Lab File

Java Programming

(IT 201)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted to:

Dr Supriya Raheja

Ast. Professor

CSE Department, ASET

Submitted by:

Shaina Mehta

A2305219268

B.tech. C.S.E.

4CSE-4Y

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY UTTAR PRADESH

NOIDA -201301

Exp No	Assignment Category	Code	Name of Experiment	Date of Allotment	Date of Evaluation	Max Marks	Marks Obtained	Faculty Sign
1			W.A.P. to check whether a number is a prime or not.	24-12-2020	15-03-2021			
2			W.A.P. to print the Fibonacci series up to a limit.	24-12-2020	15-03-2021			
3			W.A.P. to check whether a number is even or odd.	24-12-2020	15-03-2021			
4			W.A.P. to check if a character is a vowel or consonant; if none, display error.	24-12-2020	15-03-2021			
5			W.A.P. to check whether a year is a leap year or not.	24-12-2020	15-03-2021			
6			W.A.P. to print all the prime numbers up to a limit.	24-12-2020	15-03-2021			
7			W.A.P. to calculate a power of a number.	24-12-2020	15-03-2021			
8			Write a Java program to remove duplicate elements from an array.	24-12-2020	15-03-2021			

9			Write a Java program to create a matrix and fill it with prime numbers.	24-12-2020	15-03-2021			
10			Write a Java program to check whether a given matrix is Lower Triangular Matrix or not.	24-12-2020	15-03-2021			
11			Write a Java program to check given string is Palindrome String or not in Java.	24-12-2020	15-03-2021			
12			Write a Java program to get string and count number of words in a provided string.	24-12-2020	15-03-2021			
13			Write a Java program to divide a string in 'N' equal parts.	24-12-2020	15-03-2021			
14			Design a class to represent a bank account. Include the following members: Data members: Name of the depositor, account no, type of account and balance amount Methods: to deposit an amount. To withdraw an amount after checking balance, to display the name and balance. Use constructors to provide the initial values.	24-12-2020	15-03-2021			
15			Create a linked list of n nodes and then reverse the order of nodes.	24-12-2020	15-03-2021			
16			WAP to increment the employee salaries on the basis of their designation(Manager-5000, General Manager-10000, CEO-20000, worker-	24-12-2020	15-03-2021			

			2000). Use employee name, id, designation , salary as data member and inc_sal as member function.					
17			Assume that a bank maintains two kinds of account for its customers, one called as savings account and the other as current account. The saving account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Class account stores customer name, account number and the type of account. Include member functions:- • Accept deposit from a customer and update the balance. • Display the balance. • Compute and deposit interest. • Permit withdrawal and update balance. • Check for minimum balance, impose penalty and update the balance.	24-12-2020	15-03-2021			
18			Design three classes: Student, Exam and Result. The student class has data members such as roll no, name etc. Create a class Exam by inheriting the Student class. The Exam class adds data members representing the marks scored in six subjects. Derive the Result from class Exam and it has its own members such as	24-12-2020	15-03-2021			

			total marks and average. Calculate the total marks and average.					
19			Write a program with given interfaces MotorBike and Cycle, then implement in child class TwoWheeler and display distance & speed.	24-12-2020	15-03-2021			
20			An interface called RegularPolygon with two abstract methods: getNumSides and getSideLength. A class EquilateralTriangle that implements the interface, has getNumSides return 3 and getSideLength return an instance variable that is set by the constructor. A class Square that implements the interface, has getNumSides return 4 and getSideLength return an instance variable that is set by the constructor. Add a static totalSides method, that given a RegularPolygon[], returns the sum of the number of sides of all the elements. Add method: getPerimeter (n * length, where n is the number of sides).	24-12-2020	15-03-2021			
21			Create a class Employee. Derive 3 classes from this class namely, Programmer, Analyst & Project Leader. Take attributes and operations on your own.	24-12-2020	15-03-2021			
22			Create a class with a main( ) that throws an object of class Exception inside a try block. Give the constructor for Exception a String argument. Catch the exception inside a catch clause and print the String argument. Add a finally clause	24-12-2020	15-03-2021			

			and print a message to prove you were there.					
23			Create a program to ask the user for a real number and display its square root. Errors must be trapped using "try..catch".	24-12-2020	15-03-2021			
24			Create a try block that is likely to generate three types of exception and then incorporate necessary catch blocks to catch and handle them appropriately.	24-12-2020	15-03-2021			
25			Create a class MyClass and create three methods myMethod1(), Method2() and Method3(). Invoke Method2() from Method1() and Method3() from Method2(). Write a code that can throw an exception inside myMethod3().	24-12-2020	15-03-2021			
26			Write a program to deposit cash, withdraw in a bank using multithreading.	24-12-2020	15-03-2021			
27			Implement three classes: Storage, Counter, and Printer. The Storage class should store an integer. The Counter class should create a thread that starts counting from 0 (0, 1, 2, 3 ...) and stores each value in the Storage class. The Printer class should create a thread that keeps reading the value in the Storage class and printing it. Write a program that creates an instance of the Storage class and sets up a Counter and a Printer object to operate on it.	24-12-2020	15-03-2021			

28			Create a library system with methods for returning and issuing a book. Apply multithreading synchronization concept and exception handling.	24-12-2020	15-03-2021			
29			Write a program to do the following: • To store the text file contents line by line into an array. • To find the longest word in the text file. • To append the text to an existing file.	24-12-2020	15-03-2021			
30			Write a simple calculator using AWT.	24-12-2020	15-03-2021			
	Viva	Viva						

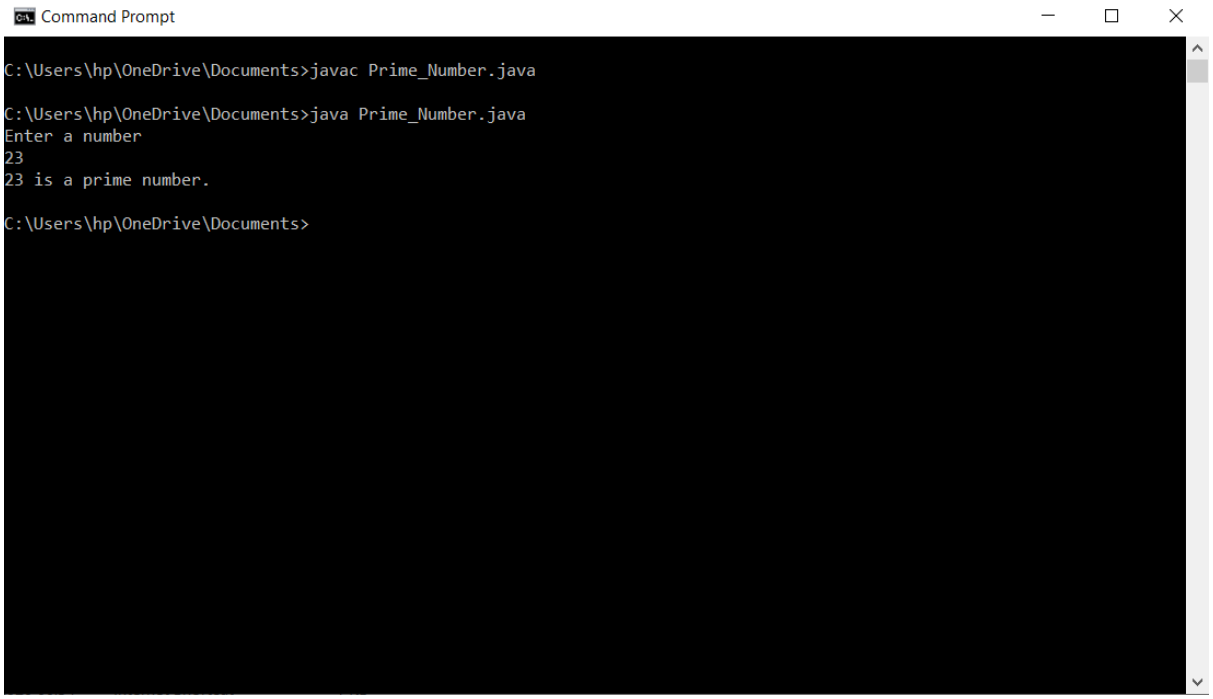
Q1) W.A.P. to check whether a number is a prime or not.

Code:

```
//Prime_Number.java
import java.util.Scanner;
public class Prime_Number
{
public static void main(String args[])
{
System.out.println("Enter a number");
Scanner input=new Scanner(System.in);
int a=input.nextInt();
int b=2;
int flag=0;
int n=a;
while(b!=n)
{
if(n%b==0)
{
flag=0;
break;
}
else
{
flag=1;
b++;
}
}
if(flag == 1)
{
System.out.println(a + " is a prime number.");
}
```

```
else
{
System.out.println(a + " is not a prime number.");
}
input.close();
}
}
```

Output:



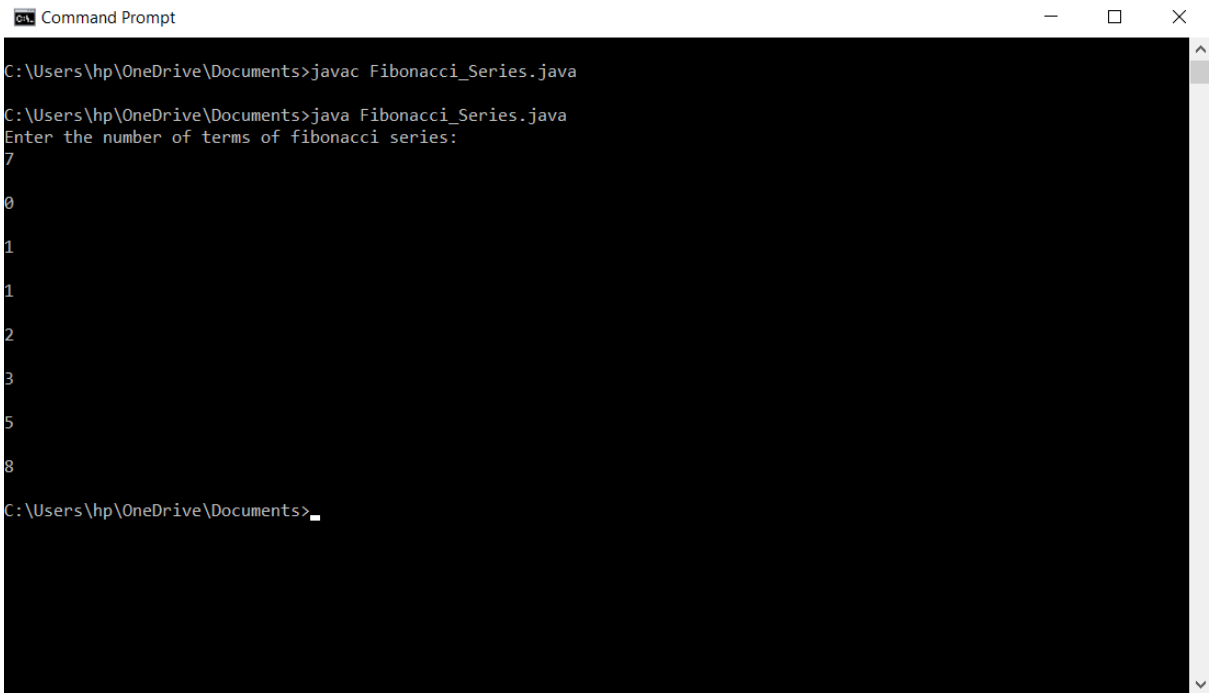
Q2) W.A.P. to print the Fibonacci series up to a limit.

Code:

```
//Fibonacci_Series.java
import java.util.Scanner;
public class Fibonacci_Series
{
public static void main(String args[])
{
System.out.println("Enter the number of terms of fibonacci series: ");
Scanner input= new Scanner(System.in);
int n=input.nextInt();
```

```
int a=0;
int b=1;
int c=1;
System.out.println("\n" + a);
System.out.println("\n" + b);
System.out.println("\n" + c);
int i=0;
while(i<n-3)
{
a=b;
b=c;
c=a+b;
System.out.println("\n" + c);
i++;
}
input.close();
}
}
```

Output:

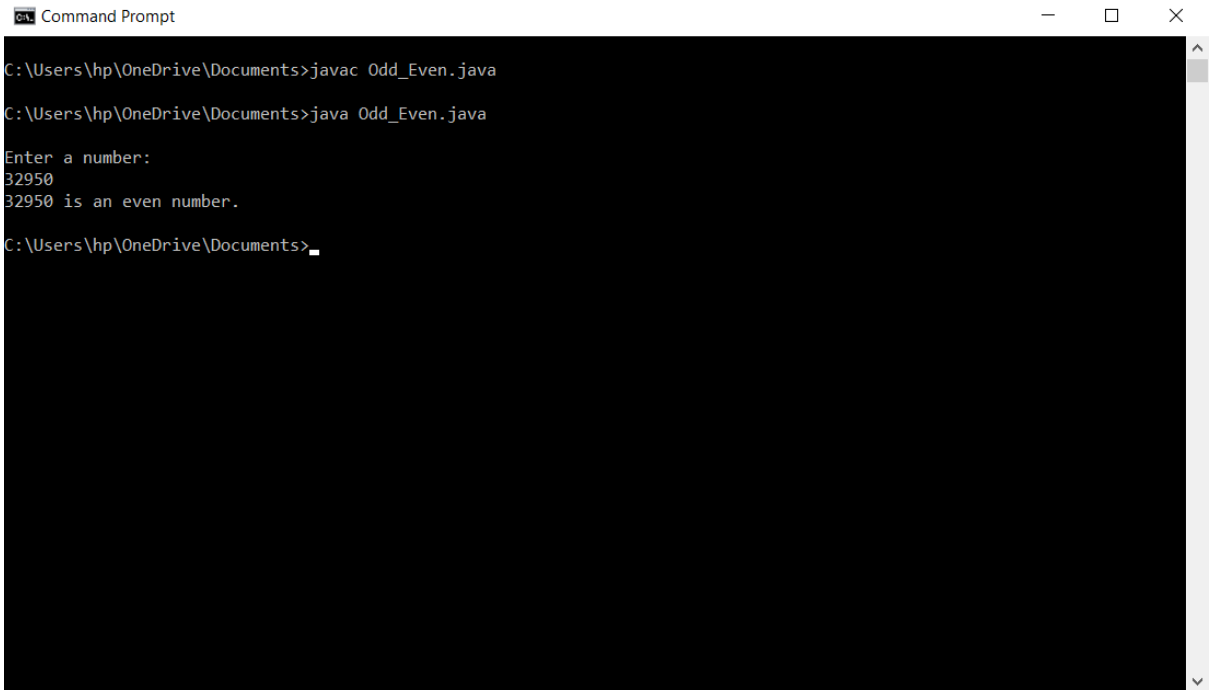


Q3) W.A.P. to check whether a number is even or odd.

