69_Debayudh Mitra

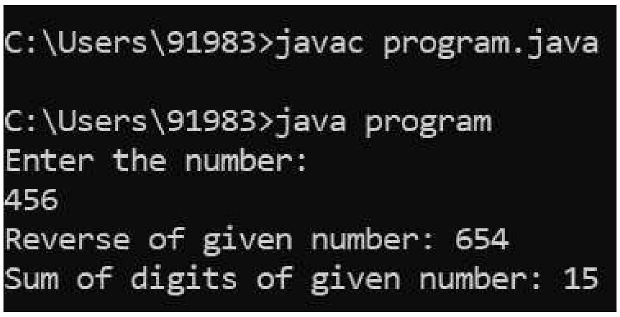
ASSIGNMENT - 1

1. Write a program to find the sum of digits and reverse of a given number (take input using command-line argument).

   Code:

```
import java.io.*;
import java.util.Scanner;
class program {
public static void main(String[] args)
{
int num, rem;
int rev = 0, sum = 0;
System.out.println("Enter the number: ");
Scanner sc = new Scanner(System.in);
num = sc.nextInt();
do {
rem = num % 10;
rev = rev * 10 + rem;
sum = sum + rem;
num = num / 10;
}
while (num > 0);
System.out.println("Reverse of given number: "+ rev);
System.out.println("Sum of digits of given number: "+ sum);
}
}
```

   Output:



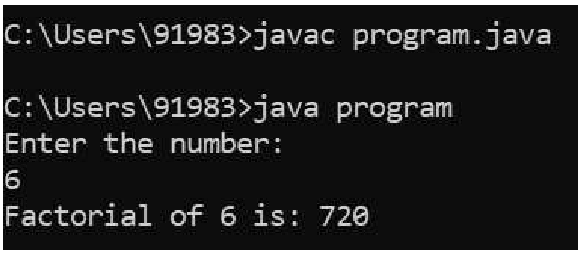
```
C:\Users\91983>javac program.java

C:\Users\91983>java program
Enter the number:
456
Reverse of given number: 654
Sum of digits of given number: 15
```

2. Write a program to find the factorial of a given integer number using recursion (take input using command-line argument).

   Code:

```
import java.io.*;
import java.util.Scanner;
class program{
 static int factorial(int n){
  if (n == 0)
    return 1;
  else
    return(n * factorial(n-1));
 }
 public static void main(String args[]){
  int i,fact=1;
  System.out.println("Enter the number: ");
  Scanner sc = new Scanner(System.in);
  int number = sc.nextInt();
  fact = factorial(number);
  System.out.println("Factorial of "+number+" is: "+fact);
 }
}
```

Output:



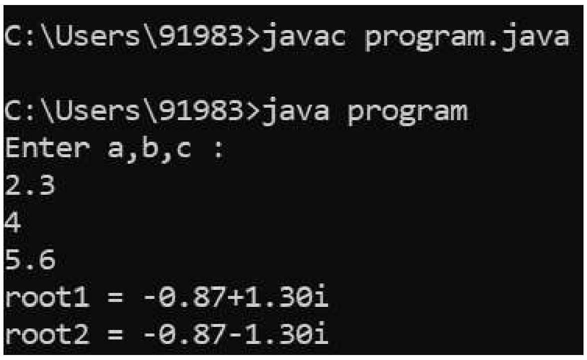
```
C:\Users\91983>javac program.java

C:\Users\91983>java program
Enter the number:
6
Factorial of 6 is: 720
```

3. Write a program to find the real roots of the quadratic equation $ax^2 + bx + c=0$ where a, b and c are constants (take input using command-line argument).

   Code:

```
import java.io.*;
import java.util.Scanner;
public class program {
  public static void main(String[] args) {
  System.out.println("Enter a,b,c : ");
  Scanner sc = new Scanner(System.in);
  double a=sc.nextDouble();
  double b=sc.nextDouble();
  double c=sc.nextDouble();
    double root1, root2;
    double determinant = b * b - 4 * a * c;
    if (determinant > 0) {
      root1 = (-b + Math.sqrt(determinant)) / (2 * a);
      root2 = (-b - Math.sqrt(determinant)) / (2 * a);
      System.out.format("root1 = %.2f and root2 = %.2f", root1, root2);
    }
    else if (determinant == 0) {
      root1 = root2 = -b / (2 * a);
      System.out.format("root1 = root2 = %.2f;", root1);
    }
    else {
      double real = -b / (2 * a);
      double imaginary = Math.sqrt(-determinant) / (2 * a);
      System.out.format("root1 = %.2f+%.2fi", real, imaginary);
      System.out.format("\nroot2 = %.2f-%.2fi", real, imaginary);
    }
  }
}
```

Output:



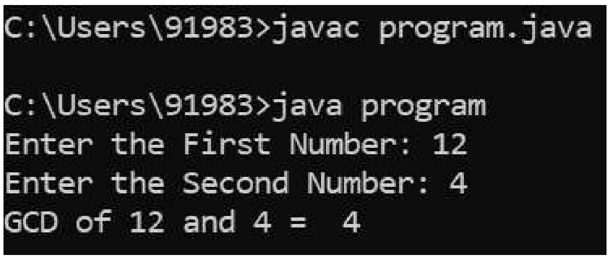
```
C:\Users\91983>javac program.java

C:\Users\91983>java program
Enter a,b,c :
2.3
4
5.6
root1 = -0.87+1.30i
root2 = -0.87-1.30i
```

4. Write a program to calculate GCD of two integer numbers (take input using command-line argument).

   Code:

```
import java.util.Scanner;
public class program  {
public static void main(String[] args)   {
int a, b, gcd = 0;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the First Number: ");
a = sc.nextInt();
System.out.print("Enter the Second Number: ");
b = sc.nextInt();
gcd = findGCD(a, b);
System.out.println("GCD of " + a + " and " + b + " =  " + gcd);
}
public static int findGCD(int a, int b)  {
while(b != 0)  {
if(a > b)  {
a = a - b;
}
else  {
b = b - a;
}
}
return a;
}
}
```

   Output:



```
C:\Users\91983>javac program.java

C:\Users\91983>java program
Enter the First Number: 12
Enter the Second Number: 4
GCD of 12 and 4 =  4
```

69_Debayudh Mitra

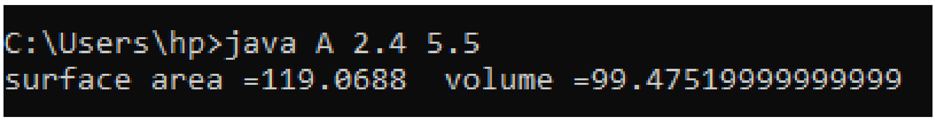
ASSIGNMENT – 2

1. Write a program to find surface area and volume of cylinder using constructor.

Code:

```
class B
{
double radius; double height; double surfacearea; double volume;
B(double y, double z)
{
radius=y; height=z;
}
void cal()
{
surfacearea=2*3.14*radius*(radius+height);
volume=3.14*radius*radius*height;
System.out.println("surface area ="+surfacearea+" volume ="+volume);
}
}
class A
{
public static void main(String[] arg)
{
double r,h;
r=Double.parseDouble(arg[0]);
h=Double.parseDouble(arg[1]);
B ob1=new B(r,h);
ob1.cal();
}
}
```

Output:



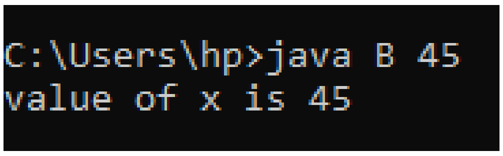
```
C:\Users\hp>java A 2.4 5.5
surface area =119.0688  volume =99.47519999999999
```

2. Create a class name first make an instance variable x and an instant method show put the main method inside the class and use the instant variable and instant method from main.

Code:

```
class B
{
int x;
B(int y)
{
x=y;
}
void fun()
{
System.out.println("value of x is "+x);
}
public static void main(String[] arg)
{
int r; r=Integer.parseInt(arg[0]); B ob1=new B(r);
ob1.fun();
}
}
```

Output:



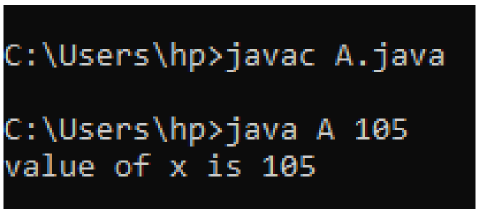
```
C:\Users\hp>java B 45
value of x is 45
```

3. Create a class make an instant variable and instant method. Use them from main which is declared in different class.

Code:

```
class B
{
int x;
B(int y)
{
x=y;
}
void fun()
{
System.out.println("value of x is "+x);
}
}
class A
{
public static void main(String[] arg)
{
int r; r=Integer.parseInt(arg[0]); B ob1=new B(r);
ob1.fun();
}
}
```

Output:



```
C:\Users\hp>javac A.java

C:\Users\hp>java A 105
value of x is 105
```

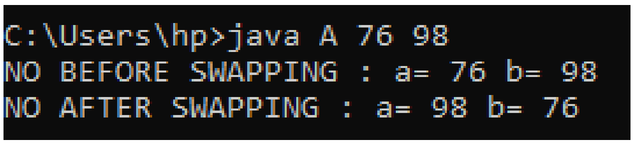
4. Write a program to swap two number using call by value method.

Code:

```
class B
{
int x,y;
B(int u,int v)
{
x=v; y=u;
}
}
class A
```

```
{
public static void main(String[] arg)
{
int a,b;
a=Integer.parseInt(arg[0]);
b=Integer.parseInt(arg[1]);
System.out.println("NO BEFORE SWAPPING : a= "+a+" b= "+b);
B ob1=new B(a,b);
a=ob1.x;
b=ob1.y;
System.out.println("NO AFTER SWAPPING : a= "+a+" b= "+b);
}
}
```

Output:



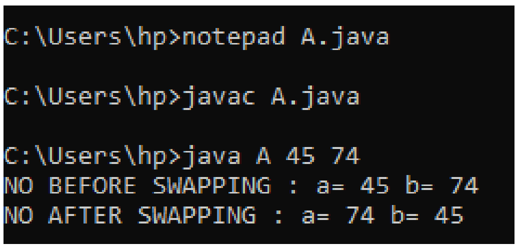
```
C:\Users\hp>java A 76 98
NO BEFORE SWAPPING : a= 76 b= 98
NO AFTER SWAPPING : a= 98 b= 76
```

5. Write a program to swap two number using call by reference method.

Code:

```
class B
{
int x,y;
B(int u,int v)
{
x=u; y=v;
}
public void swap(B ob)
{
x=ob.y; y=ob.x;
}
}
class A
{
public static void main(String[] arg)
{
int a ,b;
a=Integer.parseInt(arg[0]);
b=Integer.parseInt(arg[1]);
System.out.println("NO BEFORE SWAPPING: a= "+a+" b= "+b);
B ob1=new B(a,b);
B ob2=new B(a,b);
ob2.swap(ob1);
a=ob2.x; b=ob2.y;
System.out.println("NO AFTER SWAPPING: a= "+a+" b= "+b);
}
}
```

Output:



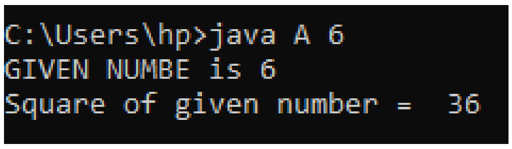
```
C:\Users\hp>notepad A.java

C:\Users\hp>javac A.java

C:\Users\hp>java A 45 74
NO BEFORE SWAPPING : a= 45 b= 74
NO AFTER SWAPPING : a= 74 b= 45
```

6. Write a program to show how a method returns an object.

Code:

```
class B{
int x;
B(int y){
x=y;
}
B sqre(){
B ob2=new B(x*x); return ob2;
}}
class A{
public static void main(String[] arg){
int a; a=Integer.parseInt(arg[0]);
System.out.println("GIVEN NO is : "+a); B ob1=new B(a);
B ob3; ob3=ob1.sqre();
System.out.println("Square of given no is : "+ob3.x);
}}
```

Output:



```
C:\Users\hp>java A 6
GIVEN NUMBE is 6
Square of given number =  36
```

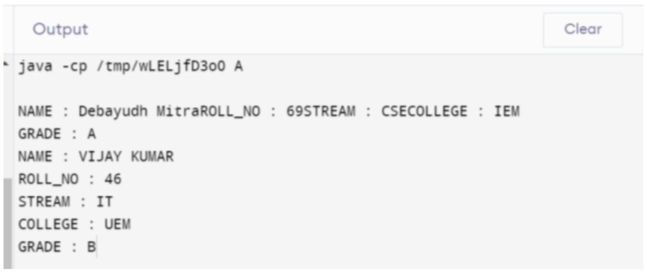
7. Writea java programtomakea student class withproper attribute likeroll no, name, stream, college, grade from main create such two students andshow there information.

