1.Type vs.Variable : A *type* refers to the classification of data , for example:int or String. Defining the operations that can be performed it takes in memory. A *variable* is an instance of a type, serving as a named location in memory where data of that will be stored.

2.Role of Archetypes in Java Development: In the context of Maven, are project templates. It helps in setting up a new Java project according to a predefined template.

3.Instance Variable vs. Local Variable: An *instance variable* is one of a class, with each object of the class having its own copy of the variable. It represents the attributes of an object and can be accessed throughout the class . A *local variable* is declared inside a method or block, existing only during the inside of that method or block, and it can't be accessible outside of it.

4.Array vs. ArrayList: An array is a fixed-size collection of elements of the same type. Its size won't change after being created. An A*rrayList*, part of the collections framework, is a dynamic-size collection of objects. It can be modified after created, such as methods for adding, removing, and searching elements.

5.How “switch” Statement Works: The switch statement allows for the conditional execution of code blocks based on the value of an expression. It compares the expression's value against multiple values and executes the block of code corresponding to the first matching case. If there are no matches, a default block can be executed.

6.Difference between “for” and “while” Loops: A for loop is used when the number of iterations is known before entering the loop, with the starting, condition, and iteration steps that set in the loop statement. A while loop is when the number of iterations is not known beforehand, executing its block of code as long as the condition remains true.

7. x =4. x is initialized to 5. the first if condition (x < 5) is false, therefore ,x++ won't executed, and x -= 1 is executed unconditionally in fragments due to the lack of braces, decreasing x to 4 after the if or else block is evaluated. In the second if condition (x >= 5), since x is now 4, the condition is false, and x *= 2 is not executed so the final value of x is 4.

8. x=4. This is because the only thing that applied to x is a decrement by 1 due to the unconditional execution of x -= 1, and the conditions for incrementing x or doubling its value are not met based on the if statements conditions.