Code:

```
//Odd_Evev.java
import java.util.Scanner;
public class Odd_Even
{
public static void main(String args[])
{
System.out.println("\nEnter a number: ");
Scanner input=new Scanner(System.in);
int a=input.nextInt();
if(a%2 == 0)
{
System.out.println(a + " is an even number.");
}
else
{
System.out.println(a + " is an odd number.");
}
input.close();
}
}
```

Output:



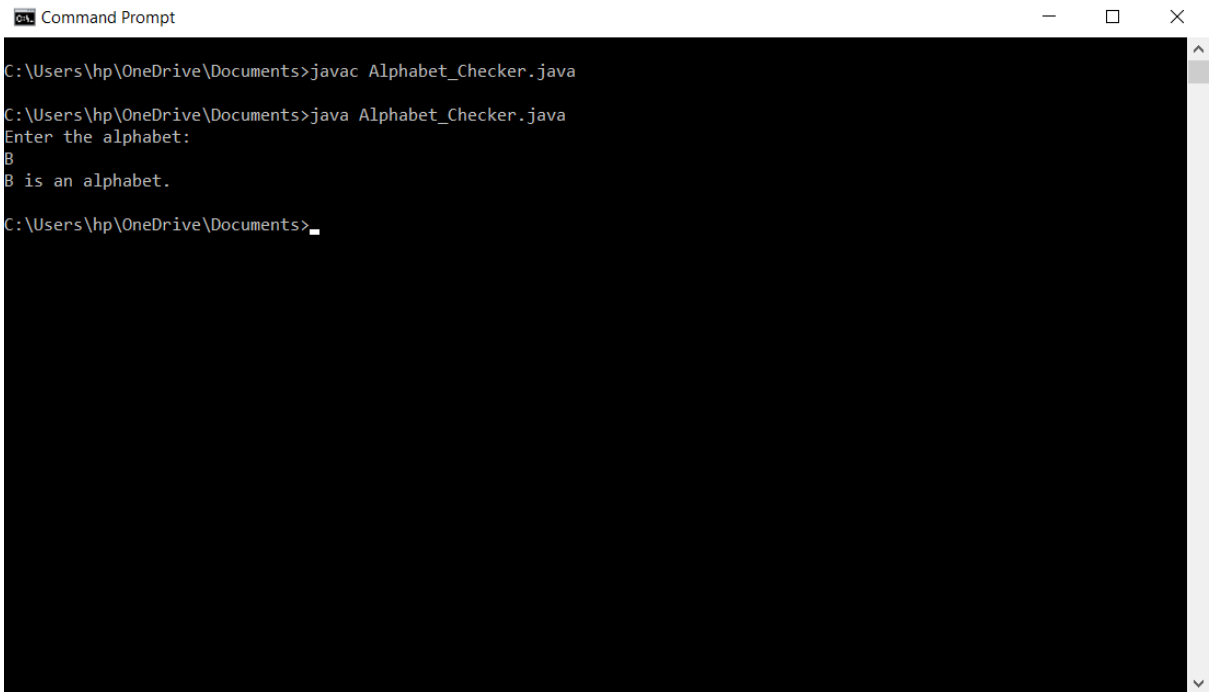
Q4) W.A.P. to check if a character is a vowel or consonant; if none, display error.

Code:

```
//Alphabet_Checker.java

import java.util.Scanner;

public class Alphabet_Checker

{

public static void main(String args[])

{

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

Scanner input= new Scanner(System.in);

char ch=input.next().charAt(0);

if(ch == 'A' || ch == 'a' || ch == 'E' || ch == 'e' || ch == 'I' || ch == 'i' || ch == 'O' || ch == 'o' || ch == 'U' ||ch == 'u')

{

System.out.println(ch + " is a vowel.");

}

else if(Character.isAlphabetic(ch))

{

System.out.println(ch + " is an alphabet.");
```

```
}
else
{
System.out.println(ch + " is not an alphabet.");
}
input.close();
}
}
```

Output:



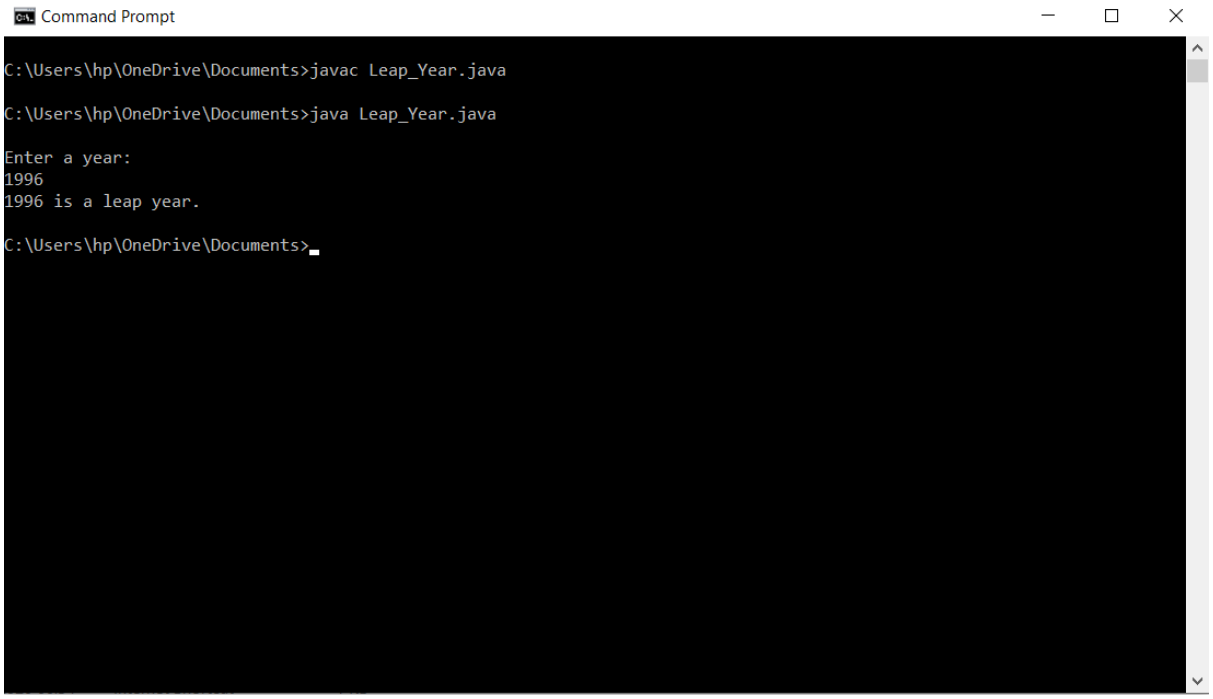
Q5) W.A.P. to check whether a year is a leap year or not.

Code:

```
//Leap_Year.java
import java.util.Scanner;
public class Leap_Year
{
public static void main(String args[])
{
System.out.println("\nEnter a year: ");
Scanner input=new Scanner(System.in);
```

```
int a=input.nextInt();
if(a%100 == 0)
{
if(a%400 ==0)
{
System.out.println(a + " is a leap year.");
}
}
else if(a%4 == 0)
{
System.out.println(a + " is a leap year.");
}
else
{
System.out.println(a + " is not a leap year.");
}
input.close();
}
}
```

Output:



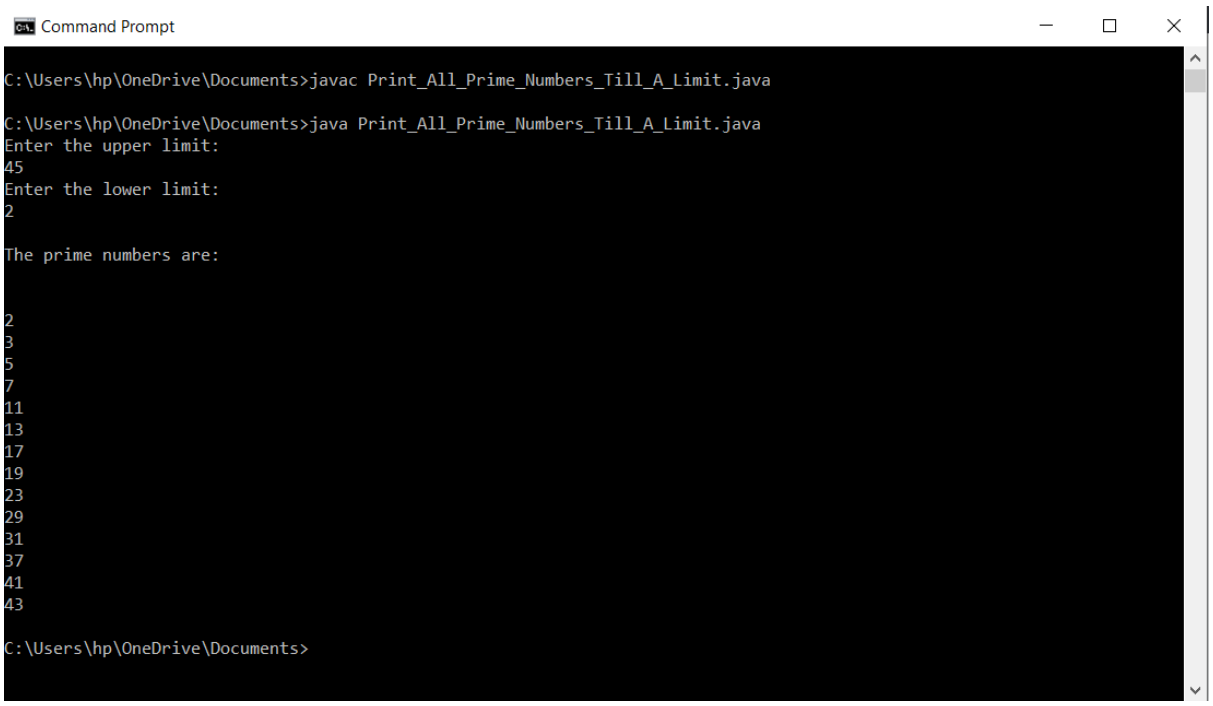
Q6) W.A.P. to print all the prime numbers up to a limit.

Code:

```
//Print_All_Prime_Numbers_Till_A_Limit.java
import java.util.Scanner;
public class Print_All_Prime_Numbers_Till_A_Limit
{
public static void main(String args[])
{
Scanner input= new Scanner(System.in);
System.out.println("Enter the upper limit: ");
int ul=input.nextInt();
System.out.println("Enter the lower limit: ");
int ll=input.nextInt();
System.out.println("\nThe prime numbers are:" );
System.out.println("\n");
for(int i=ll;i<=ul;i++)
{
int b=2;
```

```
int flag=1;
int n=i;
while(b!=n)
{
if(n%b==0)
{
flag=0;
break;
}
else
{
flag=1;
b++;
}
}
if(flag == 1)
{
System.out.println(i);
}
}
input.close();
}
}
```

Output :

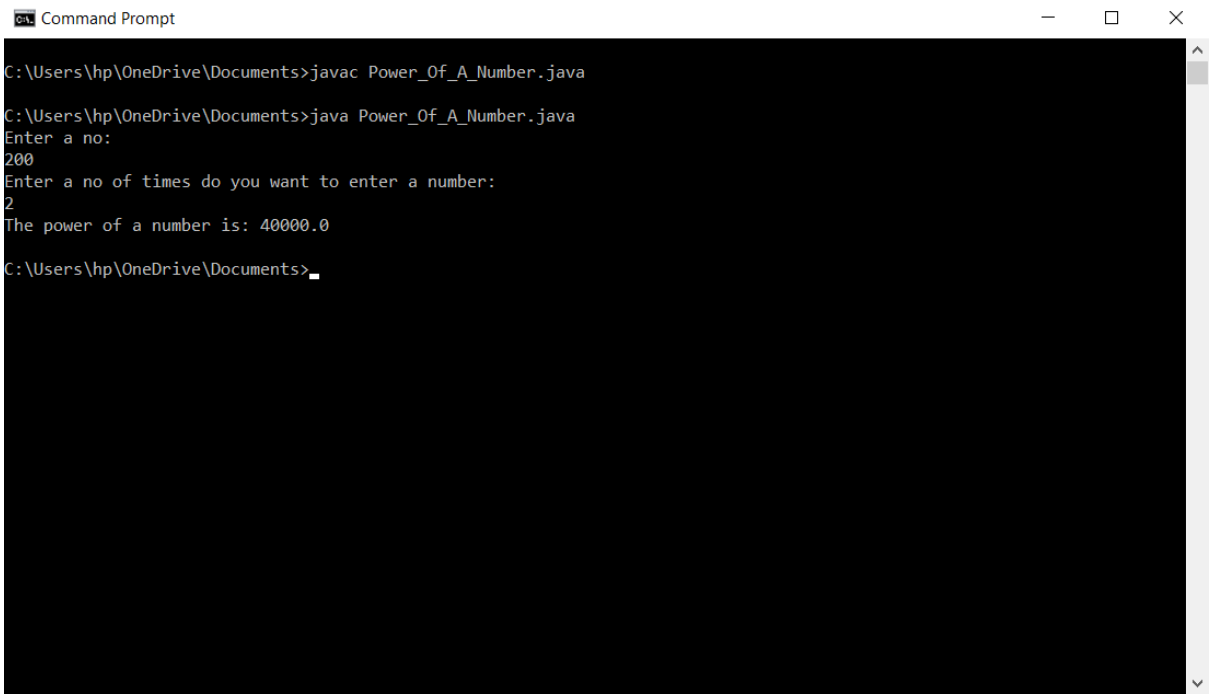


Q7) W.A.P. to calculate a power of a number.

Code:

```
//Power_Of_A_Number.java
import java.util.Scanner;
public class Power_Of_A_Number
{
public static void main(String args[])
{
System.out.println("Enter a no: ");
Scanner input= new Scanner(System.in);
int a=input.nextInt();
System.out.println("Enter a no of times do you want to enter a number: ");
int n=input.nextInt();
System.out.println("The power of a number is: " + Math.pow(a,n));
input.close();
}
}
```

Output:



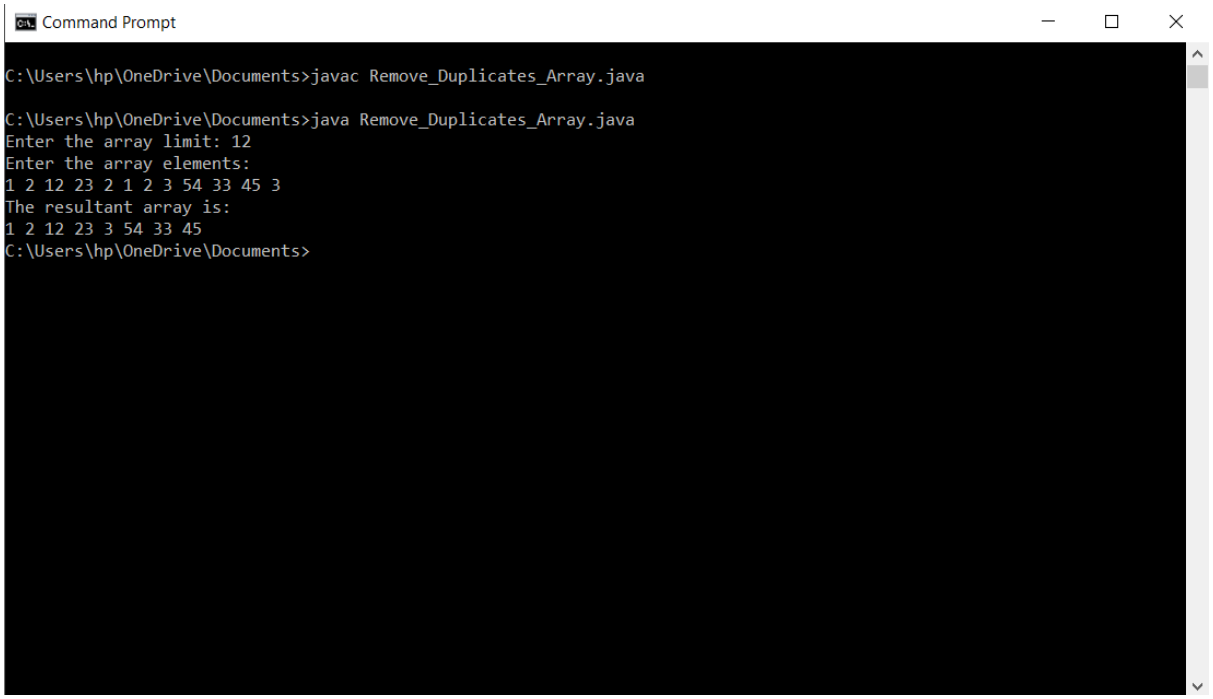
Q8) Write a Java program to remove duplicate elements from an array.

Code:

```
// Remove_Duplicates_Array.java
import java.util.Scanner;
import java.util.HashMap;
public class Remove_Duplicates_Array
{
public static void main(String[] args)
{
HashMap<Integer,Integer> map = new HashMap<>();
Scanner input = new Scanner(System.in);
System.out.print("Enter the array limit: ");
int n = input.nextInt();
int arr[] = new int[n];
System.out.println("Enter the array elements: ");
for(int i=0;i<n;i++)
{
arr[i] = input.nextInt();
```

```
}
int arr2[] = new int[100];
int ctr = 0;
for(int i=0;i<n;i++)
{
if(!map.containsKey(arr[i]))
{
map.put(arr[i],1);
arr2[ctr] = arr[i];
ctr++;
}
}
System.out.println("The resultant array is: ");
for(int i=0;i<ctr;i++)
{
System.out.print(arr2[i] + " ");
}
input.close() ;
}
}
```

Output:



Q9) Write a Java program to create a matrix and fill it with prime numbers.