Code:

```
class Student

{

String name; int rollno; String stream; String college; String grade;

Student(String n,int r,String s,String c,String g)

{

this.name=n; this.rollno=r; this.stream=s; this.college=c; this.grade=g;

}

public void display()

{

System.out.println("NAME : "+this.name);

System.out.println("ROLL_NO : "+this.rollno);

System.out.println("STREAM : "+this.stream);

System.out.println("COLLEGE : "+this.college);

System.out.println("GRADE : "+this.grade);

}}

class A{
```

```
public static void main(String[] arg){
Student s1=new Student("Debayudh Mitra",69,"CSE","IEM","A");
Student s2=new Student("VIJAY KUMAR",46,"IT","UEM","B");
s1.display();
s2.display();
}}
```

Output:



```
java -cp /tmp/wLELjfD3oO A

NAME : Debayudh MitraROLL_NO : 69STREAM : CSECOLLEGE : IEM
GRADE : A
NAME : VIJAY KUMAR
ROLL_NO : 46
STREAM : IT
COLLEGE : UEM
GRADE : B
```

69_Debayudh Mitra

ASSIGNMENT – 3

1. Design a class to represent the bank account to include the following things:

Fields:

- Name of Depositor
- Address of Depositor
- Account No.
- Balance amount in the account

Methods:

- To assign initial values
- To deposit an amount
- To withdraw an amount after checking balance
- To display the name, address & balance of a customer.

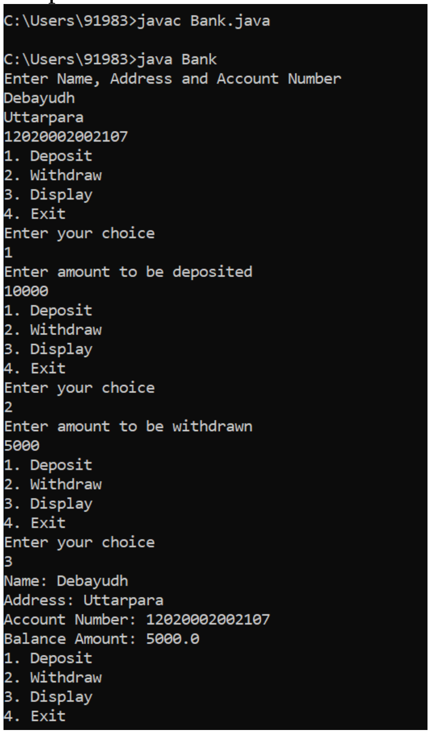
From main method create object and call these methods (menu driven code)

Code:

```
import java.util.*;
class Bank
{
   String name,address,accno;
   double balamt;
   Bank(String n,String add,String acc)
   {
      name=n;
      address=add;
      accno=acc;
      balamt=0.0;
   }
   void deposit(double d)
   {
      balamt+=d;
   }
   void withdraw(double w)
   {
      if(w>balamt)
      {
         System.out.println("Insufficient Balance");
      }
      else
      {
         balamt-=w;
      }
   }
   void display()
   {
      System.out.println("Name: "+name);
      System.out.println("Address: "+address);
      System.out.println("Account Number: "+accno);
      System.out.println("Balance Amount: "+balamt);
```

```
    }
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Name, Address and Account Number");
        String nm=sc.nextLine();
        String ad=sc.nextLine();
        String ac=sc.nextLine();
        Bank cbi=new Bank(nm,ad,ac);
        while(true)
        {
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Display");
            System.out.println("4. Exit");
            System.out.println("Enter your choice");
            int ch=sc.nextInt();
            switch(ch)
            {
                case 1:
                    System.out.println("Enter amount to be deposited");
                    double d=sc.nextDouble();
                    cbi.deposit(d);
                    break;
                case 2:
                    System.out.println("Enter amount to be withdrawn");
                    double w=sc.nextDouble();
                    cbi.withdraw(w);
                    break;
                case 3:
                    cbi.display();
                    break;
                case 4:
                    System.exit(0);
                    break;
                default:
                    System.out.println("Invalid Choice");
                    break;
            }
        }
    }
}
```

Output:



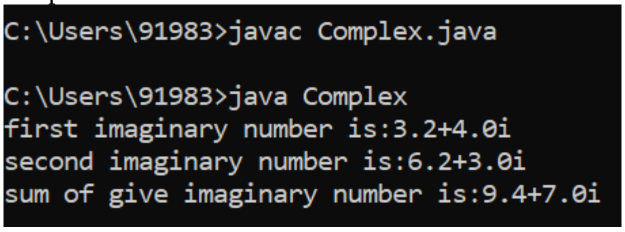
```
C:\Users\91983>javac Bank.java

C:\Users\91983>java Bank
Enter Name, Address and Account Number
Debayudh
Uttarpara
12020002002107
1. Deposit
2. Withdraw
3. Display
4. Exit
Enter your choice
1
Enter amount to be deposited
10000
1. Deposit
2. Withdraw
3. Display
4. Exit
Enter your choice
2
Enter amount to be withdrawn
5000
1. Deposit
2. Withdraw
3. Display
4. Exit
Enter your choice
3
Name: Debayudh
Address: Uttarpara
Account Number: 12020002002107
Balance Amount: 5000.0
1. Deposit
2. Withdraw
3. Display
4. Exit
```

2. Write a java program to add two complex no.

Code:

```
class Complex
{
  double real; double imag;
  Complex(double r,double i){
    real=r; imag=i;
  }
  void sum( Complex c){
    real=real+c.real; imag=imag+c.imag;
    System.out.println("sum of give imaginary number is:"+real+"+"+imag+"i");
  }
  public static void main(String[] args){
    Complex a=new Complex(3.2,4.0);
    Complex b=new Complex(6.2,3.0);
    System.out.println("first imaginary number is:"+a.real+"+"+a.imag+"i");
    System.out.println("second imaginary number is:"+b.real+"+"+b.imag+"i");
    a.sum(b);
  }
}
```

Output:



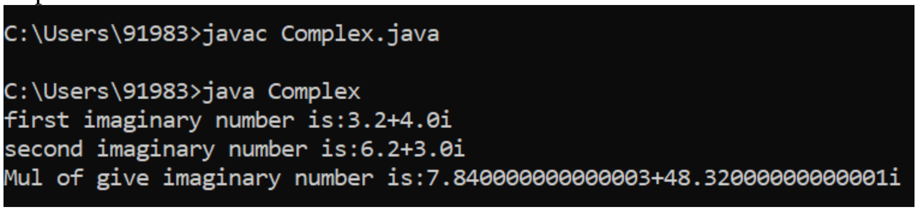
```
C:\Users\91983>javac Complex.java

C:\Users\91983>java Complex
first imaginary number is:3.2+4.0i
second imaginary number is:6.2+3.0i
sum of give imaginary number is:9.4+7.0i
```

3. Write a java program to multiply two complex no.

Code:

```
class Complex{

double real; double imag;
Complex(double r,double i ){
real=r; imag=i;
}
void mul( Complex c){
real=(real*c.real)-(imag*c.imag); imag=(real*c.imag)+(imag*c.real);
System.out.println("Mul of give imaginary number is:"+real+"+"+imag+"i");
}
public static void main(String[] args){
Complex a=new Complex(3.2,4.0);
Complex b=new Complex(6.2,3.0);
System.out.println("first imaginary number is:"+a.real+"+"+a.imag+"i");
System.out.println("second imaginary number is:"+b.real+"+"+b.imag+"i");
a.mul(b);
}
}
```

Output:



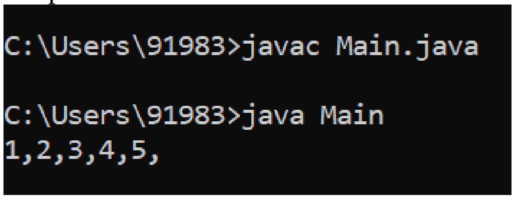
```
C:\Users\91983>javac Complex.java

C:\Users\91983>java Complex
first imaginary number is:3.2+4.0i
second imaginary number is:6.2+3.0i
Mul of give imaginary number is:7.840000000000003+48.32000000000001i
```

4. Write a java program to implement stack using basic function like push, pop and display.

Code:

```
class stack{
    int[] arr; int top=-1; stack(int n){
    arr=new int[n];
    }
    void push(int a){ top++; arr[top]=a;
    }
    void pop(){
    top--;
    }
    void disp(){
    for(int i=0;i<=top;i++){ System.out.print(arr[i]+",");
    }
    System.out.println();
    }}
    class Main{
    public static void main(String[] args){ stack s= new stack(8);
    s.push(1);
    s.push(2);
    s.push(3);
    s.push(4);
    s.push(5);
    s.push(6);
    s.push(7);
    s.pop();
    s.pop();
    s.disp();
    }
}
```

Output:



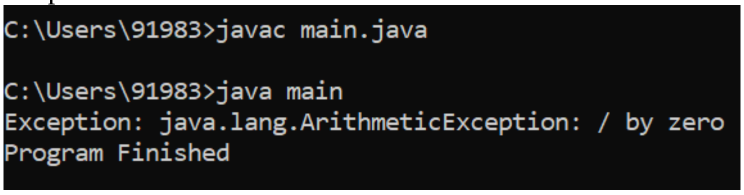
```
C:\Users\91983>javac Main.java

C:\Users\91983>java Main
1,2,3,4,5,
```

