Assignment-3

1.Implementation of super,this and method overriding

Program:

```
class Publication
{
String title;
int price;
public Publication(String title,int price)
{
this.title=title;
this.price=price;
//System.out.println(&amp;quot;In publication constructor&amp;quot;);
}
void display()
{
System.out.println(&amp;quot;In publication,display method&amp;quot;);
System.out.println(&amp;quot;Title: &amp;quot;+title+&amp;quot; Price: &amp;quot;+price);
}
}
class Book extends Publication
{
int pages;
Book(String title,int price,int pages)
{
super(title,price);
this.pages=pages;
```

```
}
void display()
{
super.display();
System.out.println(&quot;In book class&quot;);
System.out.println(&quot;Title: &quot;+title+&quot; Price: &quot;+price+&quot;
Pages:
&quot;+pages);

}
}
class CD extends Book
{
int size;
CD(String title,int price,int pages,int size)
{
super(title,price,pages);
this.size=size;
}
void display()
{
super.display();
System.out.println(&quot;In CD class&quot;);
System.out.println(&quot;Title: &quot;+title+&quot; Price: &quot;+price+&quot;
Pages:
&quot;+pages+&quot; Size: &quot;+size);
}
```

```
}
public class InheritResult
{
public static void main(String args[])
{
String title=args[0];
int price=Integer.parseInt(args[1]);
int pages=Integer.parseInt(args[1]);
int size=Integer.parseInt(args[1]);
CD cd=new CD(title,price,pages,size);
cd.display();
}
}
```

Output:

```
E:\fullstackjava&gt;javac InheritResult.java
E:\fullstackjava&gt;java InheritResult &quot;Amazing life&quot; 2400 5000 30
In publication,display method
Title: Amazing life Price: 2400
In book class
Title: Amazing life Price: 2400 Pages: 2400

In CD class
Title: Amazing life Price: 2400 Pages: 2400 Size: 2400
```

2.Method overriding

Program:

```
import java.lang.Math;
class Circle
```

```
{
public void area(int radius)
{
double areacircle=(double)Math.PI*radius*radius;
System.out.println(&amp;quot;area of circle is&amp;quot;+areacircle);
}
}
class Semicircle extends Circle
{
static int radius=5;
public void area(int radius)
{
super.area(radius);
double areasemi=((double)Math.PI*radius*radius)/2;
System.out.println(&amp;quot;area of semi circle is&amp;quot;+areasemi);
}
public static void main(String args[])
{
Semicircle sc=new Semicircle();
sc.area(radius);
}
}
```

Output:

```
E:\fullstackjava&amp;gt;javac Semicircle.java
E:\fullstackjava&amp;gt;java Semicircle
area of circle is78.53981633974483
area of semi circle is39.269908169872416
```

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

Program:-

```
import java.lang.Math;
interface Shape
{
void CalculateArea();
}
class Square implements Shape
{
public void CalculateArea()
{
int side=4;
int area=side*side;
System.out.println(&amp;quot;the area of square is &amp;quot;+area);
}
}
class Circle implements Shape
{
double radius=3;
public void CalculateArea()
{
double area=Math.PI*radius*radius;
System.out.println(&amp;quot;the area of circle is &amp;quot;+area);
}
```

```
}
class Triangle implements Shape
{
public void CalculateArea()
{
int base=3,height=4;
double area=0.5*base*height;

System.out.println(&amp;quot;the area of triangle is &amp;quot;+area);
}
}
public class Test
{
public static void main(String args[])
{
Shape sh1=new Square();
Shape sh2=new Circle();
Shape sh3=new Triangle();
sh1.CalculateArea();
sh2.CalculateArea();
sh3.CalculateArea();
}
}
```

Output:

```
E:\fullstackjava&amp;gt;javac Test.java
E:\fullstackjava&amp;gt;java Test
the area of square is 16
```

the area of circle is 28.274333882308138

the area of triangle is 6.0

4.Package

Program:

A.java:

```
package p1;
public class A
{
public void display()
{
System.out.println(&amp;quot;I am in package p1&amp;quot;);
}
}
```

B.java:

```
package p2;
import p1.A;
class B

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

Output:

E:\fullstackjava>javac A.java

E:\fullstackjava>java A

Error: Could not find or load main class A

E:\fullstackjava>javac B.java

E:\fullstackjava>java B

I am in package p1

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

Program:

```
import java.util.*;
class Count
{
public static void main(String args[])
{
String s=args[0];
try
{
int digit=0;
int character=0;
for(int i=0;i&amp;lt;s.length();i++)
{
if(Character.isLetter(s.charAt(i)))
character++;
else
digit++;
}
```

```
System.out.println(&quot;no.of characters are: &quot;+character+&quot;\nno of digits
are: &quot;+digit);
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

Output:

```
E:\fullstackjava&gt;javac Count.java
E:\fullstackjava&gt;java Count Ab567Hgj89
no.of characters are: 5
no of digits are: 5
```

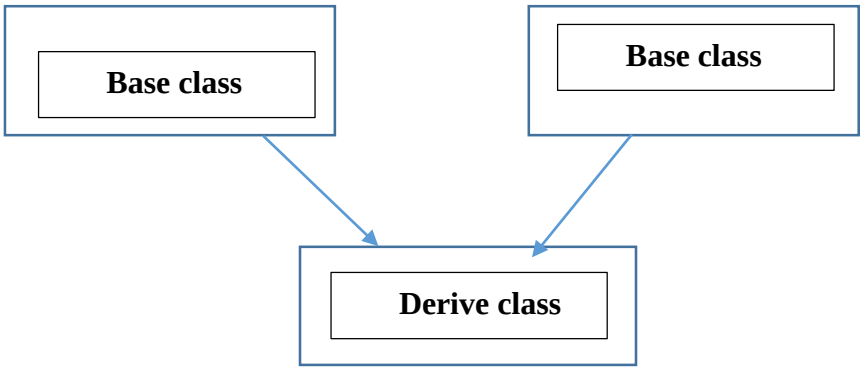
1. What is Inheritance?

Ans:- Inheritance can be defined as the process where one class acquires the properties (Methods and fields) of another. With the use of inheritance the information is made manageable in a hierarchical The class which inherits the properties of other is known as subclass (derived class, child class) whose properties are inherited is known as superclass (base class, parent class).extends is the keyword used to inherit the properties of a class.

2. What is Multiple Inheritance?

Ans:- In Multiple inheritance, one class can have more than one superclass and inherit features from all parent classes.

Java does not support Multiple inheritance.



3. What is the use of Super keyword?

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

Use of super keyword

- super variable refers immediate parent class instance.
- super variable can invoke immediate parent class method.
- super() acts as immediate parent class constructor and should be the first line in child class constructor.

4. What is abstract method?

Ans:- A method which is declared as abstract and does not have implementation is known as an abstract method.

Eg:-abstract void printStatus();//no method body and abstract

5. What is abstract class?

Ans:- 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.
- It can have final methods which will force the subclass not to change the body of the method.

6. What is the use of final modifier?

Ans:- 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.

Ans: - An interface is a reference type in Java. It is similar to class. It is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

Along with abstract methods, an interface may also contain constants, default methods, static methods, and nested types. Method bodies exist only for default methods and static methods.

Writing an interface is similar to writing a class. But a class describes the attributes and behaviors of an object. And an interface contains behaviors that a class implements.

Unless the class that implements the interface is abstract, all the methods of the interface need to be defined in the class.

An interface is similar to a class in the following ways −

- An interface can contain any number of methods.
- An interface is written in a file with a **.java** extension, with the name of the interface matching the name of the file.

- The byte code of an interface appears in a **.class** file.
- Interfaces appear in packages, and their corresponding bytecode file must be in a directory structure that matches the package name.

Syntax:-interface{

//methods

}