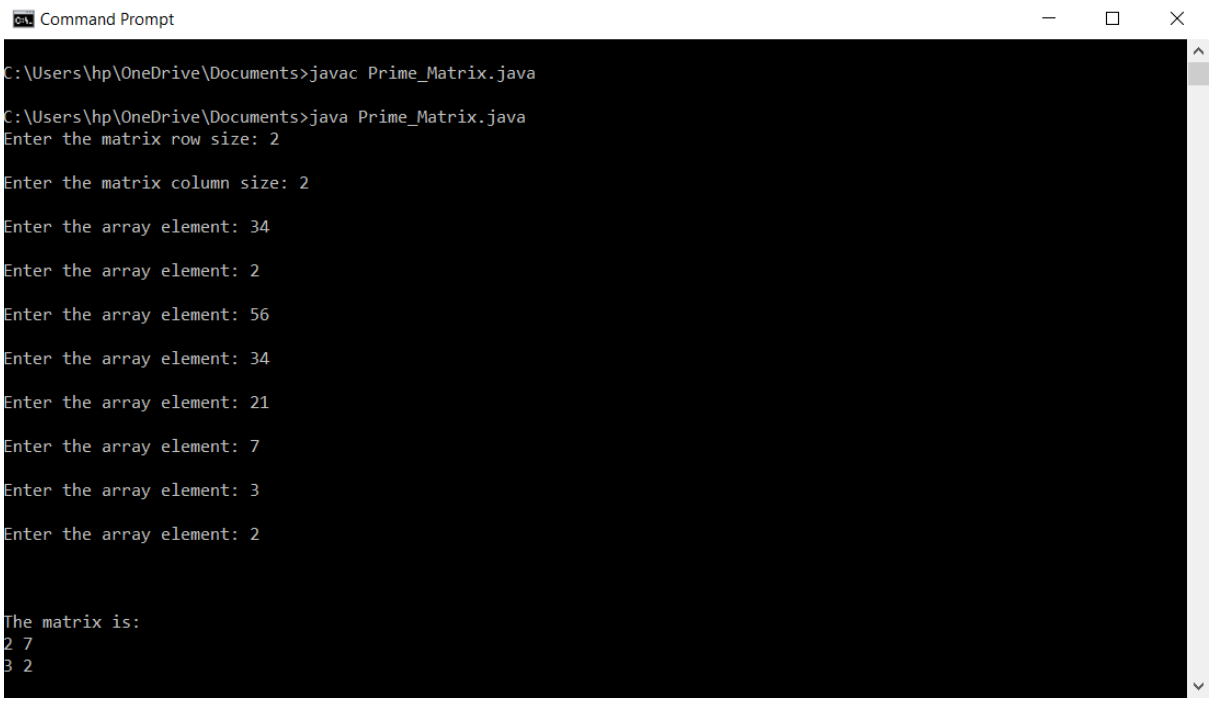
Code:

```
// Prime_Matrix.java
import java.util.Scanner;
public class Prime_Matrix
{
static boolean checkPrime(int ele)
{
int b = 2;
int n = ele;
while(b!=n)
{
if(n%b==0)
{
return false;
}
else
{
b++;
```

```
}
}
return true;
}
public static void main(String[] args)
{
Scanner input = new Scanner(System.in);
System.out.print("Enter the matrix row size: ");
int m = input.nextInt();
System.out.print("\nEnter the matrix column size: ");
int n = input.nextInt();
int arr[] = new int [m*n];
int mat[][] = new int [m][n];
int k = 0;
while(k<m*n)
{
System.out.print("\nEnter the array element: ");
int val = input.nextInt();
if(checkPrime(val))
{
arr[k] = val;
k++;
}
}
int l = 0;
for(int i=0;i<m;i++)
{
for(int j=0;j<n;j++)
{
mat[i][j] = arr[l];
l++;
```

```
}
System.out.print("\n");
}
System.out.print("\nThe matrix is:\n");
for(int i=0;i<m;i++)
{
for(int j=0;j<n;j++)
{
System.out.print(mat[i][j] + " ");
}
System.out.print("\n");
}
input.close();
}
}
```

Output:



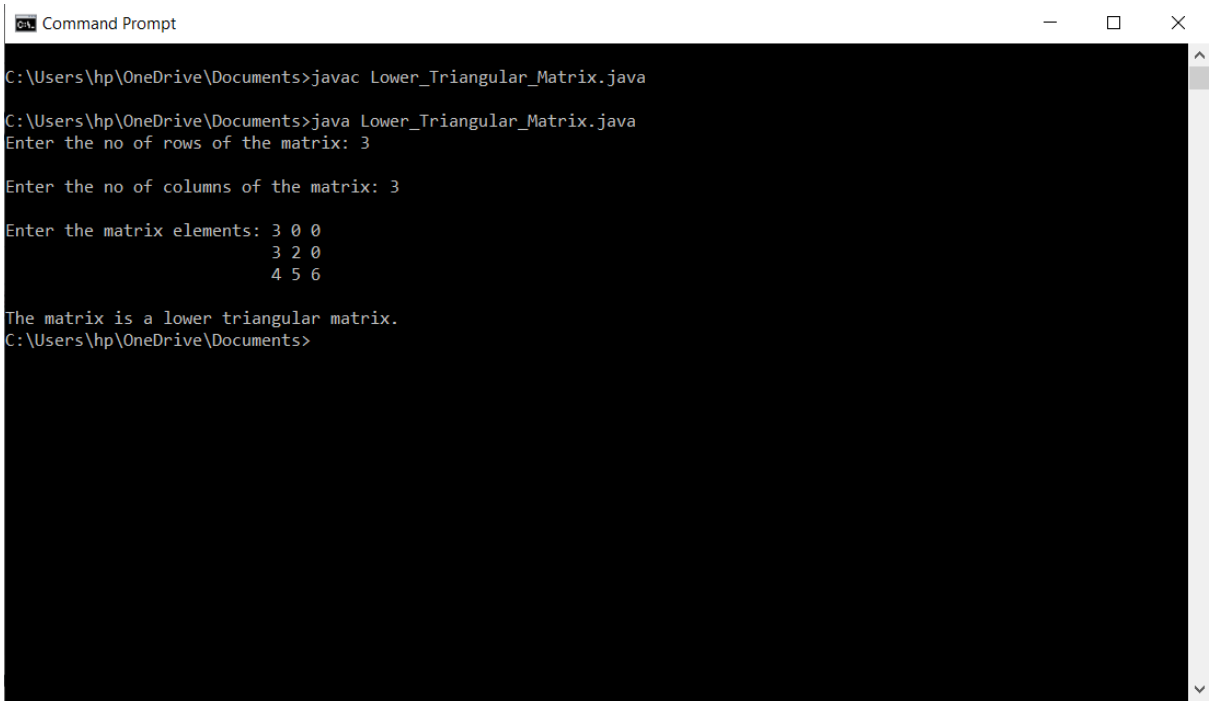
Q10) Write a Java program to check whether a given matrix is Lower Triangular Matrix or not.

Code:

```
//Lower_Triangular_Matrix.java
import java.util.Scanner;
public class Lower_Triangular_Matrix
{
static boolean check_Lower_Triangular_Matrix(int arr[][],int m,int n)
{
if(m != n)
{
return false;
}
for(int i=0;i<m-1;i++)
{
for(int j=i+1;j<n;j++)
{
if(arr[i][j]!=0)
{
return false;
}
}
}
return true;
}
public static void main(String args [])
{
Scanner input = new Scanner (System.in);
System.out.print("Enter the no of rows of the matrix: ");
int m = input.nextInt();
System.out.print("\nEnter the no of columns of the matrix: ");
int n = input.nextInt();
int arr[][] = new int [m][n];
```

```
System.out.print("\nEnter the matrix elements: ");
for(int i=0;i<m;i++)
{
for(int j=0;j<n;j++)
{
arr[i][j] = input.nextInt();
}
}
boolean b=check_Lower_Triangular_Matrix(arr,m,n);
if(b)
{
System.out.print("\nThe matrix is a lower triangular matrix.");
}
else
{
System.out.print("\nThe matrix is not a lower triangular matrix.");
}
input.close() ;
}
}
```

Output:



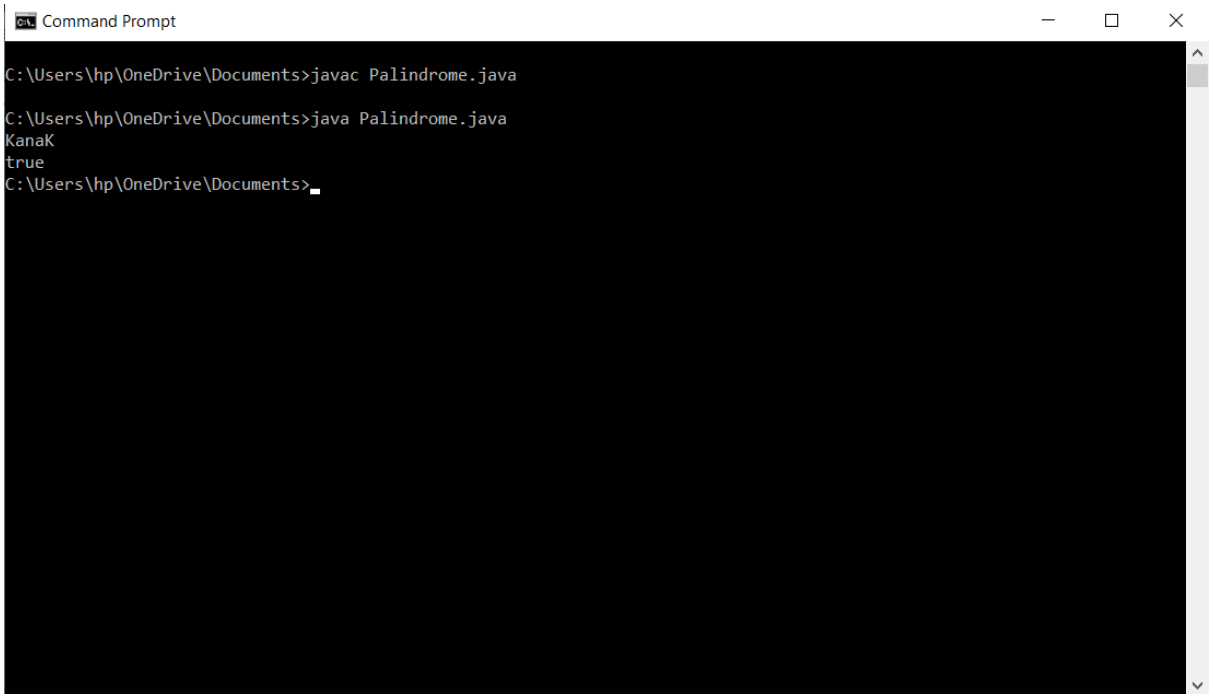
Q11) Write a Java program to check given string is Palindrome String or not in Java.

Code:

```
// Palindrome.java
import java.util.Scanner;
public class Palindrome
{
static boolean palindrome(String str, int si,int ei)
{
if(si>=ei)
{
return true;
}
char a = str.charAt(si);
char b = str.charAt(ei);
if(a != b)
{
return false;
}
boolean c = palindrome(str,si+1,ei-1);
```

```
return c;
}
public static void main(String[] args)
{
Scanner input = new Scanner (System.in);
String str = input.nextLine();
int si = 0;
int ei = str.length()-1;
boolean b = palindrome(str,si,ei);
if(b)
{
System.out.print("true");
}
else
{
System.out.print("false");
}
input.close();
}
}
```

Output:



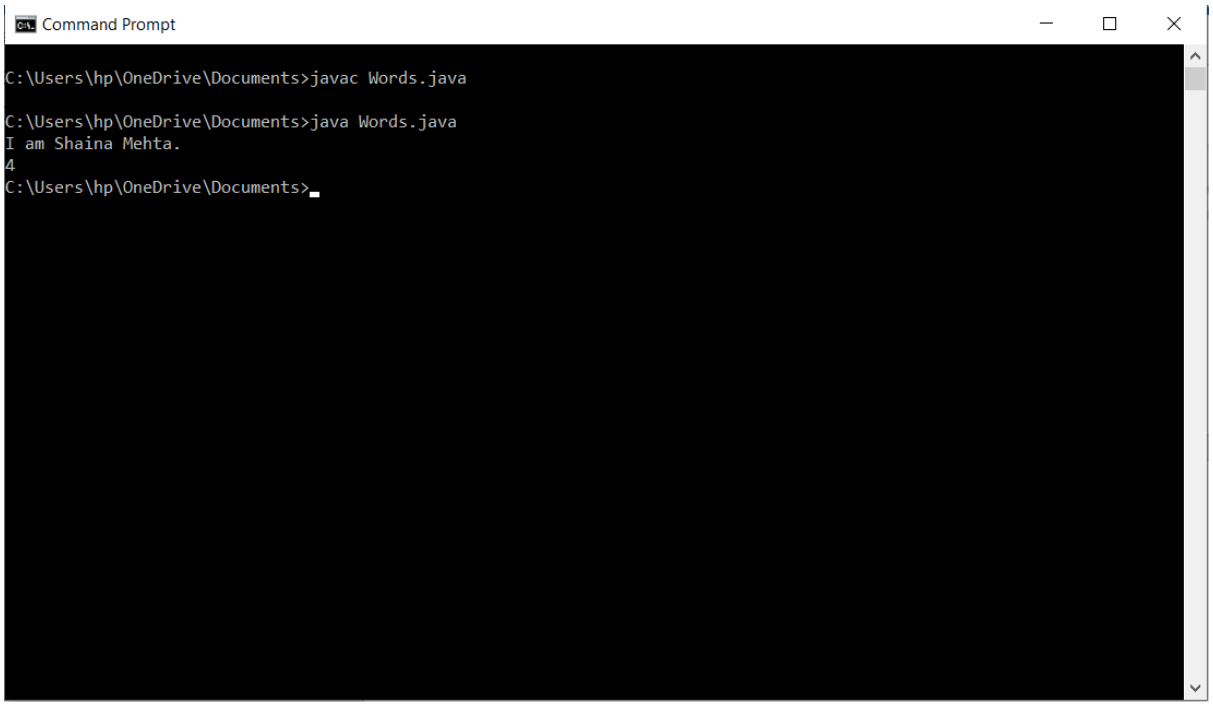
Q12) Write a Java program to get string and count number of words in a provided string.

Code:

```
// Words.java
import java.util.Scanner;
public class Words
{
public static void main(String[] args)
{
Scanner input = new Scanner (System.in);
String str = input.nextLine();
int ctr=0;
for(int i=0;i<str.length();i++)
{
char ch = str.charAt(i);
if(ch == ' ')
{
ctr++;
}
}
```

```
ctr++;
System.out.print(ctr);
input.close();
}
}
```

Output:



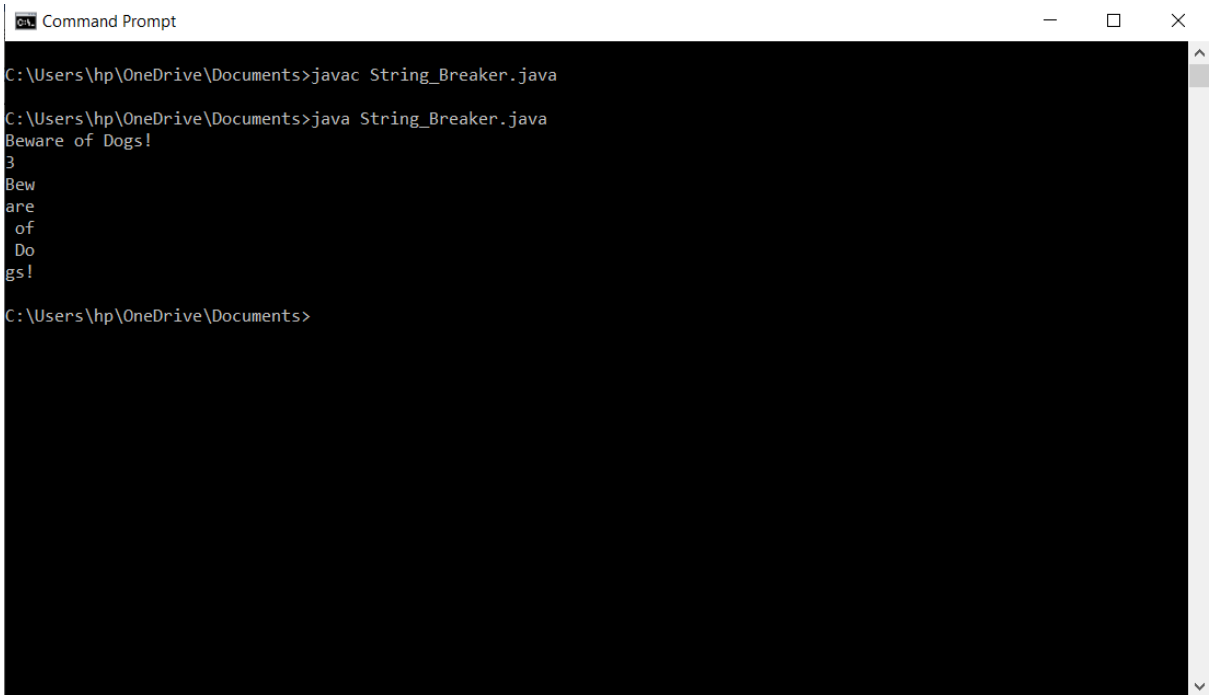
Q13) Write a Java program to divide a string in 'N' equal parts.