69_Debayudh Mitra

ASSIGNMENT – 4

1. Write a program to handle the ArithmeticException.

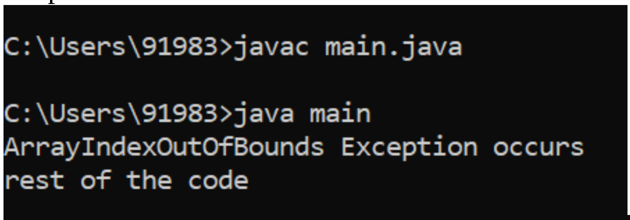
   Code:
```
public class main {
public static void main(String[] args) {
try {
int a = 10;
int b = 0;
int c = 0;
c = a / b;
System.out.println("Division is: " + c);
} catch (ArithmeticException e) {
System.out.println("Exception: " + e); }
System.out.println("Program Finished");
}}
```
   Output:



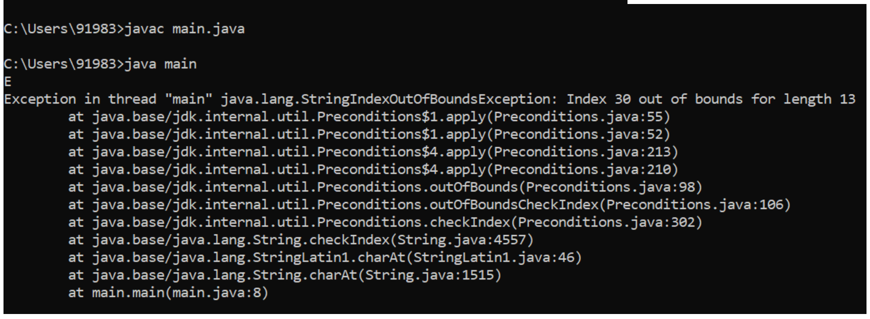
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Exception: java.lang.ArithmeticException: / by zero
Program Finished
```

2. Write a program for multiple catch to fire ArrayIndexOutOfBoundsException and StringIndexOutOfBoundsException both.

   Code:
```
public class main {
  public static void main(String[] args) {
        try{
             int a[]=new int[5];
             System.out.println(a[10]);
            }
            catch(ArithmeticException e) {
               System.out.println("Arithmetic Exception occurs");
              }
            catch(ArrayIndexOutOfBoundsException e) {
               System.out.println("ArrayIndexOutOfBounds Exception occurs");
              }
            catch(Exception e) {
               System.out.println("Parent Exception occurs");
              }
            System.out.println("rest of the code");
  }  }
import java.io.*;
import java.util.*;
class main {
            public static void main(String[] args)
            {
                        String s = "GEEKSFORGEEKS";
                        System.out.println(s.charAt(1));
                        System.out.println(s.charAt(30));
        }}
```
Output:



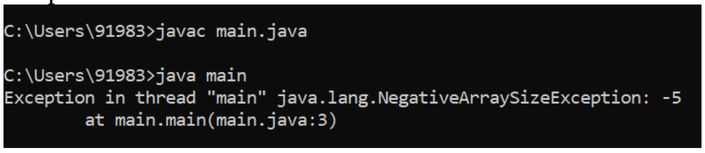
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
ArrayIndexOutOfBounds Exception occurs
rest of the code
```



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
E
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: Index 30 out of bounds for length 13
        at java.base/jdk.internal.util.Preconditions$1.apply(Preconditions.java:55)
        at java.base/jdk.internal.util.Preconditions$1.apply(Preconditions.java:52)
        at java.base/jdk.internal.util.Preconditions$4.apply(Preconditions.java:213)
        at java.base/jdk.internal.util.Preconditions$4.apply(Preconditions.java:210)
        at java.base/jdk.internal.util.Preconditions.outOfBounds(Preconditions.java:98)
        at java.base/jdk.internal.util.Preconditions.outOfBoundsCheckIndex(Preconditions.java:106)
        at java.base/jdk.internal.util.Preconditions.checkIndex(Preconditions.java:302)
        at java.base/java.lang.String.checkIndex(String.java:4557)
        at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:46)
        at java.base/java.lang.String.charAt(String.java:1515)
        at main.main(main.java:8)
```

3. Write a program to fire the NegativeArraySize exception.

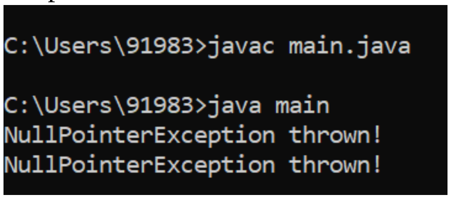
Code:
```
public class main {
   public static void main(String[] args) {
      int[] array = new int[-5];
      System.out.println("Array length: " + array.length);
   }}
```
Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Exception in thread "main" java.lang.NegativeArraySizeException: -5
        at main.main(main.java:3)
```

4. Define an object reference and initialize it to null. Try to call a method through this reference. Now wrap the in a try-catch clause to catch the exception.

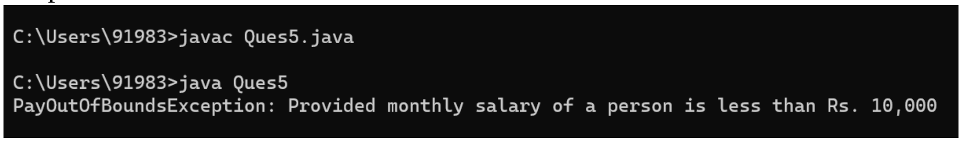
Code:
```
 public class main {
            public static void main(String[] args) {
                        String s = "abcd";
                        foo(null);
                        bar(null);
            }
            static void foo(String x){
                        try {
                                    System.out.println("First character: " + x.charAt(0));
                        }
                        catch(NullPointerException e) {
                                    System.out.println("NullPointerException thrown!");
                        }}
            static void bar(String x){
                        if(x != null)
                                    System.out.println("First character: " + x.charAt(0));
                        else
                        System.out.println("NullPointerException thrown!");
        }}
```
Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
NullPointerException thrown!
NullPointerException thrown!
```

5. Write a program in Java to create a user defined exception named PayOutOfBoundsException (provided the monthly salary of a person is less than Rs. 10,000 /) and fire the exception.

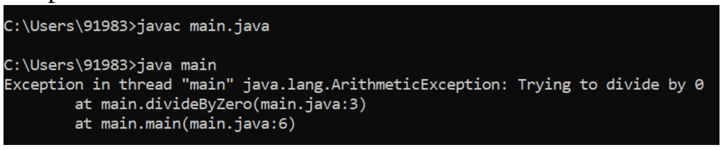
   Code:
```
class PayOutOfBoundsException extends RuntimeException{
public PayOutOfBoundsException(String e){
super(e);
}}
public class Ques5 {
public static void main(String[] args) {
int salary = 9000;
try{
if(salary<10000){
throw new PayOutOfBoundsException("Provided monthly salary of a person is less than Rs.
10,000");
} else {
System.out.println("Monthly Salary is "+salary); }
}catch(RuntimeException e){
System.out.println(e);
}}}
```
   Output:



```
C:\Users\91983>javac Ques5.java

C:\Users\91983>java Ques5
PayOutOfBoundsException: Provided monthly salary of a person is less than Rs. 10,000
```

6. Write a program to fire any checked exception manually using ‘throw’ keyword.

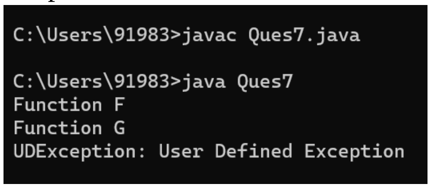
   Code:
```
class main {
        public static void divideByZero() {
          throw new ArithmeticException("Trying to divide by 0");
        }
        public static void main(String[] args) {
          divideByZero();
        }}
```
   Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Exception in thread "main" java.lang.ArithmeticException: Trying to divide by 0
        at main.divideByZero(main.java:3)
        at main.main(main.java:6)
```

7. Create a class with two methods, f( ) and g( ). In g( ), throw an exception of a new type that you define. In f( ), call g( ), catch its exception and, in the catch clause, throw a different exception (of a second type that you define). Test these methods from and within main( ).

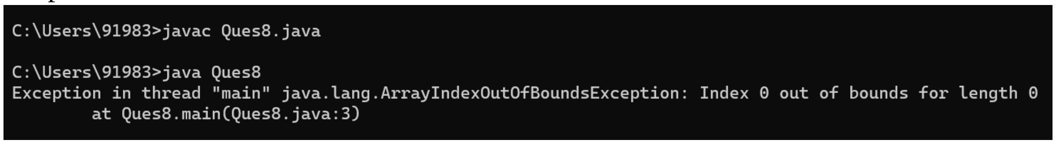
   Code:
```
class UDException extends RuntimeException{
public UDException(String e){
super(e);
}}
class A{
public void f(){
try{
System.out.println("Function F");
this.g();
}catch(RuntimeException e){
System.out.println(e);
}}
public void g() {
System.out.println("Function G");
throw new UDException("User Defined Exception");
}}
public class Ques7 {
public static void main(String[] args) {
A alpha = new A();
alpha.f();
}}
```
   Output:



```
C:\Users\91983>javac Ques7.java

C:\Users\91983>java Ques7
Function F
Function G
UDException: User Defined Exception
```

8. Write a program that takes one string and two integers as command line argument and prints the reverse of the substring of the string specified by the two numbers. The program should handle all possible exception that may arise due to bad input.
   Code:
```
public class Ques8 {
public static void main(String[] args) {
String string = args[0];
int startIndex = Integer.parseInt(args[1]);
int endIndex = Integer.parseInt(args[2]);
int length = string.length();
try{
if(startIndex<0 || endIndex>=length || endIndex<0 || startIndex>=length){
throw new StringIndexOutOfBoundsException();
} else if (startIndex>endIndex){
throw new StringIndexOutOfBoundsException("StartIndex Cannot Be Greater Than
EndIndex"); }
System.out.println(string.substring(startIndex, endIndex));
}catch(StringIndexOutOfBoundsException e){
```

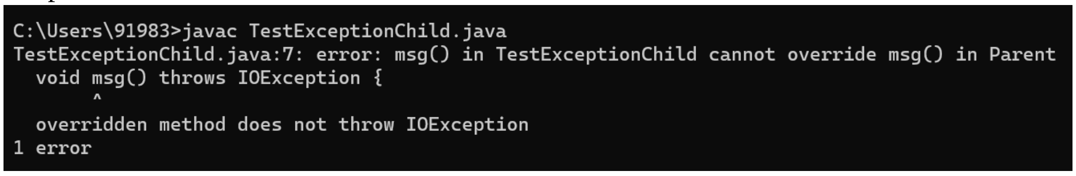
```
System.out.println(e);
}}}
```
Output:



```
C:\Users\91983>javac Ques8.java

C:\Users\91983>java Ques8
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 0 out of bounds for length 0
        at Ques8.main(Ques8.java:3)
```

9. Write a demo program to illustrate the restrictions of using ‘throws’ clause in method overriding with regard to superclass-subclass concept.

