**NAME Misbah Uddin**

**Roll NO 448**

**Section G.**

**Method Overloading:**

Method overloading in Java means you can have multiple methods in a class with the same name but different types or numbers of parameters. This lets you use the same method name for different tasks depending on what information you give it. For example, you could have a method called "add" that can add two numbers or three numbers, just by giving it different inputs. Example of method overloading:

class Calculator {

// Method to subtract two integers

int subtract(int a, int b) {

return a - b;

}

// Method to multiply two integers

int multiply(int a, int b) {

return a \*b;

}

}

Method overloading is important because it lets you use the same method name for different jobs depending on what information you give it. This makes your code flexible because one method can do various tasks. It also makes your code easier to read and manage because similar tasks are grouped together under the same name. Plus, it means you don't have to come up with complicated names for your methods, making your code easier to understand.

**2. Method Overriding:**

Method overriding in Java happens when a child class gives its own version of a method that was already defined in its parent class. The child class's method has the same name, what it returns, and what it needs as the parent class's method. When you use an object of the child class to call this method, it uses the child class's version instead of the parent class's. Example of method overriding:

class Shape {

public void draw() {

System.out.println("Drawing a shape");

}

}

Significance of method overriding:

-

Method overriding in Java is important because it allows you to use a reference variable of a parent class to point to an object of its child class and use the child class's method. This helps in reusing code because child classes can change or add to the behavior of methods inherited from parent classes. It also lets you decide which method to use based on the actual type of the object when the code is running, which is called runtime polymorphism.