Code:

```
// String_Breaker.java
import java.util.Scanner;
public class String_Breaker
{
public static void main(String[] args)
{
Scanner input = new Scanner (System.in);
String str = input.nextLine();
int n = input.nextInt();
int pi = 0;
int ni = pi+n-1;
```

```
if(str.length() % n == 0)
{
while(ni < str.length())
{
String s1 = str.substring(pi,ni+1);
System.out.print(s1 + "\n");
pi = ni+1;
ni = pi+n-1;
}
}
else
{
System.out.print("Cannot Divide The String !");
}
input.close() ;
}
}
```

Output:



Q14) Design a class to represent a bank account. Include the following members:

Data members: Name of the depositor, account no, type of account and balance amount

Methods: to deposit an amount. To withdraw an amount after checking balance, to display the name and balance.

Use constructors to provide the initial values.

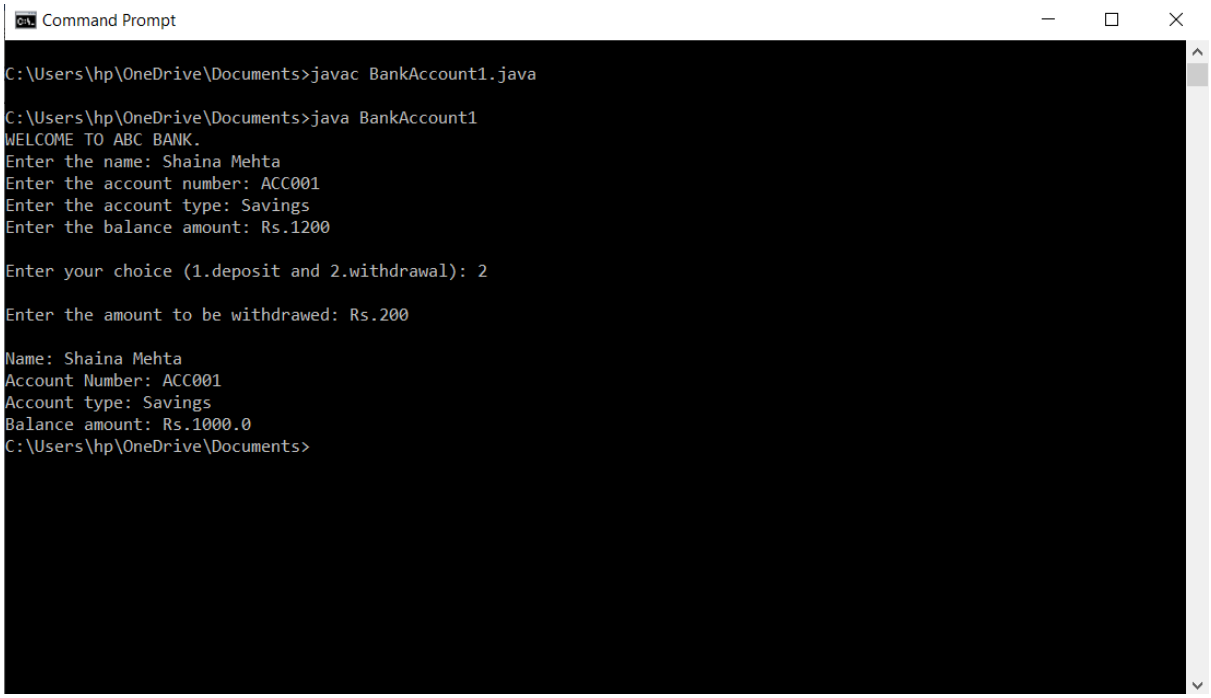
Code:

```
//BankAccount1.java
import java.util.Scanner;
class Accounts
{
Scanner input = new Scanner (System.in);
private String depoName;
private String accNo;
private String accType;
private float balAmount;
Accounts()
{
System.out.print("WELCOME TO ABC BANK.\n");
this.depoName = "No Name";
this.accNo = "ACC000";
this.accType = "Savings";
this.balAmount = 0.000F;
}
public void getAmt()
{
System.out.print("Enter the name: ");
depoName=input.nextLine();
System.out.print("Enter the account number: ");
accNo=input.nextLine();
System.out.print("Enter the account type: ");
accType=input.nextLine();
System.out.print("Enter the balance amount: Rs.");
```

```
balAmount=input.nextFloat();
}
public void compute()
{
float minbal=1000.000F;
float amtD,amtW;
int choice;
System.out.print("\nEnter your choice (1.deposit and 2.withdrawal): ");
choice=input.nextInt();
switch (choice)
{
case 1:System.out.print("\nEnter the amount to be deposited: Rs.");
amtD=input.nextFloat();
balAmount=balAmount+amtD;
break;
case 2:if(balAmount<minbal)
{
System.out.print("The amount cannot be withdrawn. So, we will cut Rs 10 from your account.");
balAmount=balAmount-10;
}
else
{
System.out.print("\nEnter the amount to be withdrawed: Rs.");
amtW=input.nextFloat();
balAmount=balAmount-amtW;
}
break;
}
}
public void showAmt()
```

```
{
compute();
System.out.print("\nName: " + depoName);
System.out.print("\nAccount Number: " + accNo);
System.out.print("\nAccount type: " + accType);
System.out.print("\nBalance amount: Rs." + balAmount);
}
}
public class BankAccount1 {
public static void main(String[] args) {
Accounts a = new Accounts();
a.getAmt();
a.showAmt();
}
}
```

Output:



Q15) Create a linked list of n nodes and then reverse the order of nodes.

Code:

```
// SLLUse.java
```

```
import java.util.Scanner;
class Node
{
public int data;
Node next;
public Node(int num)
{
this.data = num;
}
}
class SinglyLinkedList
{
private Node head;
public int size()
{
int len = 0;
Node temp = head;
if(temp == null)
{
return 0;
}
while (temp != null)
{
len++;
temp = temp.next;
}
return len;
}
public void insert(int data,int index)
{
int len = size();
```

```
if(index > len)
{
System.out.println("Wrong Index.");
return;
}
Node newNode = new Node(data);
if(head == null)
{
head = newNode;
return;
}
if(index == 0)
{
newNode.next = head;
head = newNode;
}
else
{
int ctr = 0;
Node temp = head;
while(temp != null && ctr < index-1)
{
temp = temp.next;
ctr ++;
}
if(temp != null)
{
Node a =  temp.next;
temp.next = newNode;
newNode.next = a;
}
```

```
}
}
public void delete(int index)
{
int len = size();
if(len == 0)
{
System.out.println("List Is Empty.");
return;
}
if(index >= len)
{
System.out.println("Wrong Index.");
return;
}
if(len == 1)
{
head = null;
return;
}
if(index == 0)
{
Node a = head;
head = head.next;
a = null;
}
else
{
int ctr = 0;
Node temp = head;
while(temp.next != null && ctr < index-1)
```

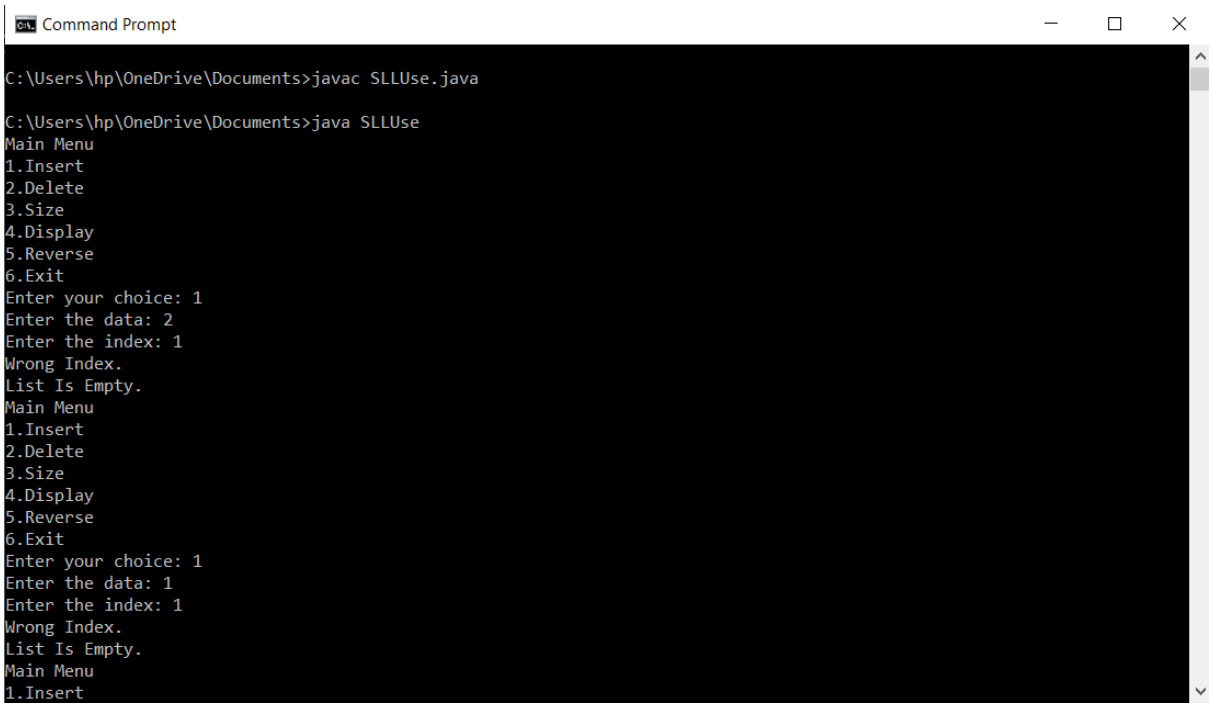
```
{
temp = temp.next;
ctr ++;
}
if(temp.next != null)
{
Node a =  temp.next;
temp.next = a.next;
a.next = null;
a = null;
}
}
}
public void display()
{
Node temp = head;
if(temp == null)
{
System.out.println("List Is Empty.");
return;
}
System.out.print("The list is: ");
while(temp != null)
{
System.out.print(temp.data + " ");
temp = temp.next;
}
System.out.print("\n");
}
public void reverse()
{
```

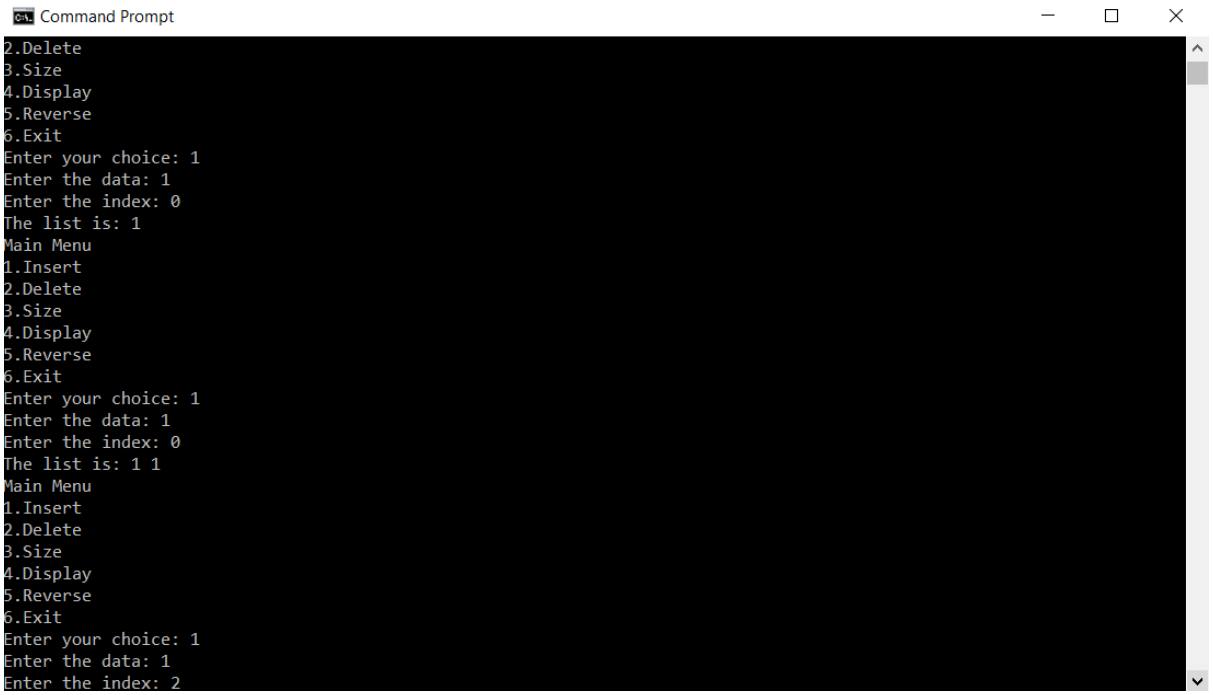
```
Node curr = head;
if(curr == null)
{
System.out.println("List Is Empty.");
return;
}
System.out.print("The reverse of the list is: ");
if(curr.next == null)
{
return;
}
Node prev = null;
Node n = curr.next;
while(curr != null)
{
curr.next = prev;
prev = curr;
curr = n;
if(curr != null)
{
n = curr.next;
}
}
head = prev;
}
}
public class SLLUse
{
public static void main(String[] args)
{
SinglyLinkedList l = new SinglyLinkedList();
```

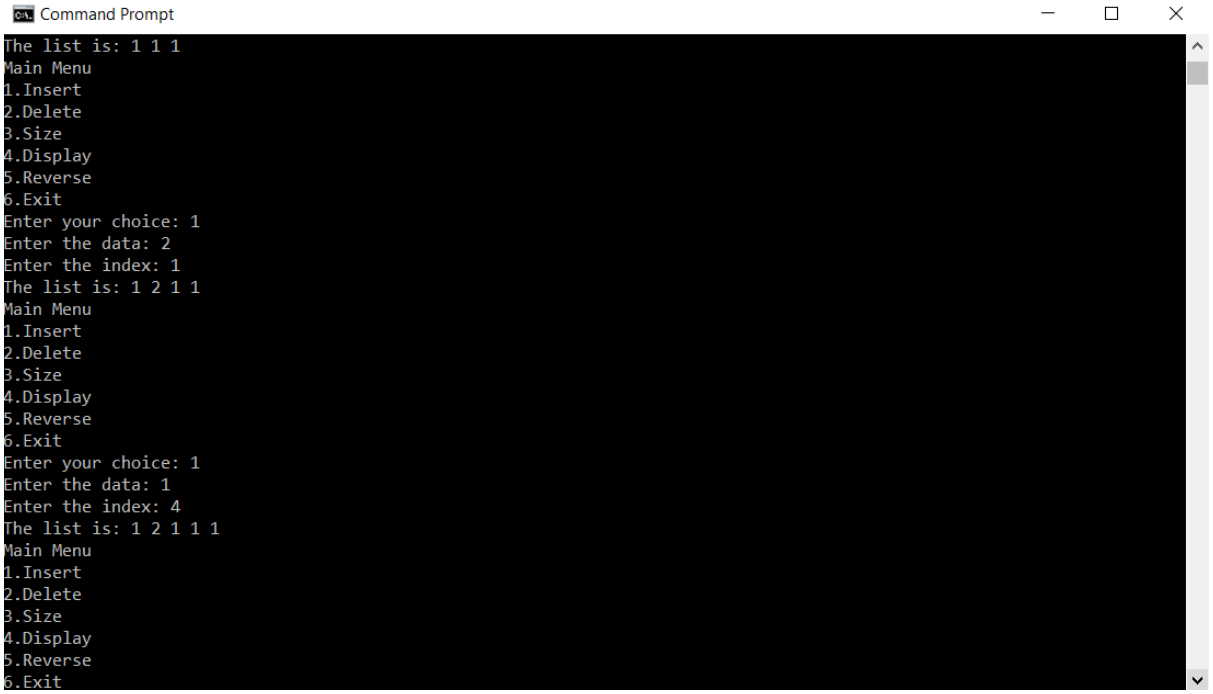
```
Scanner inp = new Scanner(System.in);
int ch;
int data;
int ind1,ind2;
int length;
while(true)
{
System.out.println("Main Menu");
System.out.println("1.Insert");
System.out.println("2.Delete");
System.out.println("3.Size");
System.out.println("4.Display");
System.out.println("5.Reverse");
System.out.println("6.Exit");
System.out.print("Enter your choice: ");
ch = inp.nextInt();
switch(ch)
{
case 1: System.out.print("Enter the data: ");
data = inp.nextInt();
System.out.print("Enter the index: ");
ind1 = inp.nextInt();
l.insert(data,ind1);
l.display();
break;
case 2: System.out.print("Enter the index: ");
ind2 = inp.nextInt();
l.delete(ind2);
l.display();
break;
case 3: length = l.size();
```