Code:
```
import java.io.*;
class Parent{
  void msg() {
    System.out.println("parent method");
    }}
public class TestExceptionChild extends Parent{
  void msg() throws IOException {
    System.out.println("TestExceptionChild"); }
  public static void main(String args[]) {
   Parent p = new TestExceptionChild();
   p.msg();
  }}
```
Output:



```
C:\Users\91983>javac TestExceptionChild.java
TestExceptionChild.java:7: error: msg() in TestExceptionChild cannot override msg() in Parent
  void msg() throws IOException {
       ^
  overridden method does not throw IOException
1 error
```

69_Debayudh Mitra

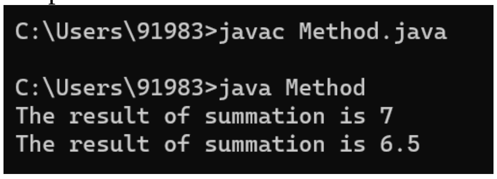
ASSIGNMENT – 5

1. Create a class and determine if method overloading holds good for return type of methods or not.

Code:

```
public class Method {
    public int  sum(int a,int b) {
        return (a+b); }
    public double sum(double a,double b) {
        return (a+b); }
    public static void main(String[] args) {
        Method method=new Method();
        int c=method.sum(3,4);
        System.out.println("The result of summation is "+c);
        double d=method.sum(1.2,5.3);
        System.out.println("The result of summation is "+d);
    }}
```

Output:



```
C:\Users\91983>javac Method.java

C:\Users\91983>java Method
The result of summation is 7
The result of summation is 6.5
```

2. Overload the constructors for classes Area and Volume of a rectangular figure and also display its area and volume. Area is the superclass and Volume is the subclass.

Code:

```
public class Box {
int h, r;
boolean isCube;
Box(int h) {
this.h = h;
this.isCube = true; }
Box(int r, int h) {
this.r = r;
this.h = h;
this.isCube = false; }
double getVolume(boolean chck) {
```

return 0.00; }

double getVolume() {

return this.isCube ? Math.pow(this.h, 3) : Math.PI * Math.pow(this.r, 2) * (this.h / 3); }

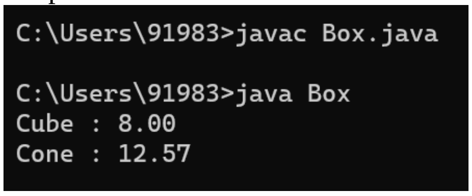
public static void main(String[] args) {

Box cube = new Box(2);

Box cone = new Box(2, 3);

System.out.println("Cube : " + String.format("%.2f", cube.getVolume()));

System.out.println("Cone : " + String.format("%.2f", cone.getVolume()));
}}
Output:



```
C:\Users\91983>javac Box.java

C:\Users\91983>java Box
Cube : 8.00
Cone : 12.57
```

3. Create a class Employee is having instance variables name and id. Create its subclass named Scientist which has instance variables no_of_publication and experience. Now create its subclass, say DScientist which has instance variable award. Put a method like:
public String toString(){ } in every class where you describe about the class and from main() method create object of each class and print each object.

Code:

public class EMP {

String name;

int id;

EMP(String name, int id) {

this.name = name;

this.id = id; }

public String toString() {

return String.format("[%s Class] Name: %s, ID: %d", this.getClass().getEnclosingClass() == null ? this.getClass().getSimpleName()

: this.getClass().getEnclosingClass().getSimpleName(), this.name, this.id); }

public static void main(String[] args) {

EMP e = new EMP("Debayudh Mitra", 1);

Scientist s = new Scientist("Debayudh Mitra", 1, 999999, "Expert");

Dscientist d = new Dscientist("Debayudh Mitra", 1, 999999, "Expert", "Professional Scientist");

System.out.println(e);

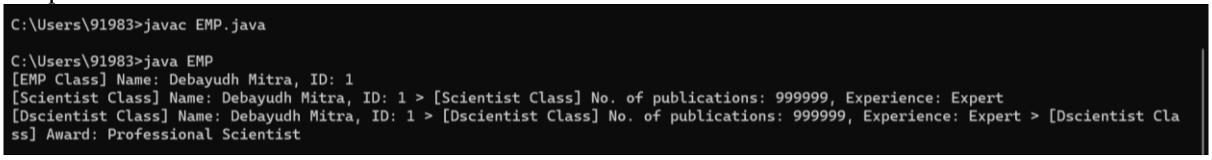
System.out.println(s);

System.out.println(d);

}}

```
class Scientist extends EMP {

int no_of_publication;

String experience;

Scientist(String name, int id, int no_of_publication, String experience) {

super(name, id);

this.no_of_publication = no_of_publication;

this.experience = experience; }

public String toString() {

return super.toString() + String.format(" > [%s Class] No. of publications: %d, Experience: %s",
this.getClass().getSimpleName(), this.no_of_publication, this.experience);

}}

class Dscientist extends Scientist {

String award;

Dscientist(String name, int id, int no_of_publication, String experience, String award) {

super(name, id, no_of_publication, experience);

this.award = award; }

public String toString() {

return super.toString() + String.format(" > [%s Class] Award: %s",
this.getClass().getSimpleName(), this.award);
}}
```

Output:



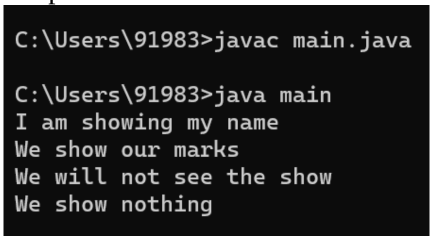
```
C:\Users\91983>javac EMP.java

C:\Users\91983>java EMP
[EMP Class] Name: Debayudh Mitra, ID: 1
[Scientist Class] Name: Debayudh Mitra, ID: 1 > [Scientist Class] No. of publications: 999999, Experience: Expert
[Dscientist Class] Name: Debayudh Mitra, ID: 1 > [Dscientist Class] No. of publications: 999999, Experience: Expert > [Dscientist Cla
ss] Award: Professional Scientist
```

4. Create a class with a method void show() and make three subclasses of it and all subclasses have this show() method overridden and call those methods using their corresponding object references.

Code:

```
class Show {

    public void show() {

        System.out.println("I am showing my name");

    }}

 class Show1 extends Show {

    public void show() {

        System.out.println("We show our marks");

    }}

class Show2 extends Show {
    public void show() {
        System.out.println("We will not see the show");
    }}
class Show3 extends Show {
    public void show() {
        System.out.println("We show nothing");
    } }
public class main {
    public static void main(String[] args) {
        Show show=new Show();
        show.show();
        Show1 show1=new Show1();
        show1.show();
        Show2 show2=new Show2();
        show2.show();
        Show3 show3=new Show3();
        show3.show();
    }}
```

Output:



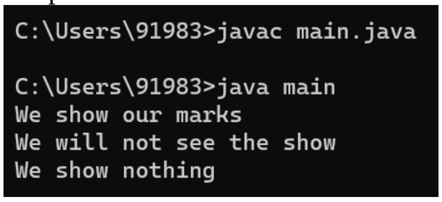
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
I am showing my name
We show our marks
We will not see the show
We show nothing
```

5. Do the problem 4 using dynamic method dispatching.

Code:

```
class A {
    void show() {
            System.out.println("We show our marks");
    } }
class B extends A {
    void show() {
            System.out.println("We will not see the show");
    } }
class C extends A {
    void show() {
            System.out.println("We show nothing");
    } }
public class main {
    public static void main(String args[]) {
            A a = new A();
            B b = new B();
            C c = new C();
            A ref;
            ref = a;
            ref.show();
            ref = b;
            ref.show();
            ref = c;
            ref.show();
    } }
```

Output:



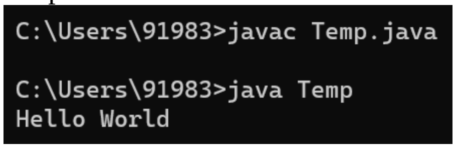
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
We show our marks
We will not see the show
We show nothing
```

6. Check without having any abstract method/s whether a class can be abstract; if so, then use that concrete method(s) from another class having main method.

Code:

```
public class Temp {
public static void main(String[] args) {
A.display();
}}
abstract class A {
static void display() {
System.out.println("Hello World");
}}
```

Output:



```
C:\Users\91983>javac Temp.java

C:\Users\91983>java Temp
Hello World
```

7. Create an abstract class with three abstract methods check whether you can we override only few methods (not all methods) in subclass or not.

Code:

public class AbsClass1 extends Parent {

void show1() {

System.out.println("Modified Show 1"); }

void show2() {

System.out.println("Modified Show 2"); }

public static void main(String[] args) {

AbsClass1 abc = new AbsClass1();

abc.show1();

abc.show2();

abc.show3();

}}

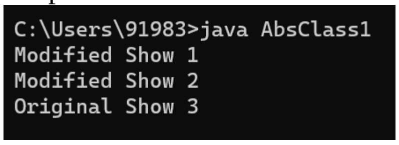
abstract class Parent {

abstract void show1();

abstract void show2();

void show3() {

System.out.println("Original Show 3");
}}
Output:



```
C:\Users\91983>java AbsClass1
Modified Show 1
Modified Show 2
Original Show 3
```

8. Create a class Parent having instance variables id, name and address. Create a class ChildOne having instance variables id, name, address and marks. Also create another class ChildTwo with instance variables id, name, address, qualification and salary. Within each class define your own method to display values of these variables. Design the program using super call with proper parameter and use object of each class from main() to display their properties.

Code:

import java.util.*;

public class Parent {

Scanner sc = new Scanner(System.in);

int id;

String name;

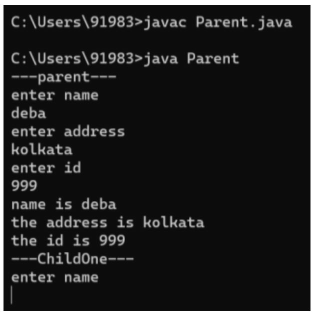
String address;

Parent() {

```
System.out.println("enter name");
this.name = sc.next();
System.out.println("enter address");
this.address = sc.next();
System.out.println("enter id");
this.id = sc.nextInt(); }
void display() {
System.out.println("name is " + name);
System.out.println("the address is " + address);
System.out.println("the id is " + id); }
public static void main(String[] args) {
System.out.println("---parent---");
Parent p = new Parent(); p.display();
System.out.println("---ChildOne---");
ChildOne c1 = new ChildOne(80); c1.display();
System.out.println("---ChildTwo---");
ChildTwo c2 = new ChildTwo(); c2.display();
}}
class ChildOne extends Parent {
Scanner sc = new Scanner(System.in);
float marks;
ChildOne(float m) {
super();
System.out.println("enter marks");
this.marks = m; }
void display() {
super.display();
System.out.println("the marks is " + marks);
}}
class ChildTwo extends Parent {
Scanner sc = new Scanner(System.in);
String qualification;
double salary;
ChildTwo() {
super();
```

```
System.out.println("enter qualification");

this.qualification = sc.next();

System.out.println("enter salary");

this.salary = sc.nextDouble(); }

void display() {

super.display();

System.out.println("qualification is " + qualification);

System.out.println("salary is " + salary);
}}
```

Output:



```
C:\Users\91983>javac Parent.java

C:\Users\91983>java Parent
---parent---
enter name
deba
enter address
kolkata
enter id
999
name is deba
the address is kolkata
the id is 999
---ChildOne---
enter name
```

