Q1) HashCode and Equals

HashCode and Equals:
HashCode:

->In Java, hashCode() is a method of the Object class that returns a hash code value for the object.
->It is used for hashing-based data structures like HashMaps.
->It is essential for efficient retrieval of objects from collections.
Equals:

->equals() is a method used for comparing the equality of two objects.
->The default equals() method in the Object class compares object references.
->It is often overridden in custom classes to compare the contents of objects

Example

```
public class Person {
    private String name;
    private int age;

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Person person = (Person) obj;
        return age == person.age && Objects.equals(name, person.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name, age);
    }
}
```

Q2)Control Statements
Control Statements:
Control statements in Java include:

if-else: Conditional statement.
switch: Multi-branch selection statement.
for, while, do-while: Looping statements.
break and continue: Alter the normal flow of control.

Q3)Looping In Java

-> For Loop

```
for (int i = 0; i < 5; i++) {
    System.out.println(i);
}
```

->While Loop

```
int i = 0;
while (i < 5) {
    System.out.println(i);
    i++;
}
```

->Do-While Loop

```
int i = 0;
do {
    System.out.println(i);
    i++;
} while (i < 5);
```

Q4) Is-a and Has-a Relationship:

Is-a Relationship (Inheritance):

It represents a generalization-specialization relationship between classes.

Example: Car is a Vehicle

Code

```
class Vehicle { /* ... */ }
class Car extends Vehicle { /* ... */ }
```

.

Has-a Relationship (Composition):

It represents a relationship where one class has another class as a part.

Example: Car has an Engine.

Code

```
class Engine { /* ... */ }
class Car {
    private Engine engine;
    // Other attributes and methods
}
```

Q5)Java Record

Java Record:

A Java record is a feature introduced in Java 14 for creating simple classes primarily used to store immutable data. It automatically generates useful methods like equals(), hashCode(), and toString().

Code
```
record Point(int x, int y) {
    // Automatically generates constructor, getters, equals, hashCode, toString
}

// Usage
Point p1 = new Point(1, 2);
Point p2 = new Point(1, 2);

System.out.println(p1.equals(p2));  // true
```

Q6)Java Enum
Java Enum:
An enum in Java is a special data type used to define a set of constant values. Enums are often used to represent things like days of the week, months, etc.

Code
```
enum Day {
    MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY
}

// Usage
Day today = Day.MONDAY;
```

Q7)Stack Memory and Heap Memory

Stack Memory vs Heap Memory:
Stack Memory:

Used for storing local variables and method call information.
Memory is managed in a last-in, first-out (LIFO) fashion.
Generally smaller in size compared to heap memory.
Memory is automatically reclaimed when the method execution completes.

Heap Memory:
Used for dynamic memory allocation, such as objects and arrays.
Objects created in heap memory persist beyond the scope of the method that created them.
Memory management is more complex, involving garbage collection to reclaim unused memory.
Generally larger in size compared to stack memory.