1. List down some of Java methods where method overloading is used.

-String.format(): This method can be overloaded to format a string in different ways, depending on the arguments that are passed to it. For example, you can use String.format() to format a string with a specific number of decimal places, or to format a string with a specific date format.

-Math.pow(): This method can be overloaded to calculate the power of a number in different ways, depending on the types of the arguments that are passed to it. For example, you can use Math.pow() to calculate the power of a number to an integer, or to calculate the power of a number to a floating-point number.

-System.out.println(): This method can be overloaded to print different types of data to the console, depending on the arguments that are passed to it. For example, you can use System.out.println() to print a string, an integer, or a floating-point number to the console.

-Object.equals(): This method can be overloaded to compare two objects in different ways, depending on the types of the objects that are being compared. For example, you can use Object.equals() to compare two strings, two integers, or two objects of a user-defined class

2.Overload method by changing the data type of the argument and keeping the same number of arguments.

```
public class OverloadedMethods

{


    public void displayValue(int num)

    {

        System.out.println("Integer number: " + num);

    }


    public void displayValue(double num)

    {

        System.out.println("Double number: " + num);

    }

}
```

3. Calculate the area of Circle, Triangle, Rectangle using method overloading.

```
public class AreaCalculator
{

    public double calculateArea(double radius)
    {
        return Math.PI * radius * radius; // Area of Circle
    }

    public double calculateArea(double base, double height)
    {
        return 0.5 * base * height; // Area of Triangle
    }

    public double calculateArea(double length, double width)
    {
        return length * width; // Area of Rectangle
    }
}
```

4. What is the advantage of method overloading?

The advantages of method overloading in Java include:

a) Readability and Maintainability: Method overloading allows you to use the same method name for similar operations, making the code more readable and easier to maintain.

b) Flexibility: It provides flexibility to developers in choosing the appropriate method based on the number and types of arguments, enhancing code reusability.

c) No need for unique names: Without method overloading, you would have to come up with unique names for similar operations, leading to long and confusing method names.

d) Polymorphism: Method overloading is a form of compile-time polymorphism, which is resolved during compilation based on the method's signature.

e) Code Organization: Overloaded methods enable you to organize related operations under the same method name, improving the structure of your code.