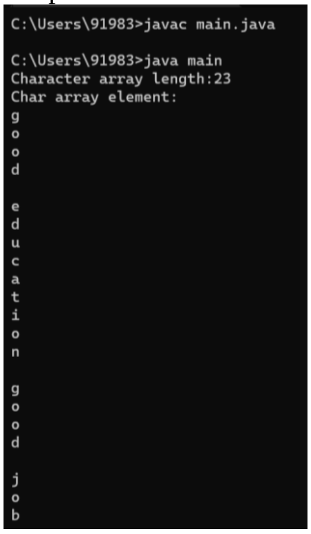
69_Debayudh Mitra

ASSIGNMENT – 6

1. Take a sting from keyboard and convert into character array (new one).
Code:

```
public class main{
public static void main(String[] args){
String s1="good education good job";
char[] ch=s1.toCharArray();
int len=ch.length;
System.out.println("Character array length:"+len);
System.out.println("Char array element:");
for(int i=0;i<len;i++){
System.out.println(ch[i]);
}}}
```

Output:

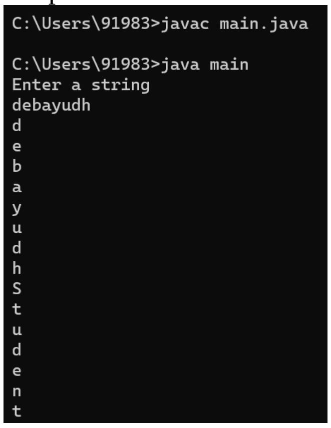


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Character array length:23
Char array element:
g
o
o
d

e
d
u
c
a
t
i
o
n

g
o
o
d

j
o
b
```

2. Take a string from keyboard and a char array (filled up to length 5). Now append the stringto that char array. Show the char array.
Code:

```
import java.util.*;
class main{
public static void main(String args[]){
int i;
String s, sp;
String str = "Student";
Scanner sc= new Scanner(System.in);
System.out.println("Enter a string");
s=sc.nextLine();
sp=s.concat(str);
char ch[] = new char[sp.length()];
for (i = 0; i <= sp.length(); i++){
ch[i] = sp.charAt(i);
System.out.println(ch[i]);
}}}
```

Output:

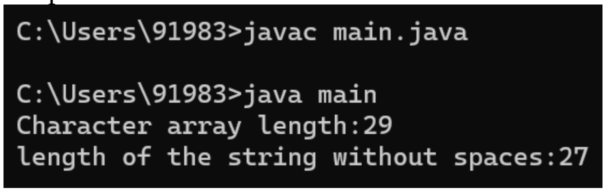


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Enter a string
debayudh
d
e
b
a
y
u
d
h
S
t
u
d
e
n
t
```

3. Find length of a string taken from keyboard and also find the length of that string except front and end spaces.
Code:

```
public class main{
public static void main(String[] args){
String s1=" OBJECT ORIENTED PROGRAMMING ";
char[] ch=s1.toCharArray();
int len=ch.length;
System.out.println("Character array length:"+len); String s2;
s2=s1.trim();
char[] p=s2.toCharArray();
int len1=p.length;
System.out.println("length of the string without spaces:"+len1);
}}
```

Output:

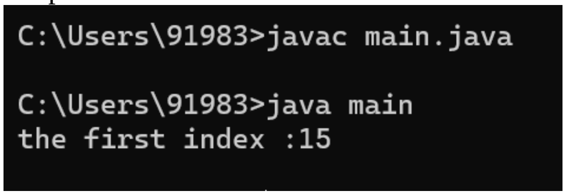


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Character array length:29
length of the string without spaces:27
```

4. Check if "Tech" presents in "University of Technology" or not. If yes return its position.
Code:

```
public class main{
public static void main(String[] args){
String s1=" UNIVERSITY OF TECHNOLOGY ";
int s2=s1.indexOf("TECH");
System.out.println("the first index :"+s2);
}}
```

Output:



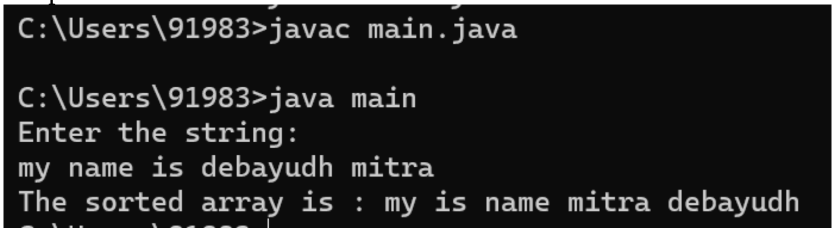
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
the first index :15
```

5. Write a program to take a sentence and convert it into string arrays and sort the words usingany sorting technique.

Code:

```
import java.util.*;
public class main{
static void sort(String []str, int n){
for (int i=1 ;i<n; i++){
String temp = str[i];
int j = i - 1;
while (j >= 0 && temp.length() < str[j].length()){
str[j+1] = str[j];
j--;
}
str[j+1] = temp;
}}
public static void main(String args[]){
Scanner sc=new Scanner(System.in);
System.out.println("Enter the string:");
String s = sc.nextLine();
String[] arr=null; arr=s.split(" ");
int len = arr.length; sort(arr,len);
System.out.print("The sorted array is : ");
for (int i=0; i<len; i++)
System.out.print(arr[i]+" ");
}}
```

Output:

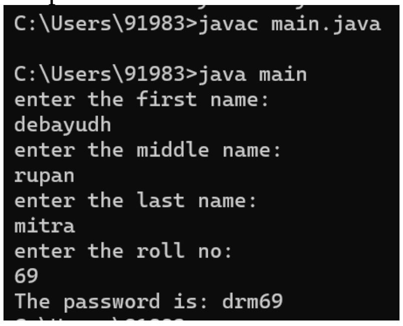


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Enter the string:
my name is debayudh mitra
The sorted array is : my is name mitra debayudh
```

6. Generate password from initials of one’s first_name, middle_name, last_name and with lastfour digit of your roll_no (if middle name is not present, it won't come).
Code:

```
import java.util.*;
class main {
public static void main(String args[]) {
Scanner sc=new Scanner(System.in);
System.out.println("enter the first name: ");
String s = sc.nextLine();
System.out.println("enter the middle name: ");
String a = sc.nextLine();
System.out.println("enter the last name: ");
String p = sc.nextLine();
System.out.println("enter the roll no: ");
int r = sc.nextInt();
while(a!=" ") {
char f=s.charAt(0);
char m=a.charAt(0);
char l=p.charAt(0);
String x=" ";
String c1=x+f+m+l;
System.out.print("The password is:");
int digit=r%10000;
String str=c1+""+digit;
System.out.print(str);
break;
    }}}
```

Output:

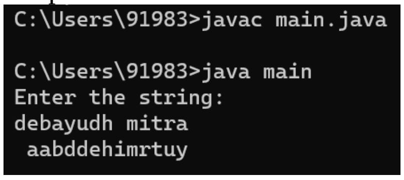


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
enter the first name:
debayudh
enter the middle name:
rupan
enter the last name:
mitra
enter the roll no:
69
The password is: drm69
```

7. Write a program in Java which will read a string and rewrite it in the alphabetical order.For example, the word STRING should be written as GINRST.
Code:

```
import java.util.*;

import java.util.Arrays;

class main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

System.out.println("Enter the string:");

String str = sc.nextLine();

char arr[] = str.toCharArray();

char temp;

int i = 0;

while (i < arr.length) {

int j = i + 1;

while (j < arr.length) {

if (arr[j] < arr[i]) {

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

} j += 1; }

i += 1; }

System.out.println(arr);
}}
```

Output:

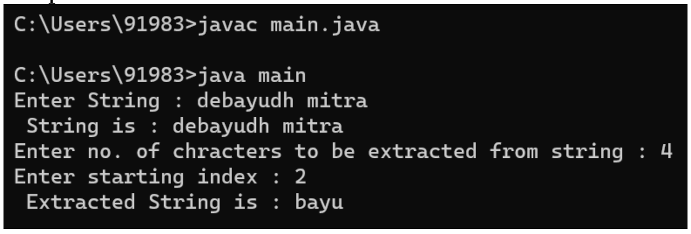


```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Enter the string:
debayudh mitra
 aabddehimrtuy
```

8. Write a program in Java to extract a portion of a character string and print the extractedstring. Assume that m characters are extracted, starting with the n-th character. The method signature will be like: voidextract(String str, int n, int m).
Code:

```
import java.util.*;

class main {

public static void main(String args[ ]) {

String str1, str2;

int m=0,n=0;

Scanner sc = new Scanner(System.in);

System.out.print("Enter String : ");
str1 = sc.nextLine();
System.out.println(" String is : "+str1);
System.out.print("Enter no. of chracters to be extracted from string : ");
m = sc.nextInt();
System.out.print("Enter starting index : ");
n = sc.nextInt();
str2=str1.substring(n,(m+n));
System.out.println(" Extracted String is : "+str2);
    }}
```

Output:



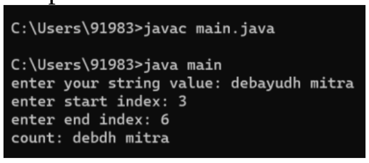
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Enter String : debayudh mitra
 String is : debayudh mitra
Enter no. of chracters to be extracted from string : 4
Enter starting index : 2
 Extracted String is : bayu
```

9. Write your own method is having a signature like String deleteMe(String str, int m)that returns the input string with the m-th element removed.

Code:

```
import java.util.Scanner;
public class main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("");
      System.out.print("enter your string value: ");
      Scanner sc = new Scanner(System.in);
      sb.append(sc.nextLine());
      System.out.print("enter start index: ");
      int start = sc.nextInt();
      System.out.print("enter end index: ");
      int end = sc.nextInt();
      System.out.println("count: "+sb.delete(start,end));
      sc.close();
        }  }
```

Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
enter your string value: debayudh mitra
enter start index: 3
enter end index: 6
count: debdh mitra
```

10. Write a program to do the following:

i) To output the question “Who is the inventor of Java”?

ii) To accept an answer.

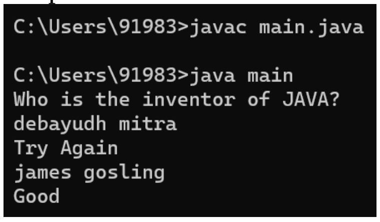
iii) To print out “Good” and then stop, if the answer is correct.

iv) To output the message “Try Again” and then stop, if the answer is wrong.
v) To display the correct answer when the answer is wrong even at the third attempts and stop.

