Lab Program - 6

Student.java

```
package CIE;
import java.util.Scanner;
public class Student {
 public String name;
 public String usn;
 public int sem;
 public void display() {
 Scanner s = new Scanner (System.in);
 System.out.println("Name :-");
  name = s.next();
 System.out.print("USN :-");
  usn = s.next();
  System.out.print("semester :-");
  sem = s.nextInt();
 }}
```

Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals extends Student{
 public double ciem[];
```

```
public void display(){
ciem = new double[5];
Scanner t = new Scanner(System.in);
System.out.println("CIE marks for 5 subjects:");
for(int i=0;i<5;i++){
  ciem[i]=t.nextDouble();
}}
```

Externals.java

```
package SEE;
import java.util.*;
import CIE.*;
public class Externals extends CIE.Student{
public double semm[];
public void display(){
semm = new double[5];
Scanner s = new Scanner(System.in);
System.out.println("SEE marks for 5 subjects
                    (out of 100):");
for(int i=0;i<5;i++)
 semm[i] = s.nextDouble();
}}
```

Main.java

```
import CIE.*;
import SEE.*;
```

```
import java.util.Scanner;
public class main {
public static void main (String args[]){
int n;
Scanner s= new Scanner (System.in);
System.out.print (" enter the number of students:");
n=s.nextInt();
CIE.Student st[] = new CIE.Student [n];
CIE.Internals in[] = new CIE.Internals[n];
SEE.Externals ex[] = new SEE.Externals[n];
for (int i=0 ; i<n; i++){
st[i] = new CIE.Student ();
in[i] = new CIE.Internals();
ex[i] = new SEE.Externals();
st[i].display();
in[i].display();
ex[i].display();
System.out.println ("Total marks of student"+
        st[i].name+" in 5 subjects are:");
for (int j=0; j<5; j++){
System.out.println (in[i].ciem[j] + [ex[i].semm[j]
2));
}}}
}
```

Output:

Enter the number of students : - 1

Name : Raman

USN : 1bm65118

Semester : 4

CIE Marks for 5 subjects (out of 50):

45

47

39

49

44

SEE marks for 5 subjects (out of 100):

78

98

87

93

89

Total marks of student raman in 5 subjects are:

84.0

96.0

82.5

91.5

88.5

Lab Program 7

Write a program to demonstrate generics with multiple object parameters.

```
// A simple Generic type with two type
// parameters : T and V
class TwoGen<T, V>{
  T ob1;
  V ob2;
// Pass constructor a reference to an object
// of type T and an object of type V.
 TwoGen(T o1, V o2){
    ob1 = o1;
    ob2 = o2;
  }
//Show types of T and V.
  void showtypes(){
  System.out.println("Type of T is" + ob1.getClass
                    ().getName());
System.out.println("Type of V is" + ob2.getclass().
                    getName());
 }
T.getob1(){
  return ob1;
 }
```

```
V getob2() {
 return ob2;
}}
// Demonstrate TwoGen.
class simpGen {
public static void main(String args[]) {
TwoGen < Integer, String > tgobj = new TwoGen
                < Integer, String > (88, "Generics");
// Show the types
tgobj.showTypes();
// Obtain and show values.
int v = tgObj.getob1();
System.out.println("value:" + v);
String str = tgobj.getob2();
System.out.println("value:" + str);
}}
```

output

```
Type of T is java.lang.Integer
Type of V is java.lang.String
value: 88
value: generics
```

Date
Page No.

Lab Program 8

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age<0. In Son class, implement a constructor that ~~takes~~ cases both father and son's age and throws an exception if son's age is >= father's age.

```
import java.util.*;
class WrongAge extends Exception {
  private String detail;
  WrongAge(String s){
  detail = s;
 }
public String toString(){
  return ("Invalid age exception:" + detail
}}
class father {
int age;
father(int x) throws WrongAge {
```

```
age = x;
if(age < 0)
  throw new WrongAge("Age cant be negative");
}}
class son extends father{
  int age1;
  son(int fage, int sage) throws WrongAge{
    super(fage);
    age1 = sage;
   if(age1 >= age)
     throw new WrongAge("Son's age cant be greater
                         than father's age");
  }}
class expmain{
public static void main(String args[]){
  Scanner s = new Scanner(System.in);
 System.out.print("Enter Father's age:");
  int m = s.nextInt();
 System.out.print("Enter son's age:");
  int n = s.nextInt();
try{
  son ob = new son(m,n);
System.out.println("Father's Age:" +ob.age);
System.out.println("Son's Age:" +ob.age1);
```

```
}
  catch (WrongAge e){
   System.out.println(e);
}}}
```

Output:

Enter Father's age: -25
Enter Son's age: 12
Invalid age exception: Age can't be negative

Enter Father's age: 25
Enter Son's age: 35
Invalid age exception: Son's age cant be greater than father's age.

Enter Father's age: 27
Enter Son's age: 12
Father's Age: 27
Son's Age: 12

Date
Page No.

Lab Program 9

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

```
class NewThread implements Runnable {
    private String name;
    private int interval;
    private Thread t;

NewThread (String threadname, int interval) {
    this.name = threadname;
    this.interval = interval;
    t = new Thread (this, name);
    t.start ();
}

public void run () {
    try {
    for (int i=5; i>0; i--) {
    System.out.println ("Thread --" + this.name);
    Thread.sleep (this.interval);
    }
}
```

```
catch (InterruptedException e){
  System.out.println(name + "Interrupted");
}
}
}
class MultiThread {
  public static void main(String args[]) {
  new NewThread("BMS College of Engineering",10000);
  new NewThread("CSE",2000);
}}
```

Output:

```
Thread -- BMS college of Engineering
Thread -- CSE
Thread -- CSE
Thread -- CSE
Thread -- CSE
Thread -- CSE
Thread -- BMS college of Engineering
Thread -- BMS college of Engineering
Thread -- BMS college of Engineering
Thread -- BMS college of Engineering
```