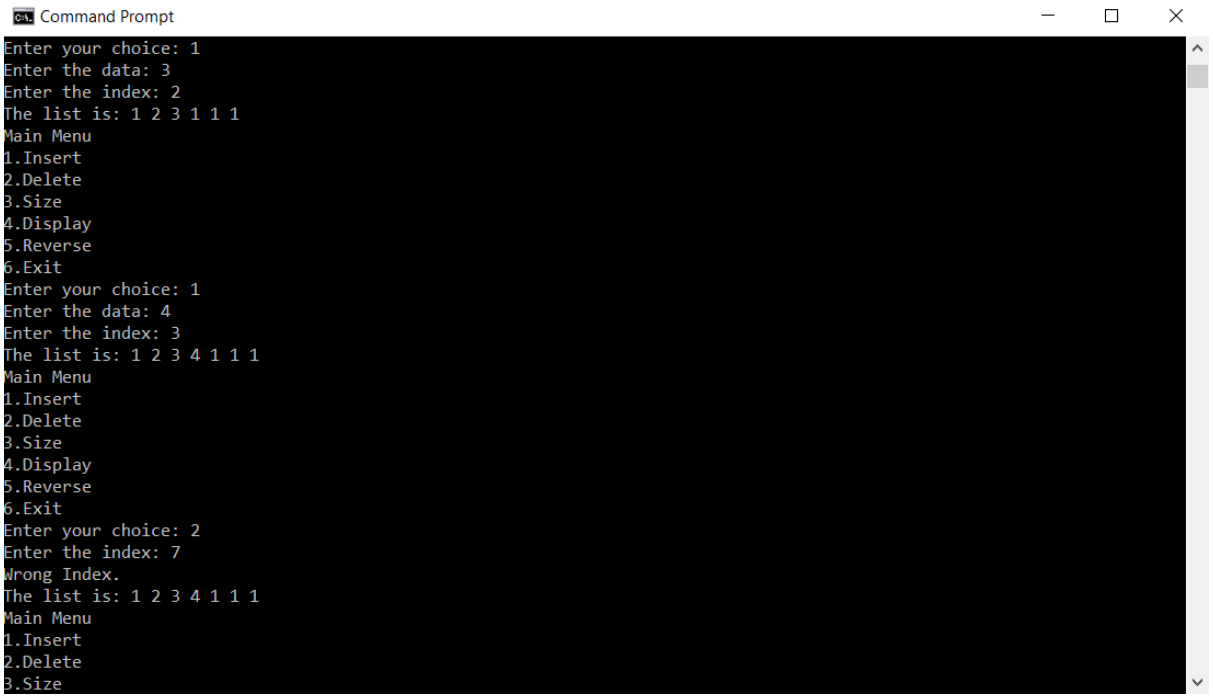
```
System.out.println("The size of the list is: " + length);
break;
case 4: l.display();
break;
case 5: l.reverse();
l.display();
break;
case 6: System.exit(0);
break;
}
}
}
}
```

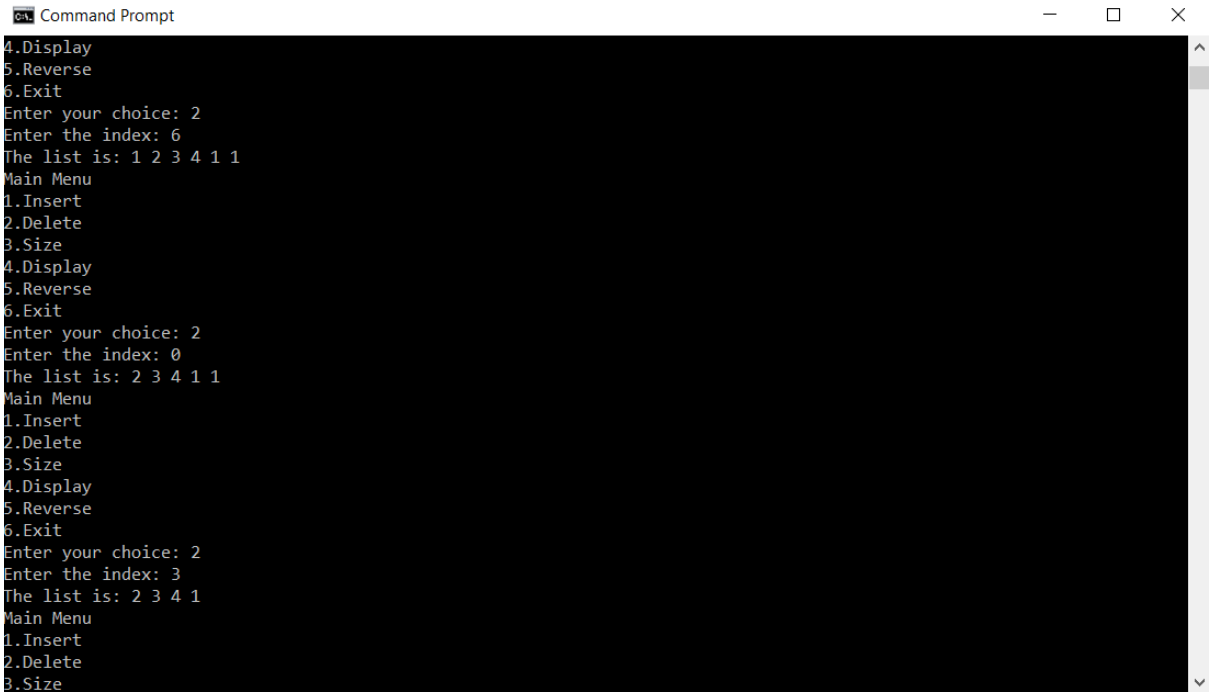
Output:

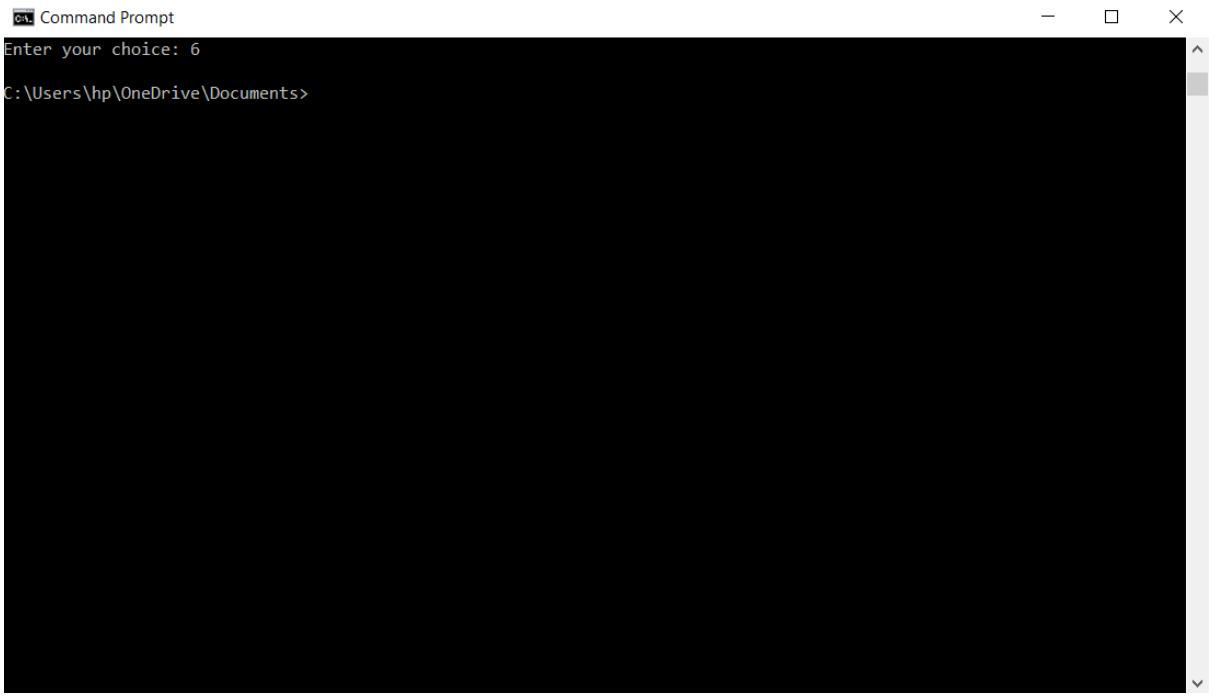



Command Prompt
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 1
Enter the index: 0
The list is: 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 1
Enter the index: 0
The list is: 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 1
Enter the index: 2


Command Prompt
The list is: 1 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 2
Enter the index: 1
The list is: 1 2 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 1
Enter the index: 4
The list is: 1 2 1 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit


Command Prompt
Enter your choice: 1
Enter the data: 3
Enter the index: 2
The list is: 1 2 3 1 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 1
Enter the data: 4
Enter the index: 3
The list is: 1 2 3 4 1 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 2
Enter the index: 7
Wrong Index.
The list is: 1 2 3 4 1 1 1
Main Menu
1.Insert
2.Delete
3.Size


Command Prompt
4.Display
5.Reverse
6.Exit
Enter your choice: 2
Enter the index: 6
The list is: 1 2 3 4 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 2
Enter the index: 0
The list is: 2 3 4 1 1
Main Menu
1.Insert
2.Delete
3.Size
4.Display
5.Reverse
6.Exit
Enter your choice: 2
Enter the index: 3
The list is: 2 3 4 1
Main Menu
1.Insert
2.Delete
3.Size



Q16) WAP to increment the employee salaries on the basis of their designation (Manager-5000, General Manager-10000, CEO-20000, worker-2000). Use employee name, id, designation, salary as data member and inc_sal as member function.

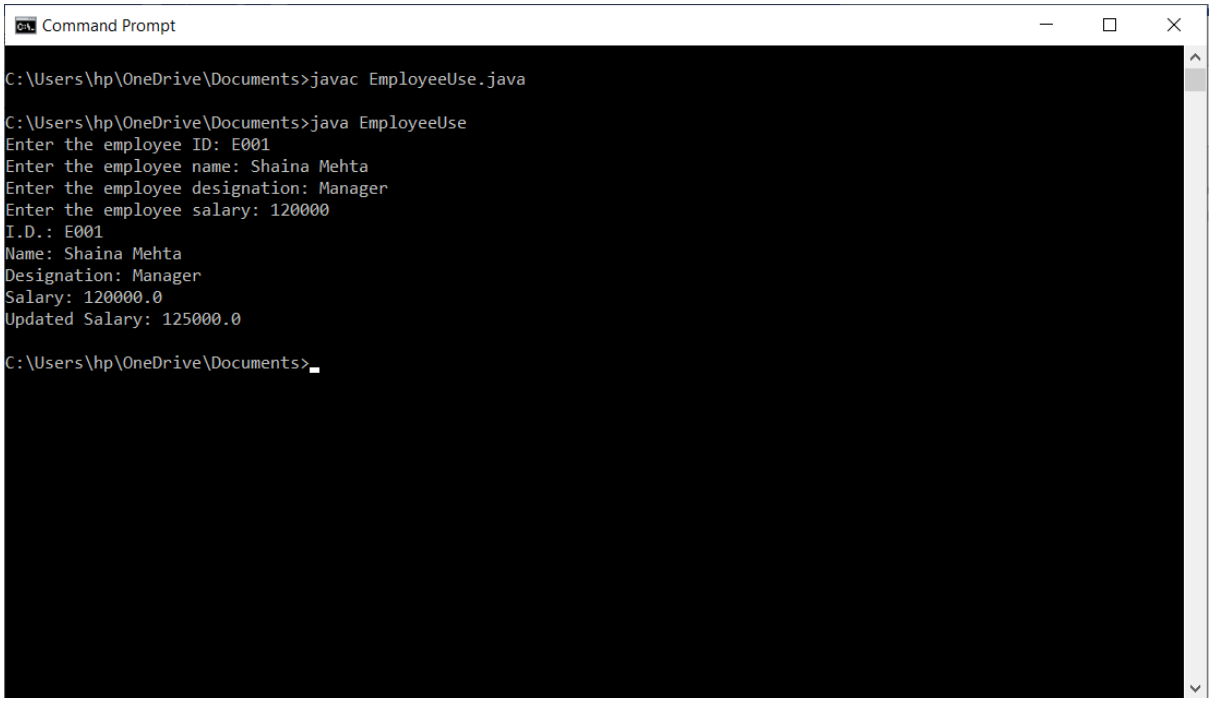
Code:

```
//EmployeeUse.java

import java.util.Scanner;

class Employee

{
```

```
Scanner input = new Scanner(System.in);
private String empName;
private String empID;
private String empD;
private float salary;
Employee()
{
this.empName = "No Name";
this.empID = "E000";
this.empD = "None";
this.salary = 0.000f;
}
public void getEmpDetails()
{
System.out.print("Enter the employee ID: ");
empID = input.nextLine();
System.out.print("Enter the employee name: ");
empName = input.nextLine();
System.out.print("Enter the employee designation: ");
empD = input.nextLine();
System.out.print("Enter the employee salary: ");
salary = input.nextFloat();
}
private void incSal()
{
if(empD.compareToIgnoreCase("Manager") == 0)
{
salary = salary + 5000;
}
else if(empD.compareToIgnoreCase("General Manager") == 0)
{
```

```
salary = salary + 10000;
}
else if(empD.compareToIgnoreCase("CEO") == 0)
{
salary = salary + 20000;
}
else
{
salary = salary + 2000;
}
}
public void showEmpDetails()
{
System.out.println("I.D.: " + empID);
System.out.println("Name: " + empName);
System.out.println("Designation: " + empD);
System.out.println("Salary: " + salary);
incSal();
System.out.println("Updated Salary: " + salary);
}
}
public class EmployeeUse
{
public static void main(String[] args)
{
Employee e = new Employee();
e.getEmpDetails();
e.showEmpDetails();
}
}
```

Output:



Q17) Assume that a bank maintains two kinds of account for its customers, one called as savings account and the other as current account. The saving account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Class account stores customer name, account number and the type of account.