Code:

```
import java.util.Scanner;
class main {
public static void main(String args[]) {
int i=0;
Scanner sc=new Scanner(System.in);
System.out.println("Who is the inventor of JAVA?");
for(i=1;i<=3;i++) {
String a=sc.nextLine();
if(a.equalsIgnoreCase("James Gosling")) {
System.out.println("Good");
break; }
else {
System.out.println("Try Again");
}}
if(i==4) {
System.out.println("James Gosling is the inventor of JAVA");
    }}}
```

Output:



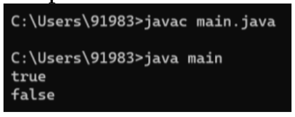
```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Who is the inventor of JAVA?
debayudh mitra
Try Again
james gosling
Good
```

11. Show that the String class type objects are immutable but StringBuffer class objects are mutable.

Code:

```
public class main {
   public static void main(String[] args) {
      String s1 = "JAVA";
      String s2 = "JAVA";
      System.out.println(s1 == s2);
      s1 = s1 + "J2EE";
      System.out.println(s1 == s2);
        }}
```

Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
true
false
```

12. Write a program in Java to create a StringBuffer object with content “Object Programming languages”. Then perform the following operations:

i) Print the object and determine its length and capacity.

ii) Inset a new string “Oriented” after the word “Object”. Print the modified string.

iii) Insert a character ‘-’ at the 6-th position. Print the modified string again.

iv) Append another string named “ are very popular.” Print the resulting string.

v) Now delete the character ‘-’ and put a space (“ ”) there. Print it.

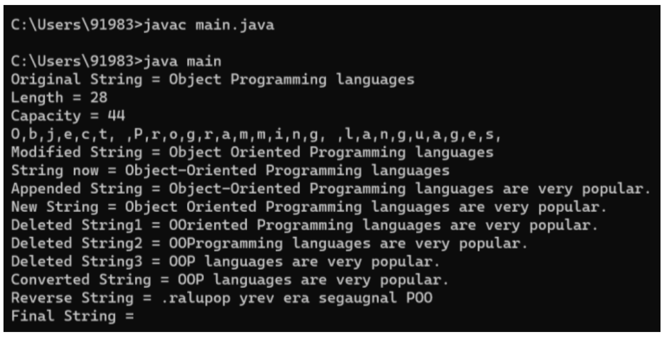
vi) Apply these operations one-by-one: (a) delete(1, 7), (b) delete(2, 10), and (c) delete(3, 13). Then print the result.

vii) Convert the StringBuffer type object into a String object. Print the final result.

Code:

```
public class main {
    public static void main(String[] args) {
        StringBuffer sb = new StringBuffer("Object Programming languages");
        System.out.println("Original String = " + sb);
        System.out.println("Length = " + sb.length());
        System.out.println("Capacity = " + sb.capacity());
        for (int i = 0; i < sb.length(); i++) {
            System.out.print(sb.charAt(i)+ ","); }
        System.out.println();
        int pos = sb.indexOf(" Programming");
        sb.insert(pos, " Oriented");
        System.out.println("Modified String = " + sb);
        sb.setCharAt(6, '-');
        System.out.println("String now = " + sb);
        sb.append(" are very popular.");
        System.out.println("Appended String = " + sb);
        StringBuffer sb1 = sb;
        sb1.deleteCharAt(6);
        sb1.insert(6, ' ');
        System.out.println("New String = " + sb1);
        sb.delete(1, 7);
        System.out.println("Deleted String1 = " + sb);
        sb.delete(2, 10);
        System.out.println("Deleted String2 = " + sb);
        sb.delete(3, 13);
        System.out.println("Deleted String3 = " + sb);
        String s = sb.toString();
        System.out.println("Converted String = " + s);
```

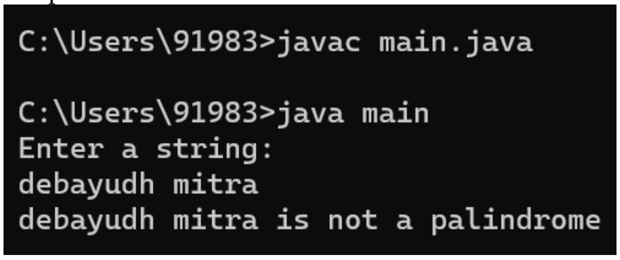
```
 sb.reverse();
 System.out.println("Reverse String = " + sb);
 sb.setLength(0);
 System.out.println("Final String = " + sb);
   }}
```
Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Original String = Object Programming languages
Length = 28
Capacity = 44
O,b,j,e,c,t, ,P,r,o,g,r,a,m,m,i,n,g, ,l,a,n,g,u,a,g,e,s,
Modified String = Object Oriented Programming languages
String now = Object-Oriented Programming languages
Appended String = Object-Oriented Programming languages are very popular.
New String = Object Oriented Programming languages are very popular.
Deleted String1 = OOriented Programming languages are very popular.
Deleted String2 = OOProgramming languages are very popular.
Deleted String3 = OOP languages are very popular.
Converted String = OOP languages are very popular.
Reverse String = .ralupop yrev era segaugnal POO
Final String =
```

13. Write a program in Java that checks whether a given string is a palindrome or not. Ignorethe cases.
Code:

```
import java.util.Scanner;
class main {
  public static void main(String args[]) {
    String str, rev = "";
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a string:");
    str = sc.nextLine();
    int length = str.length();
    for ( int i = length - 1; i >= 0; i-- )
      rev = rev + str.charAt(i);
    if (str.equals(rev))
      System.out.println(str+" is a palindrome");
    else
      System.out.println(str+" is not a palindrome");
     }}
```
Output:



```
C:\Users\91983>javac main.java

C:\Users\91983>java main
Enter a string:
debayudh mitra
debayudh mitra is not a palindrome
```

69_Debayudh Mitra

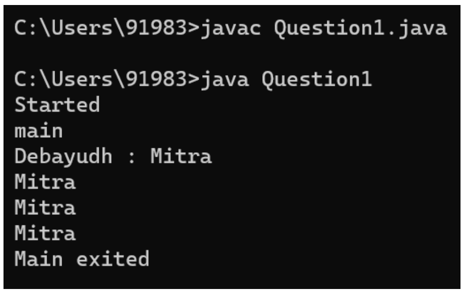
ASSIGNMENT – 7

1. Inherit a class from Thread and override the run( ) method. Inside run( ), print name of thread , and then call sleep( ). Repeat this three times, then return from run( ). Put a start-up message in the constructor. Make your thread object and main thread run to see what happens.

Code:

```
public class Question1 extends Thread {
    Question1(String name) {
        super(name);
        System.out.println("Started"); }
    public void run() {
        for (int i = 0; i < 3; i++) {
            System.out.println(this.getName());
            try {
                this.sleep(1000);
            } catch (Exception e) {
                System.out.println(e);
            } } }
    public static void main(String[] args) {
        Question1 q1 = new Question1("Mitra");
        System.out.println(Thread.currentThread().getName());
        System.out.println("Debayudh : " + q1.getName());
        q1.start();
        try {
            q1.join();
        } catch (Exception e) {
            System.out.println(e); }
        System.out.println("Main exited");
    } }
```

Output:



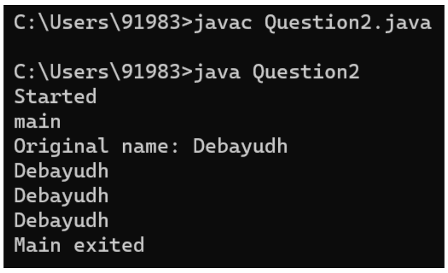
```
C:\Users\91983>javac Question1.java

C:\Users\91983>java Question1
Started
main
Debayudh : Mitra
Mitra
Mitra
Mitra
Main exited
```

2. Implement a class from Runnable and override the run( ) method. Inside run( ), print full qualified name of thread, and then call sleep( ). Repeat this three times, then return from run( ). Put a start-up message in the constructor. Make your thread object and main thread run to see what happens.

Code:

```
public class Question2 implements Runnable {
   Question2(String name) {
      System.out.println("Started"); }
   public void run() {
      for (int i = 0; i < 3; i++) {
         System.out.println(Thread.currentThread().getName());
         try {
            Thread.sleep(1000);
         } catch (Exception e) {
            System.out.println(e);
         }}}
   public static void main(String[] args) {
      Question2 q2 = new Question2("New Thread");
      Thread t = new Thread(q2, "Debayudh");
      System.out.println(Thread.currentThread().getName());
      System.out.println("Original name: " + t.getName());
      t.start();
      try {
         t.join();
      } catch (Exception e) {
         System.out.println(e); }
      System.out.println("Main exited");
   }}
```

Output:



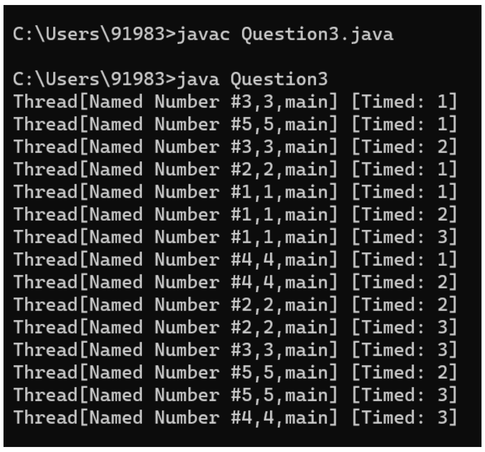
```
C:\Users\91983>javac Question2.java

C:\Users\91983>java Question2
Started
main
Original name: Debayudh
Debayudh
Debayudh
Debayudh
Main exited
```

3. Make several threads (say 5) with names (by extending thread), set their priority and run them to see what happens.

Code:

```
public class Question3 extends Thread {
   Question3(String naam) {
        super(naam); }
    public void run() {
        for (int i = 0; i < 3; i++) {
            System.out.println(this + " [Timed: " + (i + 1) + "]");
        }}
    public static void main(String[] args) {
        Question3 qs[] = new Question3[5];
        for (int i = 1; i < 6; i++) {
            qs[i - 1] = new Question3("Named Number #" + i);
            qs[i - 1].setPriority(i);
            qs[i - 1].start();
        }}}
```

Output:



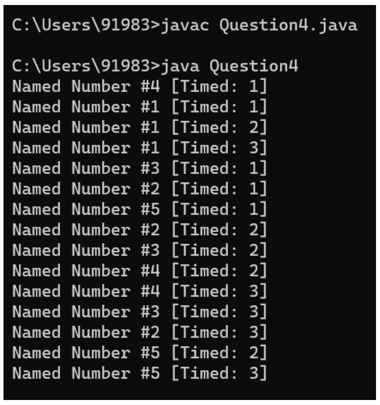
```
C:\Users\91983>javac Question3.java

C:\Users\91983>java Question3
Thread[Named Number #3,3,main] [Timed: 1]
Thread[Named Number #5,5,main] [Timed: 1]
Thread[Named Number #3,3,main] [Timed: 2]
Thread[Named Number #2,2,main] [Timed: 1]
Thread[Named Number #1,1,main] [Timed: 1]
Thread[Named Number #1,1,main] [Timed: 2]
Thread[Named Number #1,1,main] [Timed: 3]
Thread[Named Number #4,4,main] [Timed: 1]
Thread[Named Number #4,4,main] [Timed: 2]
Thread[Named Number #2,2,main] [Timed: 2]
Thread[Named Number #2,2,main] [Timed: 3]
Thread[Named Number #3,3,main] [Timed: 3]
Thread[Named Number #5,5,main] [Timed: 2]
Thread[Named Number #5,5,main] [Timed: 3]
Thread[Named Number #4,4,main] [Timed: 3]
```

4. Make several threads (say 5) with their names (implementing Runnable) set their priority and run them to see what happens.

