as needed.

5. After a designated time, ask each group to present their solutions and explain how they applied the computational thinking techniques.

## **Independent Practice:**

1. Provide each student with a problem-solving scenario.

2. Instruct the students to individually apply decomposition, abstraction, and algorithmic thinking to solve the problem.

3. Allow the students sufficient time to work on their solutions.

4. Collect the students' solutions for assessment.

## **Exit Ticket:**

1. Distribute exit tickets to each student.

2. Ask the students to briefly explain the difference between decomposition, abstraction, and algorithmic thinking.

3. Collect the exit tickets for assessment.

## **Closure:**

1. Review the key concepts of decomposition, abstraction, and algorithmic thinking.

2. Discuss the importance of these computational thinking techniques in problem-solving.

3. Encourage the students to apply these techniques in their future problem-solving endeavors, both in computer science and other subjects.

## **Common Core Standards:**

- CCSS.ELA-LITERACY.RST.9-10.3: Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks, attending to special cases or exceptions defined in the text.

- CCSS.ELA-LITERACY.RST.9-10.4: Determine the meaning of symbols, key terms, and other domain-specific words and phrases as they are used in a specific scientific or technical context relevant to grades 9-10 texts and topics.