Include member functions:-

- **Accept deposit from a customer and update the balance.**
- **Display the balance.**
- **Compute and deposit interest.**
- **Permit withdrawal and update balance.**
- **Check for minimum balance, impose penalty and update the balance.**

Code:

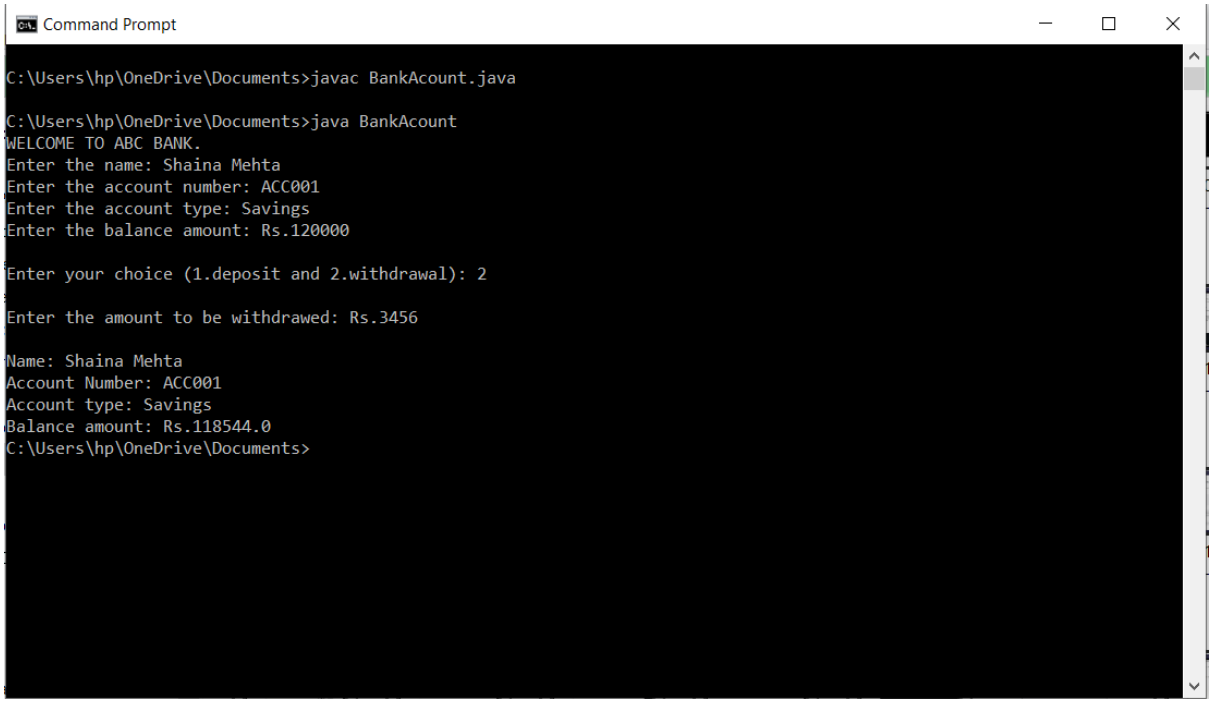
```
//BankAcount.java

import java.util.Scanner;

class Account

{

Scanner input = new Scanner (System.in);

private String depoName;

private String accNo;

private String accType;

private float balAmount;
```

```
Account()
{
System.out.print("WELCOME TO ABC BANK.\n");
this.depoName = "No Name";
this.accNo = "ACC000";
this.accType = "Savings";
this.balAmount = 0.000F;
}
public void getAmt()
{
System.out.print("Enter the name: ");
depoName=input.nextLine();
System.out.print("Enter the account number: ");
accNo=input.nextLine();
System.out.print("Enter the account type: ");
accType=input.nextLine();
System.out.print("Enter the balance amount: Rs.");
balAmount=input.nextFloat();
}
public void compute()
{
float minbal=1000.000F;
float amtDS,amtWS,amtDC,amtWC;
float ci=2000.000F;
int choiceS,choiceC;
if(accType.compareToIgnoreCase("Savings")==0)
{
balAmount=balAmount+ci;
System.out.print("\nEnter your choice (1.deposit and 2.withdrawal): ");
choiceS=input.nextInt();
switch (choiceS)
```

```
{
case 1:System.out.print("\nEnter the amount to be deposited: Rs.");
amtDS=input.nextFloat();
balAmount=balAmount+amtDS;
break;
case 2:System.out.print("\nEnter the amount to be withdrawed: Rs.");
amtWS=input.nextFloat();
balAmount=balAmount-amtWS;
break;
}
}
else
{
System.out.print("\nEnter your choice (1.deposit and 2.withdrawal): ");
choiceC=input.nextInt();
switch (choiceC)
{
case 1:System.out.print("\nEnter the amount to be deposited: Rs.");
amtDC=input.nextFloat();
balAmount=balAmount+amtDC;
break;
case 2:if(balAmount<minbal)
{
System.out.print("The amount cannot be withdrawn. So, we will cut Rs 10 from your account.");
balAmount=balAmount-10;
}
else
{
System.out.print("\nEnter the amount to be withdrawed: Rs.");
amtWC=input.nextFloat();
```

```
balAmount=balAmount-amtWC;
}
break;
}
}
}
public void showAmt()
{
compute();
System.out.print("\nName: " + depoName);
System.out.print("\nAccount Number: " + accNo);
System.out.print("\nAccount type: " + accType);
System.out.print("\nBalance amount: Rs." + balAmount);
}
}
public class BankAcount
{
public static void main(String[] args)
{
Account a = new Account();
a.getAmt();
a.showAmt();
}

}
```

Output:



Q18) Design three classes: Student, Exam and Result. The student class has data members such as roll no, name etc. Create a class Exam by inheriting the Student class. The Exam class adds data members representing the marks scored in six subjects. Derive the Result from class Exam and it has its own members such as total marks and average. Calculate the total marks and average.

Code:

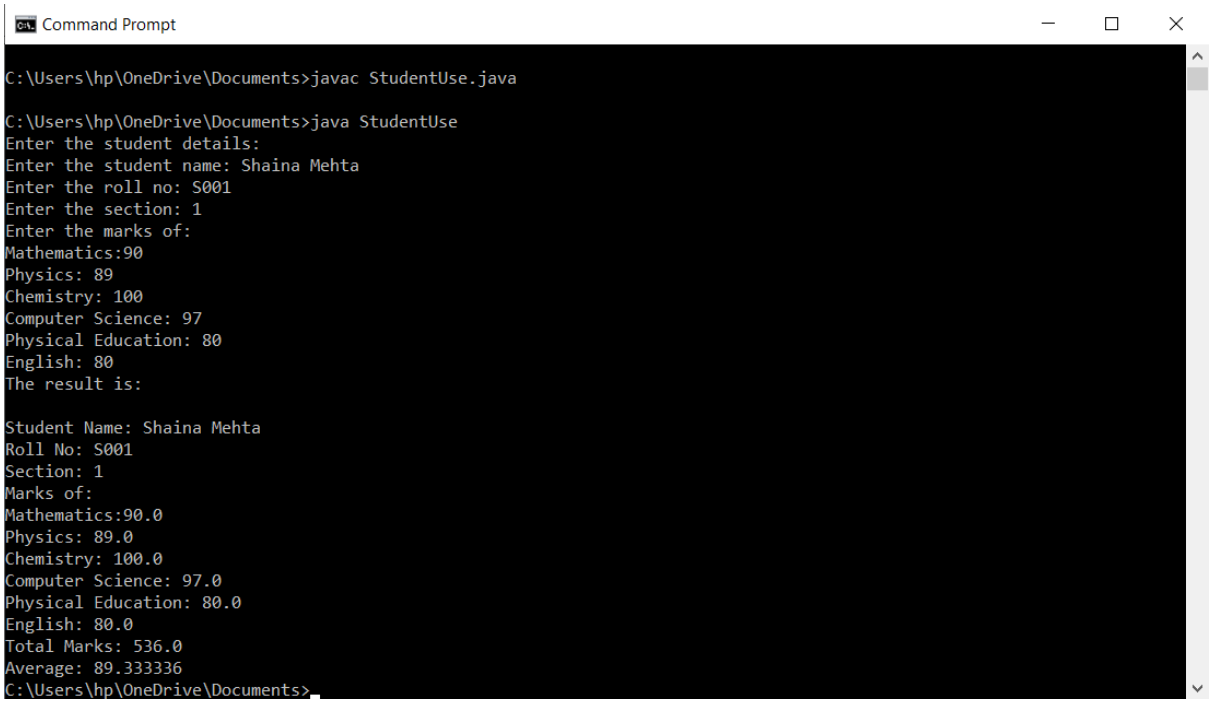
```
// StudentUse.java

import java.util.Scanner;

class Student

{

public String sName;

public String sRollNo;

public char sSec;

public Student()

{

this.sRollNo = "S000";

this.sName = "No Name";

this.sSec = 0;

}

}
```

```
class Exam extends Student
{
public float maths;
public float phy;
public float chem;
public float cs;
public float phe;
public float english;
public float total;
public float avg;
public Exam()
{
super();
this.maths = 0.000f;
this.phy = 0.00f;
this.chem = 0.000f;
this.cs = 0.000f;
this.phe = 0.000f;
this.english = 0.000f;
this.total = 0.000f;
this.avg = 0.000f;
}
public void calTotalAndAverage()
{
total = maths + phy + chem + cs + phe + english;
avg = total / 6;
}
}
class Result extends Exam
{
Scanner input = new Scanner(System.in);
```

```
public Result()
{
super();
}
public void getStuDetails()
{
System.out.print("Enter the student details: \n");
System.out.print("Enter the student name: ");
super.sName=input.nextLine();
System.out.print("Enter the roll no: ");
super.sRollNo=input.nextLine();
System.out.print("Enter the section: ");
super.sSec=input.next().charAt(0);
System.out.print("Enter the marks of: \nMathematics:");
super.maths=input.nextInt();
System.out.print("Physics: ");
super.phy=input.nextInt();
System.out.print("Chemistry: ");
super.chem=input.nextInt();
System.out.print("Computer Science: ");
super.cs=input.nextInt();
System.out.print("Physical Education: ");
super.phe=input.nextInt();
System.out.print("English: ");
super.english=input.nextInt();
}
public void showResult()
{
System.out.print("The result is: \n");
System.out.print("\nStudent Name: " + super.sName);
System.out.print("\nRoll No: " + super.sRollNo);
```

```
System.out.print("\nSection: " + super.sSec);
System.out.print("\nMarks of:" + "\n" + "Mathematics:" + super.maths);
System.out.print("\nPhysics: " + super.phy);
System.out.print("\nChemistry: " + super.chem);
System.out.print("\nComputer Science: " + super.cs);
System.out.print("\nPhysical Education: " + super.phe);
System.out.print("\nEnglish: " + super.english);
super.calTotalAndAverage();
System.out.print("\nTotal Marks: " + super.total);
System.out.print("\nAverage: " + super.avg);
}
}
public class StudentUse
{
public static void main(String[] args)
{
Result a = new Result();
a.getStuDetails();
a.showResult();
}
}
```

Output:



Q19) Write a program with given interfaces MotorBike and Cycle, then implement in child class TwoWheeler and display distance & speed.

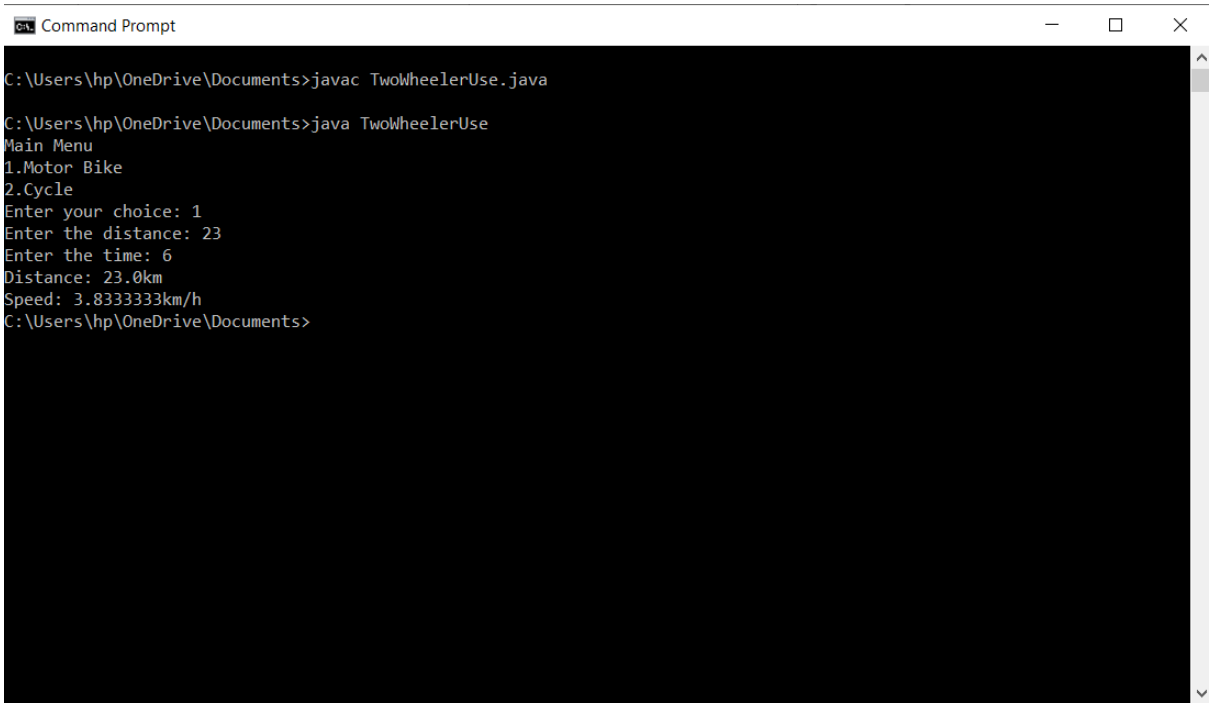
Code:

```
//TwoWheelerUse.java
import java.util.Scanner;
interface MotorBike
{
public void inputDistanceTime();
public void outputDistanceSpeed();
}
interface Cycle
{
public void getDistanceTime();
public void showDistanceTime();
}
class TwoWheeler implements MotorBike,Cycle
{
Scanner input=new Scanner(System.in);
private float distance;
```

```
private float Times;
private float speed;
TwoWheeler()
{
this.distance=0.000F;
this.Times=0.000F;
this.speed=0.000F;
}
public void inputDistanceTime()
{
System.out.print("Enter the distance: ");
distance=input.nextFloat();
System.out.print("Enter the time: ");
Times=input.nextFloat();
}
public void SpeedCalc()
{
speed=distance/Times;
}
public void outputDistanceSpeed()
{
System.out.print("Distance: " + distance + "km");
SpeedCalc();
System.out.print("\nSpeed: " + speed + "km/h");
}
public void getDistanceTime()
{
System.out.print("Enter the distance: ");
distance=input.nextFloat();
System.out.print("Enter the time: ");
Times=input.nextFloat();
```

```
}
public void showDistanceTime()
{
System.out.print("Distance: " + distance + "km");
SpeedCalc();
System.out.print("\nSpeed: " + speed + "km/h");
}
}
public class TwoWheelerUse
{
public static void main(String[] args)
{
Scanner sc=new Scanner(System.in);
TwoWheeler t=new TwoWheeler();
int choice;
System.out.print("Main Menu \n1.Motor Bike \n2.Cycle \nEnter your choice: ");
choice=sc.nextInt();
switch (choice)
{
case 1: t.inputDistanceTime();
t.outputDistanceSpeed();
break;
case 2: t.getDistanceTime();
t.showDistanceTime();
break;
}
sc.close() ;
}
}
```

Output:



Q20) An interface called RegularPolygon with two abstract methods: getNumSides and getSideLength. A class EquilateralTriangle that implements the interface, has getNumSides return 3 and getSideLength return an instance variable that is set by the constructor. A class Square that implements the interface, has getNumSides return 4 and getSideLength return an instance variable that is set by the constructor. Add a static totalSides method, that given a RegularPolygon[], returns the sum of the number of sides of all the elements. Add method: getPerimeter (n * length, where n is the number of sides).