Code:

```
public class Question4 implements Runnable {
    public void run() {
        for (int i = 0; i < 3; i++) {
            System.out.println(Thread.currentThread().getName() + " [Timed: " + (i + 1) + "]");
        }}
    public static void main(String[] args) {
        Thread qs[] = new Thread[5];
        for (int i = 1; i < 6; i++) {
            qs[i - 1] = new Thread(new Question4(), "Named Number #" + i);
            qs[i - 1].setPriority(i);
            qs[i - 1].start();
        }}}
```

Output:



```
C:\Users\91983>javac Question4.java

C:\Users\91983>java Question4
Named Number #4 [Timed: 1]
Named Number #1 [Timed: 1]
Named Number #1 [Timed: 2]
Named Number #1 [Timed: 3]
Named Number #3 [Timed: 1]
Named Number #2 [Timed: 1]
Named Number #5 [Timed: 1]
Named Number #2 [Timed: 2]
Named Number #3 [Timed: 2]
Named Number #4 [Timed: 2]
Named Number #4 [Timed: 3]
Named Number #3 [Timed: 3]
Named Number #2 [Timed: 3]
Named Number #5 [Timed: 2]
Named Number #5 [Timed: 3]
```

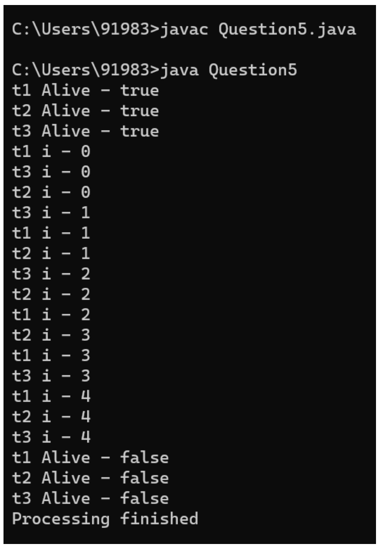
5. Write a program to use join() and isAlive() in Multi-Threading.

Code:

```
class MyRunnableClass implements Runnable {
   public void run() {
      for (int i = 0; i < 5; i++) {
         System.out.println(Thread.currentThread().getName() + " i - " + i);
         try {
            Thread.sleep(100);
         } catch (InterruptedException e) {
            e.printStackTrace();
         }}}}
public class Question5 {
   public static void main(String[] args) {
      Thread t1 = new Thread(new MyRunnableClass(), "t1");
      Thread t2 = new Thread(new MyRunnableClass(), "t2");
      Thread t3 = new Thread(new MyRunnableClass(), "t3");
      t1.start();
      t2.start();
      t3.start();
      System.out.println("t1 Alive - " + t1.isAlive());
      System.out.println("t2 Alive - " + t2.isAlive());
      System.out.println("t3 Alive - " + t3.isAlive());
      try {
         t1.join();
         t2.join();
         t3.join();
      } catch (InterruptedException e) {
         e.printStackTrace(); }
```

```
        System.out.println("t1 Alive - " + t1.isAlive());
        System.out.println("t2 Alive - " + t2.isAlive());
        System.out.println("t3 Alive - " + t3.isAlive());
        System.out.println("Processing finished");
    }}
```

Output:



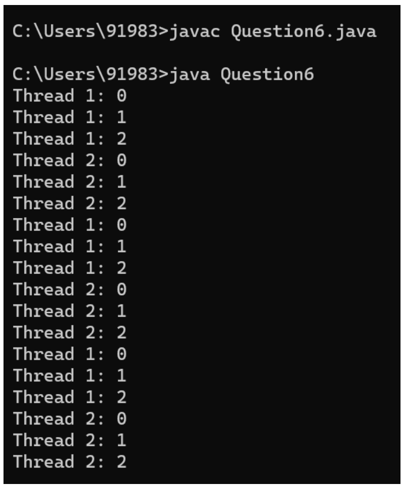
```
C:\Users\91983>javac Question5.java

C:\Users\91983>java Question5
t1 Alive - true
t2 Alive - true
t3 Alive - true
t1 i - 0
t3 i - 0
t2 i - 0
t3 i - 1
t1 i - 1
t2 i - 1
t3 i - 2
t2 i - 2
t1 i - 2
t2 i - 3
t1 i - 3
t3 i - 3
t1 i - 4
t2 i - 4
t3 i - 4
t1 Alive - false
t2 Alive - false
t3 Alive - false
Processing finished
```

6. Implement a program of locking of a common method by several threads. (Using the keyword ‘synchronized’).

Code:

```
public class Question6 implements Runnable {
    public void run() {
        for (int i = 0; i < 3; i++) {
            showMe();
            try {
                Thread.sleep(500);
            } catch (Exception e) {
                System.out.println(e);
            }}}
    synchronized void showMe() {
        for (int i = 0; i < 3; i++) {
            System.out.println(Thread.currentThread().getName() + ": " + i);
            try {
                Thread.sleep(100);
            } catch (Exception e) {
                System.out.println(e);
            }}}
    public static void main(String[] args) throws InterruptedException {
        Question6 base = new Question6();
        Thread t1 = new Thread(base, "Thread 1");
        Thread t2 = new Thread(base, "Thread 2");
        t1.start();
        t2.start();
        t1.join();
        t2.join();
    }}
```

Output:



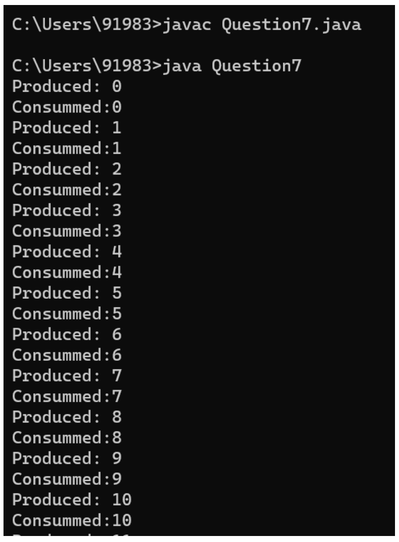
```
C:\Users\91983>javac Question6.java

C:\Users\91983>java Question6
Thread 1: 0
Thread 1: 1
Thread 1: 2
Thread 2: 0
Thread 2: 1
Thread 2: 2
Thread 1: 0
Thread 1: 1
Thread 1: 2
Thread 2: 0
Thread 2: 1
Thread 2: 2
Thread 1: 0
Thread 1: 1
Thread 1: 2
Thread 2: 0
Thread 2: 1
Thread 2: 2
```

7. Write a program to implement inter-thread communication the consumer consumes items produced by the producer with proper synchronization.

Code:

```
class ItemQueue {

    int item;

    boolean valueSet = false;

    synchronized int getItem() {

        while (!valueSet)

            try {

                wait();

            } catch (InterruptedException e) {

                System.out.println("InterruptedException caught"); }

        System.out.println("Consummed:" + item);

        valueSet = false;
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            System.out.println("InterruptedException caught"); }
        notify();
        return item; }
    synchronized void putItem(int item) {
        while (valueSet)
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught"); }
        this.item = item;
        valueSet = true;
        System.out.println("Produced: " + item);
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
        }
        notify(); }}
class Producer implements Runnable {
    ItemQueue itemQueue;

    Producer(ItemQueue itemQueue) {
        this.itemQueue = itemQueue;
        new Thread(this, "Producer").start(); }
    public void run() {
        int i = 0;
        while (true) {
            itemQueue.putItem(i++);
        }}}
class Consumer implements Runnable {
    ItemQueue itemQueue;
    Consumer(ItemQueue itemQueue) {
        this.itemQueue = itemQueue;
        new Thread(this, "Consumer").start(); }
    public void run() {
        while (true) {
            itemQueue.getItem();
        }}}
class Question7 {
    public static void main(String args[]) {
        ItemQueue itemQueue = new ItemQueue();
        new Producer(itemQueue);
        new Consumer(itemQueue);
    }}
```

Output:

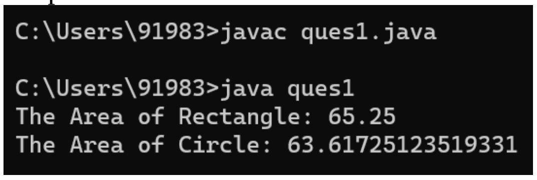


```
C:\Users\91983>javac Question7.java

C:\Users\91983>java Question7
Produced: 0
Consummed:0
Produced: 1
Consummed:1
Produced: 2
Consummed:2
Produced: 3
Consummed:3
Produced: 4
Consummed:4
Produced: 5
Consummed:5
Produced: 6
Consummed:6
Produced: 7
Consummed:7
Produced: 8
Consummed:8
Produced: 9
Consummed:9
Produced: 10
Consummed:10
```

69_Debayudh Mitra

ASSIGNMENT – 8

1. Create an interface named Shape with a field pie (=3.14). Create two subclasses of it named Circle and Rectangle create object of the two classes and calculate their area.
   Code:
```
interface Shape {
public double pi = Math.PI;
public double getArea(); }
class Rectangle implements Shape {
public double length;
public double breadth;
public Rectangle(double length, double breadth) {
this.length = length;
this.breadth = breadth; }
public double getArea() {
return length * breadth;
}}
class Circle implements Shape {
public double radius;
public Circle(double radius) {
this.radius = radius; }
public double getArea() {
return pi * Math.pow(radius, 2);
}}
public class ques1 {
public static void main(String[] args) {
Rectangle rectangle = new Rectangle(7.5, 8.7);
Circle circle = new Circle(4.5);
System.out.println("The Area of Rectangle: " + rectangle.getArea());
System.out.println("The Area of Circle: " + circle.getArea());
}}
```
   Output:

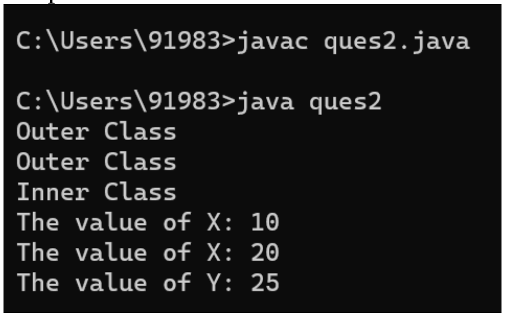


```
C:\Users\91983>javac ques1.java

C:\Users\91983>java ques1
The Area of Rectangle: 65.25
The Area of Circle: 63.61725123519331
```

2. Create a class which contains an inner class. Show that inner class can use member of outer class directly, but Outer class can use member of Inner class only through its object. Check the name of class file, you created.
   Code:
```
class Outer {
int x;
Outer(int x) {
this.x = x;
System.out.println("Outer Class"); }
void outerShow() {
System.out.println("The value of X: " + x); }
class Inner { int y;
Inner(int y) {
this.y = y;
System.out.println("Inner Class"); }
void innerShow() {
```

```
System.out.println("The value of X: " + x);
System.out.println("The value of Y: " + y);
}}}
public class ques2 {
public static void main(String[] args) {
Outer out = new Outer(10);
Outer.Inner in = new Outer(20).new Inner(25);
out.outerShow();
in.innerShow();
}}
```

Output:



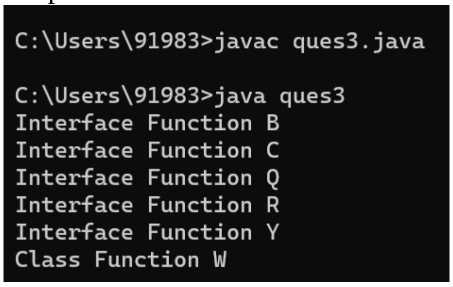
```
C:\Users\91983>javac ques2.java

C:\Users\91983>java ques2
Outer Class
Outer Class
Inner Class
The value of X: 10
The value of X: 20
The value of Y: 25
```

3. Create two interfaces, each with two methods. Inherit a new interface from the two, adding a new method. Create a class by implementing the new interface and also inheriting from a concrete class. In main() method, create an object of derived class and call the methods [do all without package statement].

