Assignment 3

1. **Create a class Publication with data members title(String) and price(int). From this class derive two classes Book and CD. Class Book adds pages(int) and CD adds Size(int). Each of these classes should have constructors and display(). Write a java program to implement this using super, this and method overriding concepts.**

Program:

```
class Publications{
String title="The Magic Day";
double price=599.00;
int pages;
double size;
}
class Book extends Publications{
    public Book(int pages){
        this.pages=pages;
                }
                public void display(){
                System.out.println("The price of Book :"+super.title+" is :"+super.price+" with page numbers:"+pages);
    }
        }
        class CD extends Publications{
    public CD(double size){
        this.size=size;
      }
                public void display(){
                System.out.println("The price of CD :"+title+" is :"+super.price+" with the size of : "+size);
    }
}
class Main{
    public static void main(String args[]){
        Book b=new Book(125);
        CD c=new CD(23.2);
        b.display();
        c.display();
```

```
    }
    }
```

Output:

```
F:\RSC>javac Main.java

F:\RSC>java Main
The price of Book :The Magic Day is :599.0 with page numbers:125
The price of CD :The Magic Day is :599.0 with the size of : 23.2

F:\RSC>
```

2. **Write a simple java program to demonstrate method overriding**.

Program :

```
class Override{
   public void print()
   {
      System.out.println("Hello there !!");
   }
}
class Override2 extends Override{
   public void print(){
      System.out.println("How are you ?");
   }
   public static void main( String args[]) {
      Override2 obj = new Override2();
      obj.print();
   }
}
```

Output:

F:\RSC>javac Override2.java

F:\RSC>java Override2
How are you ?

3. **Write a java program to create an interface called Shape with CalculateArea(). Create three classes namely Square,Circle,Triangle which implements Shape.**

Program:

```
interface Shape{
    public double Calculatearea();
}
class Circle implements Shape{
    double rad;
    public Circle(double r){
        rad=r;
    }
    public double Calculatearea(){
        return 3.14 * rad* rad;
    }
}
class Rectangle implements Shape{
    double length;
    double breadth;
    public Rectangle(double l, double b){
        length = l;
        breadth = b;
    }
    public double Calculatearea(){
        return length * breadth;
    }
}


class Triangle implements Shape{
    double base;
```

```
    double height;
    public Triangle(double b, double h){
        base=b;
                height=h;
    }
    public double Calculatearea(){
        return (base*height)/2;
    }
}

class Interface{
    public static void main(String args[]){
        Circle c = new Circle(3.52);
                Rectangle r = new Rectangle(5.0, 2.5);
                Triangle t = new Triangle(5.0, 2.5);
        System.out.println("Area of circle: " + c.Calculatearea());

        System.out.println("Area of rectangle: " + r.Calculatearea());
                System.out.println("Area of triangle: " + t.Calculatearea());
    }
}
```

Output:

```
F:\RSC>javac Interface.java

F:\RSC>java Interface
Area of circle: 38.90585600000001
Area of rectangle: 12.5
Area of triangle: 6.25
```

4. **Create two packages p1 and p2. The package p1 contains class A which contains one display(). Create class B in package p2. The main method of class B invoke A's display(). Write a java program to do this.**

Program:

A.java

```
package p1;
public class A{
	public void display() {
		System.out.println("Hello ! I am in A class");
	}
}
```

B.java:

```
package p2;
import p1.*;

public class B{
	public static void main(String[] args){
		A a=new A();
		a.display();
	}
}
```

Output :

```
F:\RSC>javac A.java

F:\RSC>javac B.java

F:\RSC>java B
```

Hello ! I am in A class

5. **Write a java program to count numbers, characters in the command line arguments using Exception handling mechanism.**

Program:

```
public class CommandLine {
   public static void main(String args[]) {
          try{
                int d=0,c=0;
                for(int i=0;i<args.length;i++){
                      if(Character.isLetter(args.charAt(i)))
                            c++;
                      else
                            d++;
                }
                System.out.println("CHARACTERS  "+c+"  DIGITS "+d);
          }
          catch(Exception e){
                System.out.println(e);
          }
   }
}
```

Output :

F:\RSC>javac CommandLine.java

F:\RSC>java CommandLine fs108giet117

CHARACTERS 6 DIGITS 6

1. What is Inheritance?

The process by which one class acquires the properties and functionalities of another class is called inheritance. Inheritance allows us to reuse the code and improves reusability in our java application.

In Inheritance, each class is allowed to have one direct superclass, and each superclass has the potential for an unlimited number of *subclasses*.

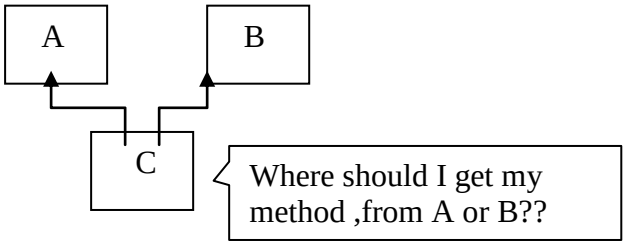
subclass (child) - the class that inherits from another class

superclass (parent) - the class being inherited from

2. What is Multiple Inheritance?

When one class extends more than one classes then this is called multiple inheritance. Java does not support multiple inheritance using classes to avoid the ambiguity caused by it.But,it can be achieved or implemented using interfaces.

Suppose C is inherited from A,B classes.If C wants to access the same method overridden in class A and B there arises the ambiguity from which class it should be accessed.



So, Multiple Inheritance is not possible in Java.

3. **What is the use of Super keyword?**

The super keyword refers to superclass (parent) objects.It is used to call superclass methods, and to access the superclass constructor.

The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

4. What is abstract method?

A method which is declared as abstract and does not have implementation is known as an abstract method. If a class has an abstract method it should be declared abstract, the vice versa is not true, which means an abstract class doesn't need to have an abstract method compulsory.

Example : abstract void Print();

5. What is abstract class?

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated. It can have constructors and static methods also.

Example : abstract class RSC{}

6. What is the use of final modifier?

The final modifier keyword makes that the programmer cannot change the value anymore.The final modifier can be associated with methods, classes and variables. Once declared final –

- A final class cannot be instantiated.
- A final method cannot be overridden.
- A final variable cannot be reassigned.

7. What is interface? Write the syntax interface.

The interface in Java is *a mechanism to achieve abstraction*. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.

There are mainly three reasons to use interface:

1.)It is used to achieve abstraction.

2.)By interface, we can support the functionality of multiple inheritance.

3.)It can be used to achieve loose coupling.

Syntax:

```
interface <interface_name>{
    // declare constant fields
   // declare methods that abstract
    // by default.
   }
```

8.What is package?

PACKAGE in Java is a collection of classes, sub-packages, and interfaces. It helps organize your classes into a folder structure and make it easy to locate and use them. More importantly, it helps improve code reusability.

Each package in Java has its unique name and organizes its classes and interfaces into a separate namespace, or name group.

9.What is exception?

An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

An exception can occur for many different reasons. Following are some scenarios where an exception occurs.

- A user has entered an invalid data.

- A file that needs to be opened cannot be found.
- A network connection has been lost in the middle of communications or the JVM has run out of memory.

10. What is the use of finally block?

It is a block that is used *to execute important code* such as closing connection, stream etc.This finally block is always executed whether exception is handled or not.Java finally block follows try or catch block.

Syntax :

```
try{

 //Statements that may cause an exception

}

catch{

//Handling exception

}

finally{

//Statements to be executed

}
```