Code :

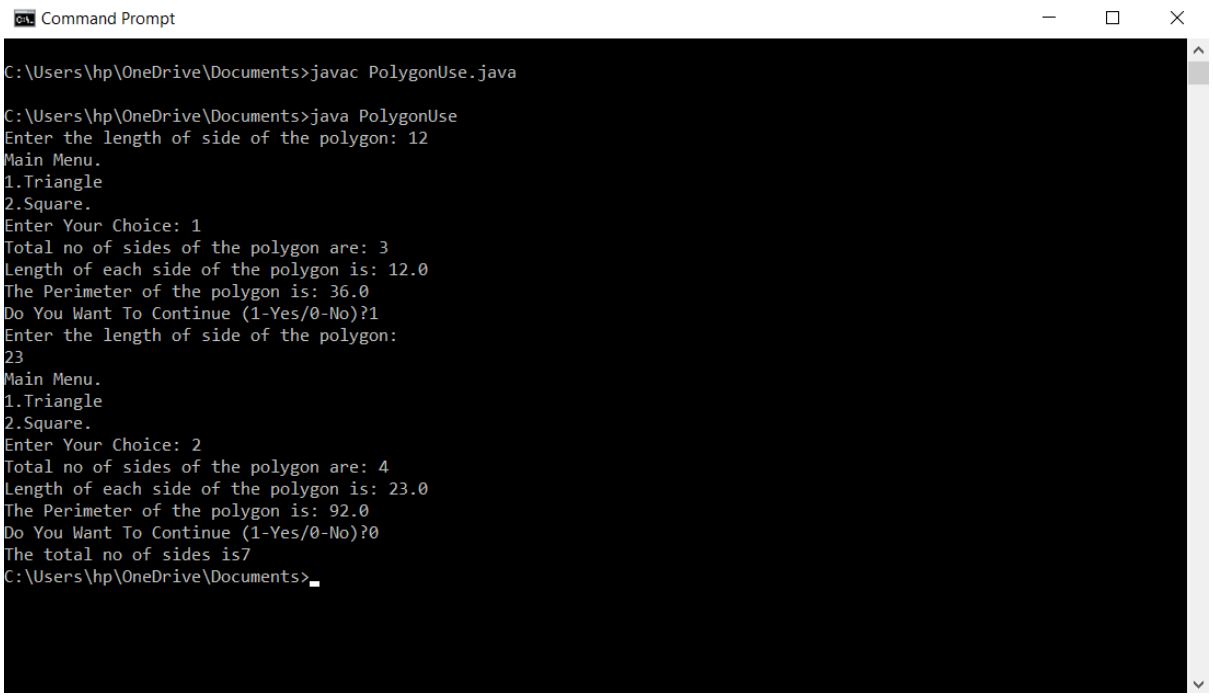
```
//PolygonUse.java

import java.util.Scanner;

interface RegularPolygon{

int getNumSide();

float getSideLength();

}

class EquilateralTriangle implements RegularPolygon{

float length;

EquilateralTriangle(float len){

this.length=len;

}

public int getNumSide(){
```

```
return 3;
}
public float getSideLength() {
return length;
}
}
class Square implements RegularPolygon{
float length;
Square(float len){
this.length=len;
}
public int getNumSide(){
return 4;
}
public float getSideLength() {
return length;
}
}
public class PolygonUse {
static int total=0;
public static void totalSides(int c) {
if(c==1) {
total+=3;
}
else {
total+=4;
}
}
public static float getPerimeter(int sides, float length){
return sides*length;
}
```

```
public static void Display(RegularPolygon rp){
System.out.println("Total no of sides of the polygon are: "+ rp.getNumSide());
System.out.println("Length of each side of the polygon is: "+rp.getSideLength());
System.out.println("The Perimeter of the polygon is: "+ getPerimeter(rp.getNumSide(),rp.getSideLength()));
}
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
int ch;
int cont;
float length;
do {
System.out.print("Enter the length of side of the polygon: ");
length=sc.nextFloat();
System.out.println("Main Menu.");
System.out.println("1.Triangle");
System.out.println("2.Square.");
System.out.print("Enter Your Choice: ");
ch=sc.nextInt();
if(ch==1) {
EquilateralTriangle et = new EquilateralTriangle(length);
totalSides(ch);
Display(et);
}
else if(ch==2) {
Square sq = new Square(length);
totalSides(ch);
Display(sq);
}
else {
System.out.println("Invalid Choice.");
```

```
}
System.out.print("Do You Want To Continue (1-Yes/0-No)?");
cont=sc.nextInt();
}while(cont!=0);
System.out.print("The total no of sides is" + total);
sc.close();
}
}
```

Output :



Q21) Create a class Employee. Derive 3 classes from this class namely, Programmer, Analyst & Project Leader. Take attributes and operations on your own.

Code:

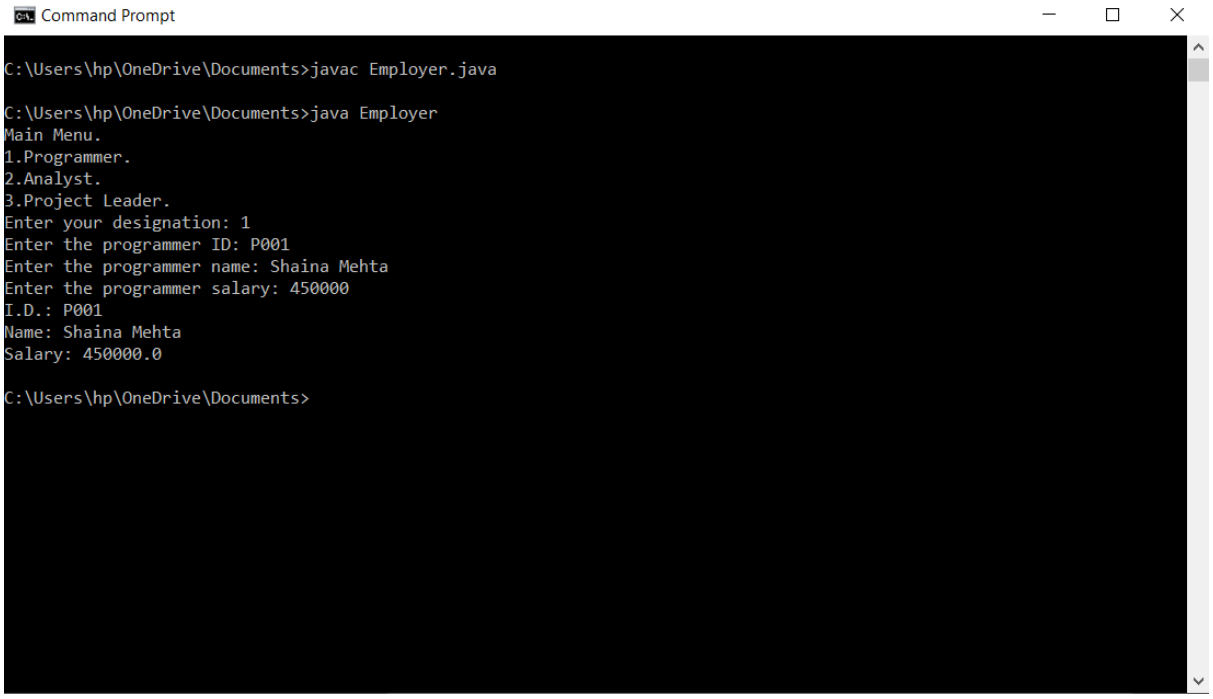
```
//Employer.java
import java.util.Scanner;
abstract class Employees
{
public abstract void getData();
public abstract void showData();
}
```

```
class Programmer extends Employees
{
Scanner sc1=new Scanner(System.in);
private String pName;
private String pID;
private float pSalary;
public void getData()
{
System.out.print("Enter the programmer ID: ");
pID = sc1.nextLine();
System.out.print("Enter the programmer name: ");
pName = sc1.nextLine();
System.out.print("Enter the programmer salary: ");
pSalary = sc1.nextFloat();
}
public void showData()
{
System.out.println("I.D.: " + pID);
System.out.println("Name: " + pName);
System.out.println("Salary: " + pSalary);
}
}
class Analyst extends Employees
{
Scanner sc2=new Scanner(System.in);
private String aName;
private String aID;
private float aSalary;
public void getData()
{
System.out.print("Enter the analyst ID: ");
```

```
aID = sc2.nextLine();
System.out.print("Enter the analyst name: ");
aName = sc2.nextLine();
System.out.print("Enter the analyst salary: ");
aSalary = sc2.nextFloat();
}
public void showData()
{
System.out.println("I.D.: " + aID);
System.out.println("Name: " + aName);
System.out.println("Salary: " + aSalary);
}
}
class Project_Leader extends Employees
{
Scanner sc3=new Scanner(System.in);
private String plName;
private String plID;
private float plSalary;
public void getData()
{
System.out.print("Enter the project leader ID: ");
plID = sc3.nextLine();
System.out.print("Enter the project leader name: ");
plName = sc3.nextLine();
System.out.print("Enter the project leader salary: ");
plSalary = sc3.nextFloat();
}
public void showData()
{
System.out.println("I.D.: " + plID);
```

```
System.out.println("Name: " + plName);
System.out.println("Salary: " + plSalary);
}
}
public class Employer
{
public static void main(String[] args)
{
Programmer p = new Programmer();
Scanner sc = new Scanner(System.in);
Analyst a = new Analyst();
Project_Leader pl = new Project_Leader();
int choice;
System.out.print("Main Menu. \n1.Programmer. \n2.Analyst. \n3.Project Leader. \nEnter your designation: ");
choice = sc.nextInt();
switch (choice)
{
case 1: p.getData();
p.showData();
break;
case 2: a.getData();
a.showData();
break;
case 3: pl.getData();
pl.showData();
break;
}
sc.close();
}
}
```

Output:



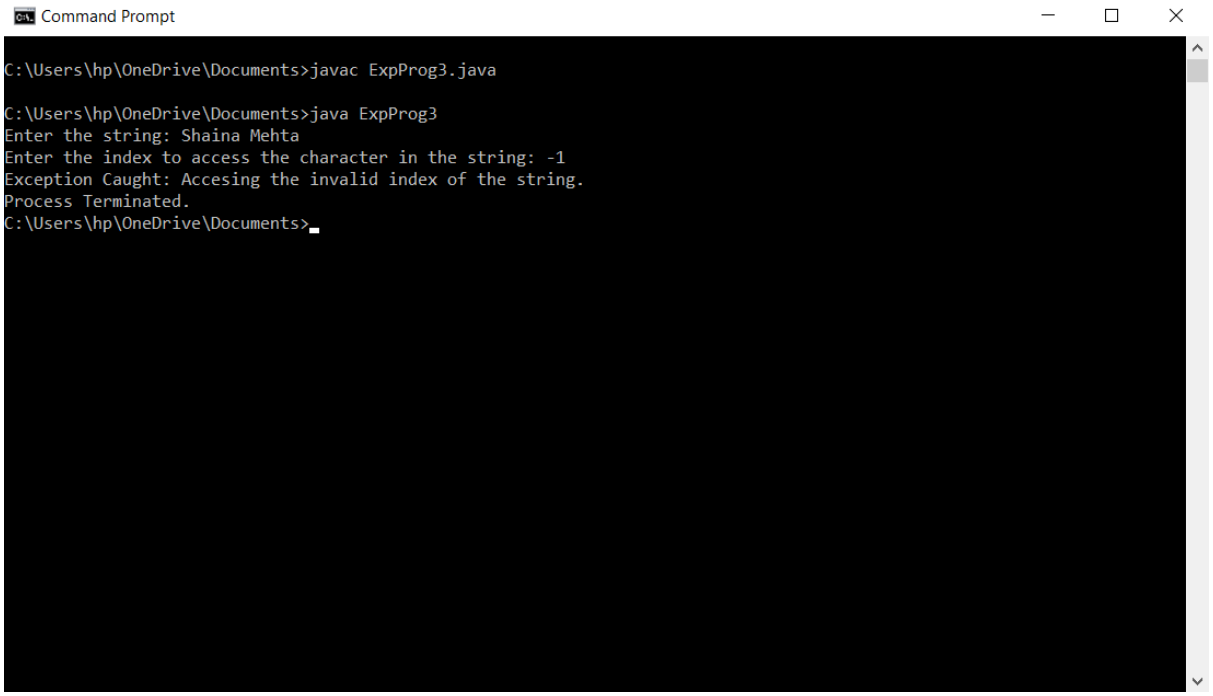
Q22) Create a class with a main( ) that throws an object of class Exception inside a try block. Give the constructor for Exception a String argument. Catch the exception inside a catch clause and print the String argument. Add a finally clause and print a message to prove you were there.

Code:

```
//ExpProg3.java

import java.util.Scanner;

class Exceptions

{

Scanner sc = new Scanner(System.in);

public Exceptions(String s)

{

try {

System.out.print("Enter the index to access the character in the string: ");

int c= sc.nextInt();

char ch = s.charAt(c);

System.out.print("The character present in that index of the string is: " + ch);

}

catch(StringIndexOutOfBoundsException e)
```

```
{
System.out.print("Exception Caught: Accesing the invalid index of the string.");
}
finally {
System.out.print("\nProcess Terminated.");
}
}
}
public class ExpProg3
{
public static void main(String[] args)
{
Scanner sc1 = new Scanner(System.in);
System.out.print("Enter the string: ");
String str = sc1.nextLine();
Exceptions e = new Exceptions(str);
sc1.close();
}
}
```

Output:



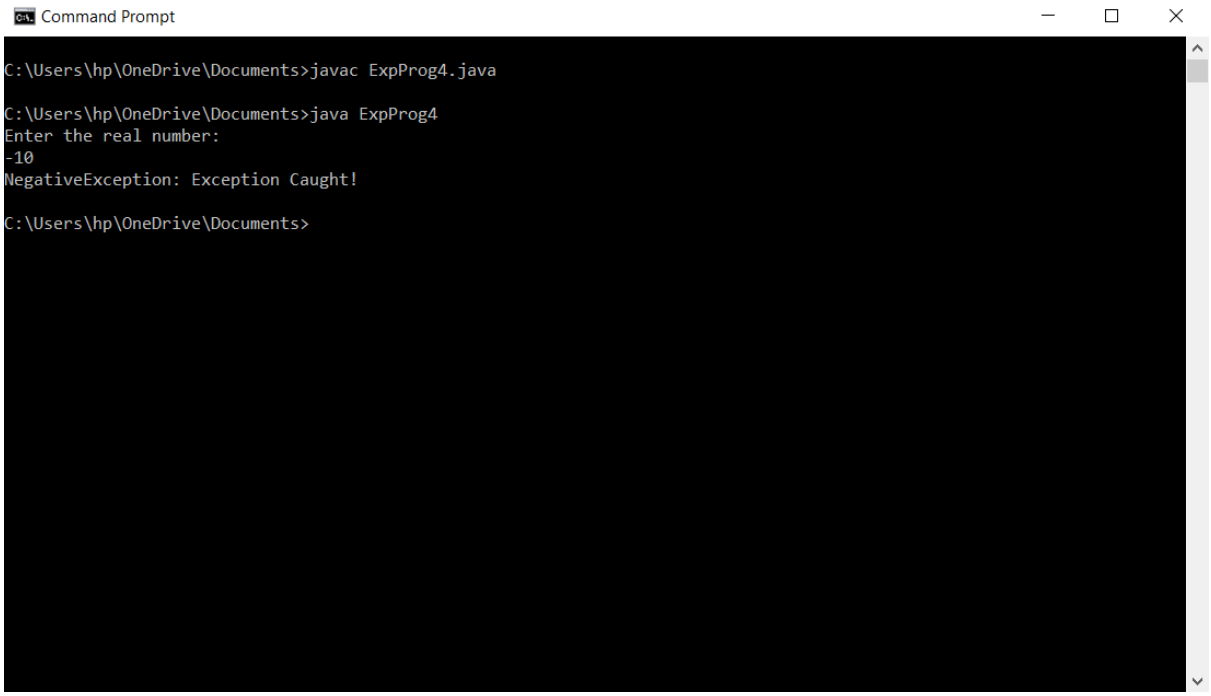
Q23) Create a program to ask the user for a real number and display its square root. Errors must be trapped using "try..catch"

Code:

```
//ExpProg4.java
import java.util.Scanner;
class NegativeException extends Exception{
NegativeException(String str){
super(str);
}
}
public class ExpProg4 {
public static void squareRoot() throws NegativeException{
Scanner sc = new Scanner(System.in);
System.out.println("Enter the real number:");
double a=sc.nextDouble();
if(a<0) {
throw new NegativeException("Exception Caught!");
}
else {
```

```
double b=Math.sqrt(a);
System.out.println("The square root of a number is: " + b);
}
sc.close();
}
public static void main (String [] args) {
try {
squareRoot();
}
catch(NegativeException e){
System.out.println(e);
}
}
}
```

Output:



Q24) Create a try block that is likely to generate three types of exception and then incorporate necessary catch blocks to catch and handle them appropriately.

