DAY 10

1. What is a Decision-Making Statement?

A **decision-making statement** is a control flow statement that allows you to execute or skip a block of code based on a specified condition. The condition is always a boolean expression, meaning it evaluates to either true or false.

Real-time use case:

Think about an ATM. The program uses decision-making statements to:

- **Check your PIN:** if the PIN is correct, allow access.
- **Check your balance:** if the withdrawal amount is less than or equal to your balance, dispense the cash; else, show an error.

---

2. Types of Decision-Making Statements in Java

Java offers several statements for making decisions, each suited for different scenarios.

a) if Statement

The if statement is the simplest form. It executes a block of code only if a condition is true.

b) if-else Statement

This statement provides two options. If the condition is true, the if block is executed; otherwise, the else block is executed.

c) if-else if-else Statement

This is used for handling multiple conditions. It checks the first if condition. If it's false, it moves to the next else if, and so on. If all conditions are false, the final else block is executed.

d) Nested if Statements

This involves placing an if statement inside another if or else block. It's used when you need to check a condition only after another condition has already been met.

---

3. Syntax and Hands-on Examples

if Statement

The code inside the curly braces {} will only run if the condition is true.

Java

```
int score = 85;

if (score > 80) {

   System.out.println("You got an A!");

}
```

if-else Statement

This guarantees that one of the two blocks of code will always be executed.

Java

```
int temperature = 25;
if (temperature > 30) {
   System.out.println("It's a hot day.");
} else {
   System.out.println("It's a pleasant day.");
}
```

if-else if-else Statement

This is a robust way to handle multiple outcomes.

Java

```
int grade = 75;
if (grade >= 90) {
   System.out.println("Excellent!");
} else if (grade >= 70) {
   System.out.println("Good!");
} else {
   System.out.println("Keep trying!");
}
```

Nested if Statements

This is useful for multi-level conditions.

Java

```
boolean hasLicense = true;
int age = 20;
if (age >= 18) {
   if (hasLicense) {
      System.out.println("You can drive.");
   } else {
      System.out.println("You are old enough, but need a license.");
   }
} else {
```

```
    System.out.println("You are not old enough to drive.");
}
```

---

4. Interview Questions

1. **What is a decision-making statement, and what is its purpose?**
    - **Expected Answer:** It's a control flow statement that executes code based on a boolean condition. Its purpose is to allow programs to make decisions.
2. **Explain the difference between an if statement and an if-else statement.**
    - **Expected Answer:** An if statement only executes a block of code if the condition is true. An if-else statement provides an alternative block of code to execute if the condition is false.
3. **When would you use an if-else if-else structure instead of multiple if statements?**
    - **Expected Answer:** Use if-else if-else when the conditions are mutually exclusive (only one can be true). This is more efficient because the compiler stops checking conditions once one is met.
4. **What is a nested if statement? Give a simple example.**
    - **Expected Answer:** An if statement placed inside another if or else block. An example could be checking a username and then checking the password.
5. **What data type must the condition inside an if statement evaluate to?**
    - **Expected Answer:** The condition must evaluate to a boolean value (true or false).