BCA Semester: SEM-5
Roll Number: --
Student Name: Jat Kishanbhai Laxmanbhai
Subject: Java Programming
Subject Code: BCA-501

BCA - 501: Java Programming

1. Explain method overloading with example.

Method Overloading: Creating method with same name but with different number of parameters, parameter type or return type this situation is known as Method Overloading in Java.

Mostly that methods working similar task with different parameter.

When we call a method by an object, Java matches the name of method first and then check number and type of parameters to decide witch one of the definitions to execute. This is part of *polymorphism* specifically *compile-time polymorphism*

To create an overloaded method, we need to create method with same name but declare with different number or type of parameters.

In other hand the definition of method can be same or different, in some cases like add two int and return int, as well as with same definition add two double and return double here both have same definition `return a + b`.

This example show that some time method definition can be same but that always be known as method overloading

```
class Calc {
  int sum(int a, int b) {
    return a + b;
  }

  double sum(double a, double b) {
    return a + b
  }
}
```

Example : with different number of parameters

InfoPrinter.java

```
class InfoPrinter {
  public void printInfo(String name) {
    System.out.println("Name: " + name);
  }

  public void printInfo(String name, String address) {
    printInfo(name);
    System.out.println("Address: " + address);
  }

  public void printInfo(String name, String address, String phone_number) {
    printInfo(name, address);
    System.out.println("Phone no.: " + phone_number);
  }
}

public class MethodNotSame {
  public static void main(String[] args) {
    InfoPrinter my_info = new InfoPrinter();
    my_info.printInfo("Computer Programmer", "127.0.0.1", "1111111111");
  }
}
```

In this simple example of method overloading we see that in `InfoPrinter` class hava three methods with different number of parameters

1. first have only one parameter `name`
2. second have `name` and `address`
3. third have `name` , `address` and `phone_number`

In this example main class `MethodNotSame` , this program calling third method of `InfoPrinter` that have three parameters

it's call `void printInfo(String name, String address, String phone_number);`

2. What is package? How to create a package?

One of the main features of OOP is ability to reuse the code already created and structure the program code.

For normal reuseblity is using one class and extend that class to another new class for using first class code

But when we need to use code that is not in that existing file or making sort code file for *simplicity* with more *functionality*, that time we need to use package.

Package : Package is way of *grouping multiple classes and interfaces* together. package is one type of *container* that contain classes and interfaces with *relational manner*.

Benefit :

- simplicity - code become more simple to understand and maintain
- reusability - code can be reuse in any another file where that code is not written.
- more functionality - depth functional operation done with out any complexity
- abstraction & encapsulation - no need to understand all code that use in program execution, and remove complexity of the code

Java API Packages :

Java API provides a large number of packages according to functionality. Most of time we use the builtin packages thay are available in Java system packages

Packages that we already used in *Java system packages* like use math builtin method. That `math` package is access from java built in package library.

Example : Java API Packages

```
Java
 |- lang -- Language support classes (primitive types, string, math function, thread,
 |          exception)
 |- util -- Language utility classes (vectors, hash table, random number, date)
 |- io   -- Input/Output support classes. for I/O data (file operation)
 |- awt  -- Createing graphical inteface (window, button, list, menu)
 |- net  -- Network operation, network communication, internet server
 |- applet -- For creating and emplementing applet
```

We can say that package is sub-set of *library* and that contains number of classes and interfaces that have meaningful connection with that package name.

We can create package like above `math` package, we can create *self defined package* for our code *simplecity* and *reusability*.

How to create a package : Best structured way to create self defined packages.

Use Java IDE or Extension in code editor for package creation support.

1. Create package structure: Create package folder where your main Java program file is located, with same name of that package name you wont to use.

2. Define package: In that package folder create Java files for your classes and interfaces you wont to create and use in main Java program,

    - in top of that all classes and interfaces Java file write `package <package_name>;`

(Example package my_package;).
   - in that package all directly useable classes and interfaces are declare with `public` statement.
   - Example: google/Google.java

   ```
   package google;

   public class GoogleCrome {
     public void search(String query) {
       System.out.println("Search result for " + query);
     }
   }
   ```

3. Use package: use `import` statement to import that package classes and Interfaces

   - Sysntex: in file where you wont to use that package code, top first line write `import <package_name>.<class_name>;`
   - you can use `*` instant of using `<class_name>` for importing all classes and interfaces from that package.
   - Now you can use that class or interface in your main java program.
   - Example: App.java

   ```
   import google.GoogleCrome;

   public App {
     public static void main(String[] args) {
       GoogleCrome myBrowser = new GoogleCrome();
       myBrowser.search("Use of java in web development side");
     }
   }
   ```

3. Write a program to swap two number in java.

```
public class SwapTwoNumber {
  static void print(int a, int b) {
    System.out.println("a - " + a + ", b - " + b);
  }

  static void simpleSwap() {
    int a = 10, b = 20;
    print(a, b);
    int c = a;
    a = b;
    b = c;
    print(a, b);
  }

  static void bitSwap() {
```

```
    int a = 111, b = 999;
    print(a, b);
    a ^= b;
    b ^= a;
    a ^= b;
    print(a, b);
  }

  public static void main(String[] args) {
    simpleSwap();
    bitSwap();
  }
}
```

4. How iteration statement works in java?

A computer is well suited to running some block of code number of time. for example getting sum of odd and even number from 1 to 1000.

Example: Simple algorithm of getting total of odd and even number that use iteration statement

```
odd_sum = 0
even_sum = 0
i = 1
while(i <= 1000):
    if((i % 2) == 0):
        even_sum = even_sum + i;
    else:
        odd_sum = odd_sum + i;
    i = i + 1;

print odd_sum
print even_sum
```

Every programming language must have features that instruct a computer to perform such repetitive task.

Iteration : The process of repeatedly executing a block of code statement with specific condition that known as iteration statement.

In simple words iteration statement is one type of looping statement that execute same code while condition is true.

If Iteration have not any condition that goes to *infinite loop*. like while loop haveing always `true` condition.

Java Iteration statement :

- Java support iteration statement loop like *while loop*, *do..while loop*, *for loop*, *for each loop*

- As well as the *recursion* of the method that also work like iteration we need to create that type of method for specific iteration
- In Java the process of simple Iteration statement

  Example: getting sum of odd and even number 1 to 1000

  ```
  int i = 1, odd_sum = 0, even_sum = 0;

  while(i <= 1000) {
    if (i % 2 == 0)
      even_sum += i++;
    else
      odd_sum += i++;
  }

  System.out.println("Even number total: " + even_sum);
  System.out.println("Odd number total: " + odd_sum);
  ```

  - here, in this example iterating 1 to 1000 number we use `while` loop with condition `i <= 1000` that specifying that while `i` is less than and equal to 1000 that while loop code block is execute.
- And the *for each loop* is best example of the iterating array elements from array one by one.

  Example :

  ```
  String[] sentence = {"Running", "Same", "Code", "Each", "Time", "with", "Condition"}

  for (String word: sentence) {
    System.out.println(word);
  }
  ```

  This for each loop is automatically fetching array element and put that into for each loop body for operations. This is perfect example of iteration statement that more specifically use with array structure.