Code:

```
interface A {
public void B();
public void C(); }
interface P {
public void Q();
public void R(); }
interface X extends A, P {
public void Y(); }
class Z {
public void W() {
System.out.println("Class Function W");
}}
class Inherit extends Z implements X {
public void B() {
System.out.println("Interface Function B"); }
public void C() {
System.out.println("Interface Function C"); }
public void Q() {
System.out.println("Interface Function Q"); }
public void R() {
System.out.println("Interface Function R"); }
public void Y() {
System.out.println("Interface Function Y");
}}
public class ques3 {
public static void main(String[] args) {
Inherit in = new Inherit();
in.B();         in.C();  in.Q();
in.R();         in.Y();  in.W();
```

}}
Output:

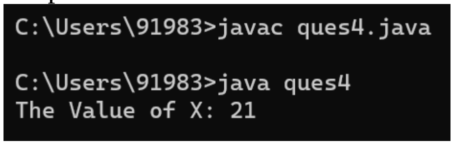


```
C:\Users\91983>javac ques3.java

C:\Users\91983>java ques3
Interface Function B
Interface Function C
Interface Function Q
Interface Function R
Interface Function Y
Class Function W
```

4. Write a program to demonstrate anonymous inner class (using super class and interface).
Code:

```
interface A {
public int x = 21;
public void getValue(); }
public class ques4 {
public static void main(String[] args) {
A a = new A() {
public void getValue() {
System.out.println("The Value of X: " + x);
}};
a.getValue();
}}
```

Output:

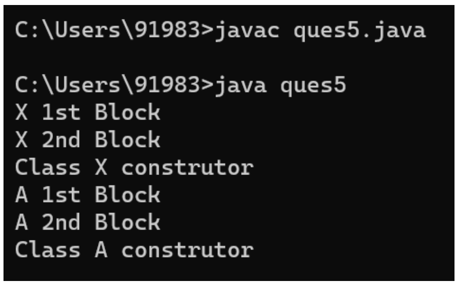


```
C:\Users\91983>javac ques4.java

C:\Users\91983>java ques4
The Value of X: 21
```

5. Show that an ordinary block is executed when an object is created and also the order of execution of these blocks (for multiple blocks/ inherited blocks).
Code:

```
class X {{
System.out.println("X 1st Block"); }
X() {
System.out.println("Class X construtor"); }
{ System.out.println("X 2nd Block");
}}
class A extends X {{
System.out.println("A 1st Block"); }
A() {
super();
System.out.println("Class A construtor"); }
{ System.out.println("A 2nd Block");
}}
public class ques5 {
public static void main(String[] args) {
new A();
}}
```

Output:

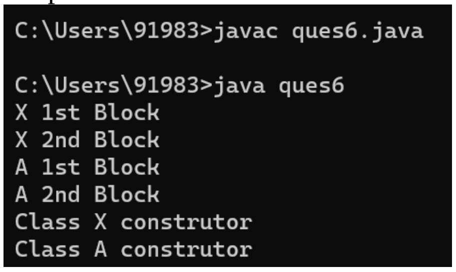


```
C:\Users\91983>javac ques5.java

C:\Users\91983>java ques5
X 1st Block
X 2nd Block
Class X construtor
A 1st Block
A 2nd Block
Class A construtor
```

6. Show that a static block is executed at the time of class loading and also the order of execution of these blocks (for multiple blocks/ inherited blocks).
   Code:

```
class X {
static {
System.out.println("X 1st Block"); }
X() {
System.out.println("Class X construtor"); }
static {
System.out.println("X 2nd Block");
}}
class A extends X {
static {
System.out.println("A 1st Block"); }
A() {
super();
System.out.println("Class A construtor"); }
static {
System.out.println("A 2nd Block");
}}
public class ques6 {
public static void main(String[] args) {
new A(); }}
```

   Output:

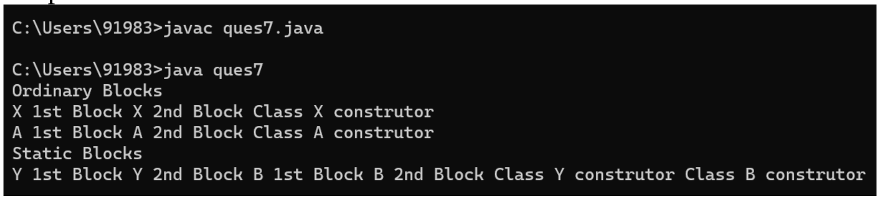


```
C:\Users\91983>javac ques6.java

C:\Users\91983>java ques6
X 1st Block
X 2nd Block
A 1st Block
A 2nd Block
Class X construtor
Class A construtor
```

7. Write a program to show the difference between ordinary block and static block.
   Code:

```
class X {{
System.out.print("X 1st Block "); }
X() {
System.out.println("Class X construtor "); }
{ System.out.print("X 2nd Block ");
}}
class A extends X {
{ System.out.print("A 1st Block "); }
A() {
super();
System.out.print("Class A construtor "); }
{ System.out.print("A 2nd Block ");
}}
class Y {
static {
System.out.print("Y 1st Block "); }
Y() {
System.out.print("Class Y construtor "); }
static {
System.out.print("Y 2nd Block ");
}}
class B extends Y {
static {
System.out.print("B 1st Block "); }
B() {
super();
System.out.println("Class B construtor ");
}
static {
System.out.print("B 2nd Block ");
}}
public class ques7 {
public static void main(String[] args) { System.out.println("Ordinary Blocks"); new A();
System.out.println("\nStatic Blocks"); new B();
}}
```

Output:



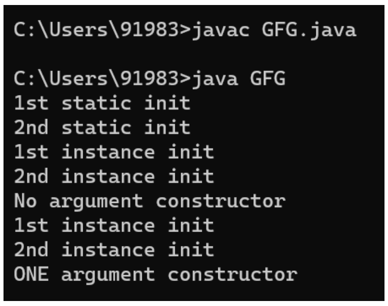
```
C:\Users\91983>javac ques7.java

C:\Users\91983>java ques7
Ordinary Blocks
X 1st Block X 2nd Block Class X construtor
A 1st Block A 2nd Block Class A construtor
Static Blocks
Y 1st Block Y 2nd Block B 1st Block B 2nd Block Class Y construtor Class B construtor
```

8. Write a program to demonstrate the order of execution among the parent and child class's static and non-static blocks.

Code:

```
class GFG {
    GFG(int x) {
            System.out.println("ONE argument constructor"); }
    GFG() {
            System.out.println("No argument constructor"); }
    static {
            System.out.println("1st static init"); }
    { System.out.println("1st instance init"); }
    { System.out.println("2nd instance init"); }
    static {
            System.out.println("2nd static init"); }
    public static void main(String[] args) {
            new GFG();
            new GFG(8);
    }}
```

Output:



```
C:\Users\91983>javac GFG.java

C:\Users\91983>java GFG
1st static init
2nd static init
1st instance init
2nd instance init
No argument constructor
1st instance init
2nd instance init
ONE argument constructor
```

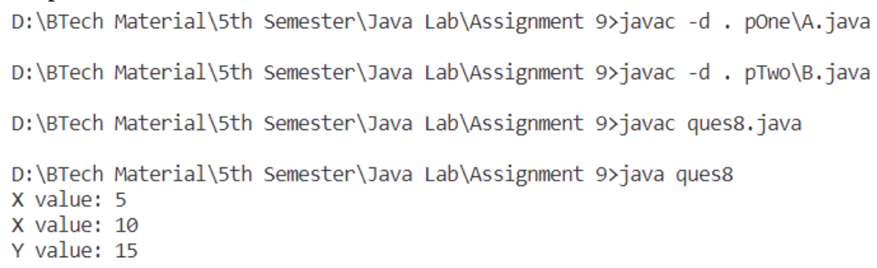
9. Create a class with variable(s) and method(s) (all will be default accessed) under package pOne. Now create a class under package pTwo, which is a subclass of the firstly created class. In the method here (i.e. class of pTwo) call variable(s) and method(s) of previous class (i.e. class of pOne). If errors come, rectify them. Now from Main (under working directory) check access to second class's members.

   Code:

```
package pOne;
public class A {
public int x;
public A(int x) {
this.x = x; }
public void showX() {
System.out.println("X value: " + x);
}}
package pTwo;
import pOne.*;
public class B extends A {
public int y;
public B(int x, int y) {
super(x);
this.y = y; }
public void showY() {
super.showX();
System.out.println("Y value: " + y);
}}
import pOne.A;
import pTwo.B;
public class ques8 {
public static void main(String[] args) {
A a = new A(5);
B b = new B(10, 15);
a.showX();
b.showY();
}}
package pOne;
public class A {
public int x;
public A(int x) {
this.x = x; }
public void showX() {
System.out.println("X value: " + x);
}}
package pTwo;
import pOne.*;
public class B extends A {
```

```
public int y;
public B(int x, int y) {
super(x);
this.y = y; }
public void showY() {
super.showX();
System.out.println("Y value: " + y);
}}
import pOne.A;
import pTwo.B;
public class ques8 {
public static void main(String[] args) {
A a = new A(5);
B b = new B(10, 15);
a.showX();
b.showY();
}}
```

Output:



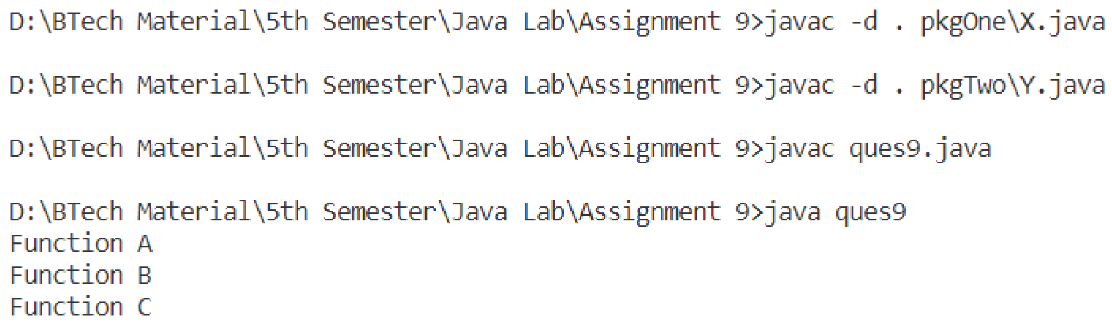
```
D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac -d . pOne\A.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac -d . pTwo\B.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac ques8.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>java ques8
X value: 5
X value: 10
Y value: 15
```

10. Create an interface containing three methods, in a package ‘pkgOne’. Implement the interface from a class under package pkgTwo. From the main(), under the working directory, create an object of the class and call methods of the interface.

Code:

```
package pkgOne;
public interface X {
public void A();
public void B();
public void C(); }
package pkgTwo;
import pkgOne.*;
public class Y implements X {
public void A() {
System.out.println("Function A"); }
public void B() {
System.out.println("Function B"); }
public void C() {
System.out.println("Function C");
}}
import pkgTwo.Y;
public class ques9 {
public static void main(String[] args) {
Y y = new Y();
y.A();
y.B();
y.C();
}}
```

Output:



```
D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac -d . pkgOne\X.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac -d . pkgTwo\Y.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>javac ques9.java

D:\BTech Material\5th Semester\Java Lab\Assignment 9>java ques9
Function A
Function B
Function C
```