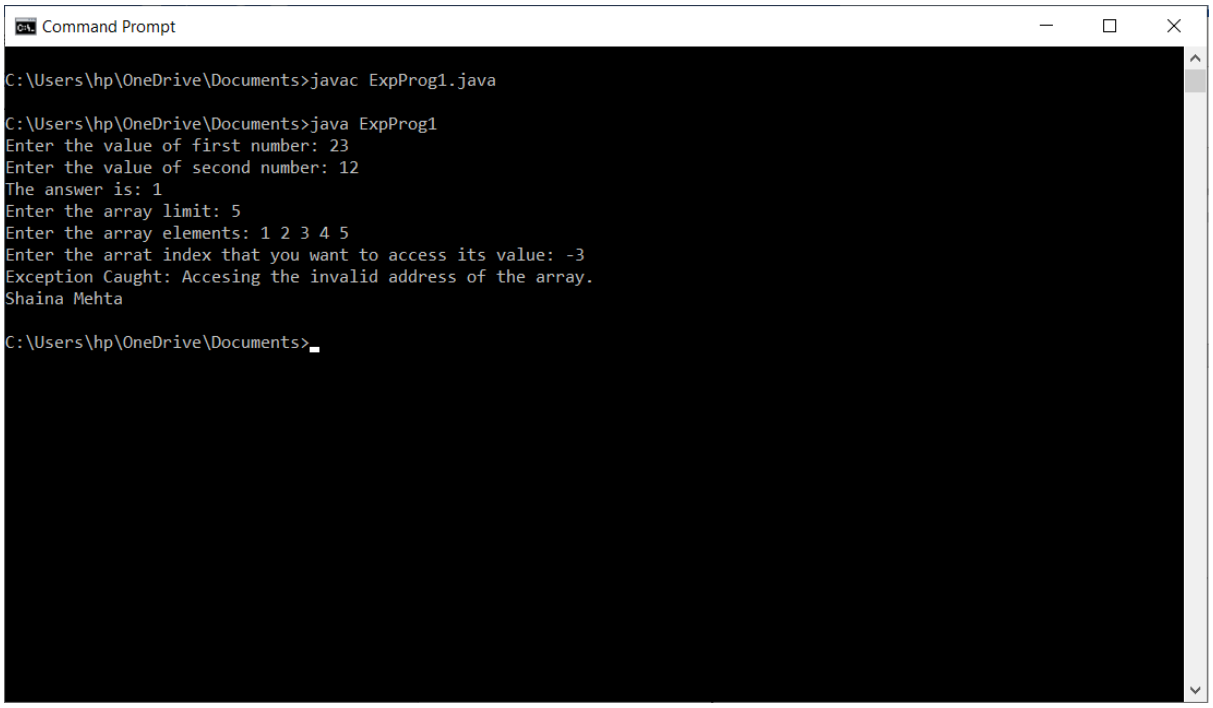
Code:

```
//ExpProg1.java
```

```
import java.util.Scanner;
public class ExpProg1
{
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
try {
System.out.print("Enter the value of first number: ");
int a=sc.nextInt();
System.out.print("Enter the value of second number: ");
int b=sc.nextInt();
int c=a/b;
System.out.print("The answer is: " + c);
System.out.print("\nEnter the array limit: ");
int n = sc.nextInt();
System.out.print("Enter the array elements: ");
int arr[] = new int [n];
for(int i=0;i<n;i++)
{
arr[i]=sc.nextInt();
}
System.out.print("Enter the arrat index that you want to access its value: ");
int ch=sc.nextInt();
System.out.print("The element present in that array index is: " + arr[ch]);
}
catch(ArithmeticException e)
{
System.out.print("Exception Caught: Divided by Zero.");
}
catch(NegativeArraySizeException e1)
{
```

```
System.out.print("Exception Caught: Invalid size of the array.");
}
catch(ArrayIndexOutOfBoundsException  e2)
{
System.out.print("Exception Caught: Accesing the invalid address of the array.");
}
System.out.println("\nShaina Mehta");
sc.close();
}
}
```

Output:



Q25) Create a class MyClass and create three methods myMethod1(), Method2() and Method3(). Invoke Method2() from Method1() and Method3() from Method2(). Write a code that can throw an exception inside myMethod3()

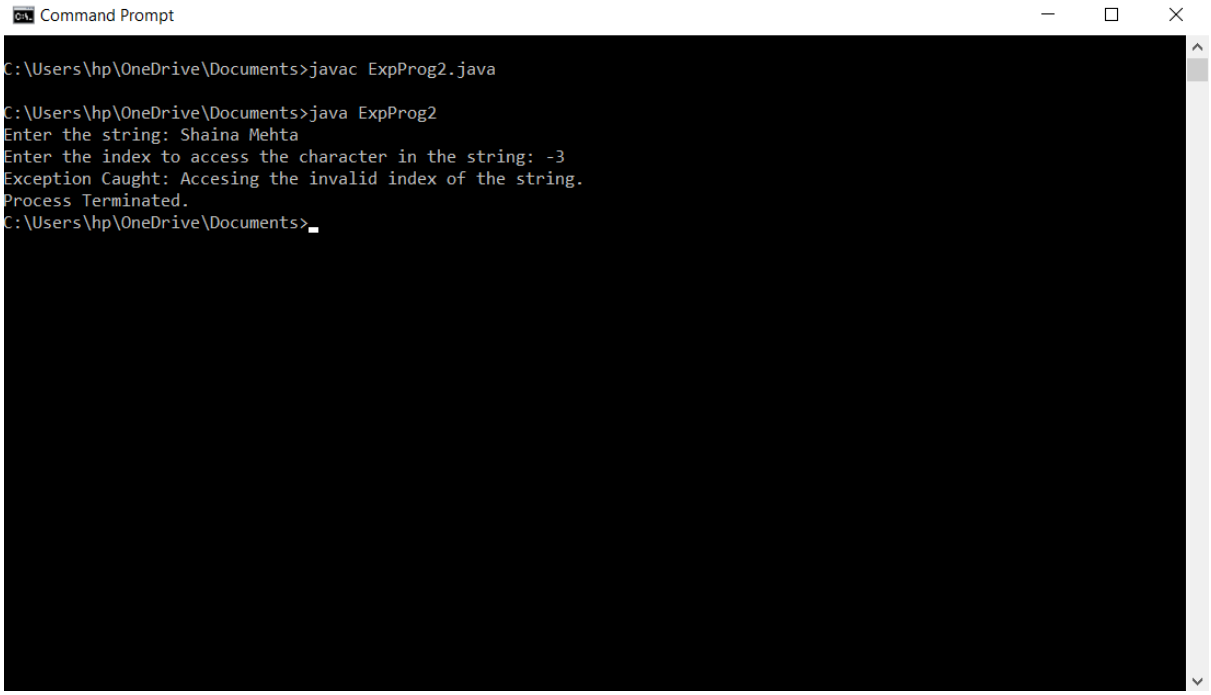
Code:

```
//ExpProg2.java
import java.util.Scanner;
class MyClass
{
```

```
Scanner sc=new Scanner(System.in);
private void MyMethod1()
{
System.out.print("Enter the string: ");
String str = sc.nextLine();
System.out.print("Enter the index to access the character in the string: ");
int c= sc.nextInt();
char ch = str.charAt(c);
System.out.print("The character present in that index of the string is: " + ch);
}
private void MyMethod2()
{
MyMethod1();
}
public void MyMethod3()
{
try {
MyMethod2();
}
catch(StringIndexOutOfBoundsException e)
{
System.out.print("Exception Caught: Accesing the invalid index of the string.");
}
finally {
System.out.print("\nProcess Terminated.");
}
}
}
public class ExpProg2
{
public static void main(String[] args)
```

```
{
MyClass m= new MyClass();
m.MyMethod3();
}
}
```

Output:



Q26) Write a program to deposit cash, withdraw in a bank using multithreading.

Code:

```
//BankAccountUse.java
import java.util.Scanner;
class accounts{
Scanner input = new Scanner(System.in);
String ID;
String name;
float balAmt;
boolean value;
accounts(){
this.ID="ACC000";
```

```
this.name="No Name";
this.balAmt=0.000F;
this.value=false;
}
synchronized public void getAmt(){
float a;
float b;
int choice;
//System.out.print("\n");
while(!value) {
try {
wait(10);
}
catch(InterruptedException e)
{
System.out.println(e);
}
System.out.print("Enter the account number: ");
ID=input.next();
System.out.print("Enter the name: ");
name=input.next();
System.out.print("Enter the balance amount: Rs.");
balAmt=input.nextFloat();
System.out.print("Do ypu want to deposit(0) or withdraw(1) the amount.");
choice = input.nextInt();
switch(choice)
{
case 0: System.out.print("Enter the amount to be deposited.");
a = input.nextInt();
balAmt+=a;
break;
```

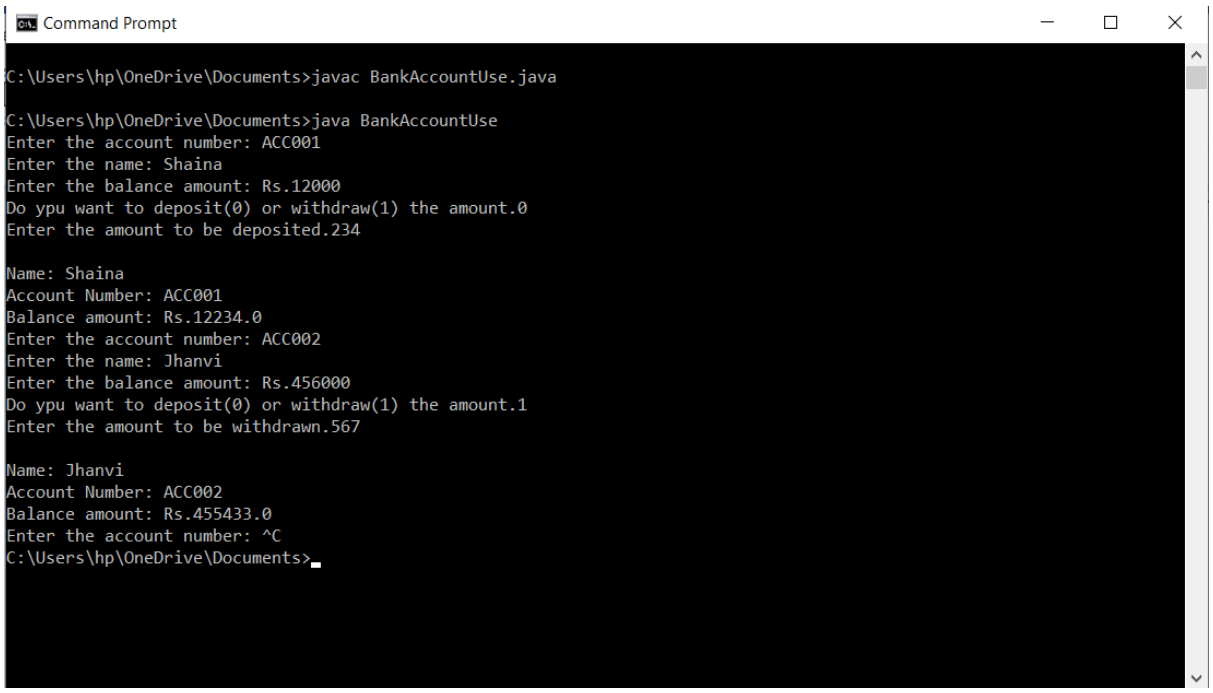
```
case 1:System.out.print("Enter the amount to be withdrawn.");
b = input.nextInt();
balAmt-=b;
break;
}
value = true;
notify();
}
}
synchronized public void showAmt(){
while(value) {
try {
wait(10);
}
catch(InterruptedException e)
{
System.out.println(e);
}
System.out.print("\nName: " + name);
System.out.print("\nAccount Number: " + ID);
System.out.print("\nBalance amount: Rs." + balAmt);
System.out.print("\n");
value=false;
notify();
}
}
}
class Insert implements Runnable{
accounts a;
Thread t;
Insert(accounts a){
```

```
this.a=a;
t = new Thread(this,"Insert");
}
public void run() {
while(true)
{
a.getAmt();
}
}
}
class Display implements Runnable{
accounts a;
Thread t;
Display(accounts a){
this.a=a;
t = new Thread(this,"Display");
}
public void run() {
while(true)
{
a.showAmt();
}
}
}
public class BankAccountUse {
public static void main(String[] args) {
accounts a = new accounts();
Insert i = new Insert(a);
Display d = new Display(a);
i.t.start();
d.t.start();
```

}

}

Output:



Q27) Implement three classes: Storage, Counter, and Printer. The Storage class should store an integer. The Counter class should create a thread that starts counting from 0 (0, 1, 2, 3 ...) and stores each value in the Storage class. The Printer class should create a thread that keeps reading the value in the Storage class and printing it. Write a program that creates an instance of the Storage class and sets up a Counter and a Printer object to operate on it.

Code:

//CSP.java

class Storage

{

private int mem;

public void setStorage(int data)

{

this.mem=data;

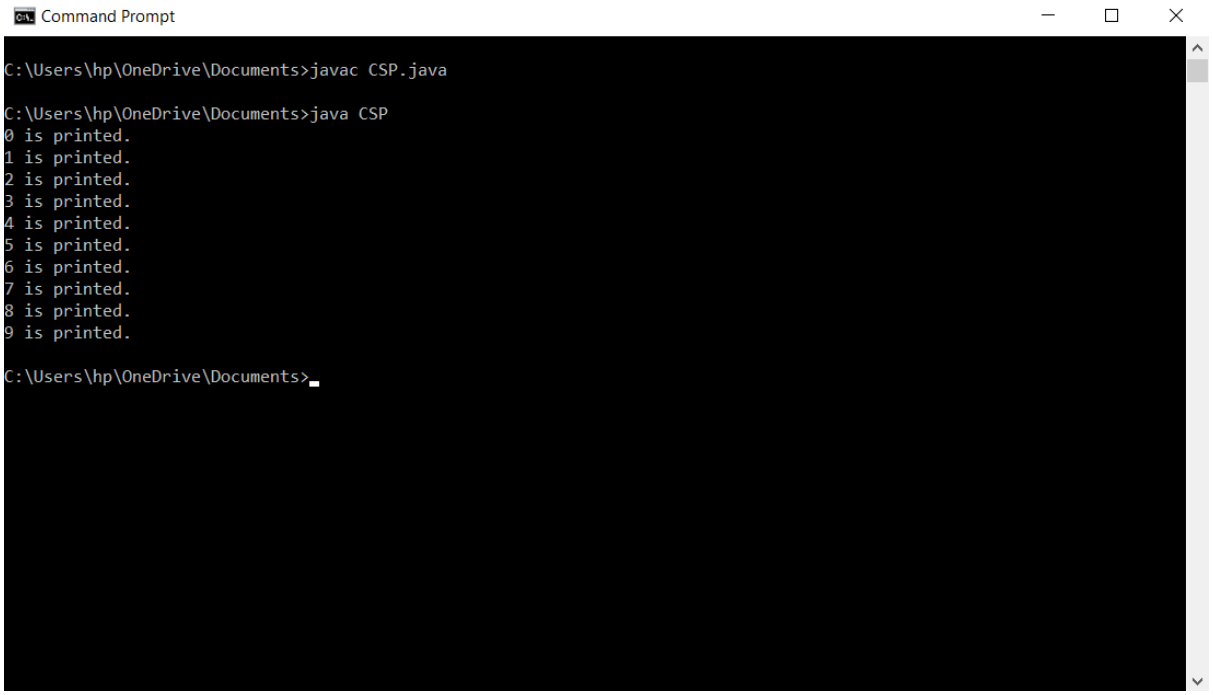
}

public int getStorage()

```
{
return mem;
}
}
class Counter extends Thread
{
Storage s;
Thread t;
Counter(Storage s)
{
this.s=s;
}
public void run()
{
int i=0;
while(i<10)
{
s.setStorage(i);
i++;
}
}
}
class Printer extends Thread
{
Storage s;
Thread t;
Printer(Storage s)
{
this.s=s;
}
public void run()
```

```
{
int i=9;
while(i>=0)
{
System.out.println(Math.abs(i-s.getStorage()) + " is printed.");
i--;
}
}
}
public class CSP {
public static void main(String[] args) {
Storage s = new Storage();
Counter ct = new Counter(s);
Printer pt = new Printer(s);
ct.start();
try {
ct.join();
}
catch(InterruptedException e)
{
System.out.println(e);
}
pt.start();
}
}
```

Output:



Q28) Create a library system with methods for returning and issuing a book. Apply multithreading synchronization concept and exception handling.

Code:

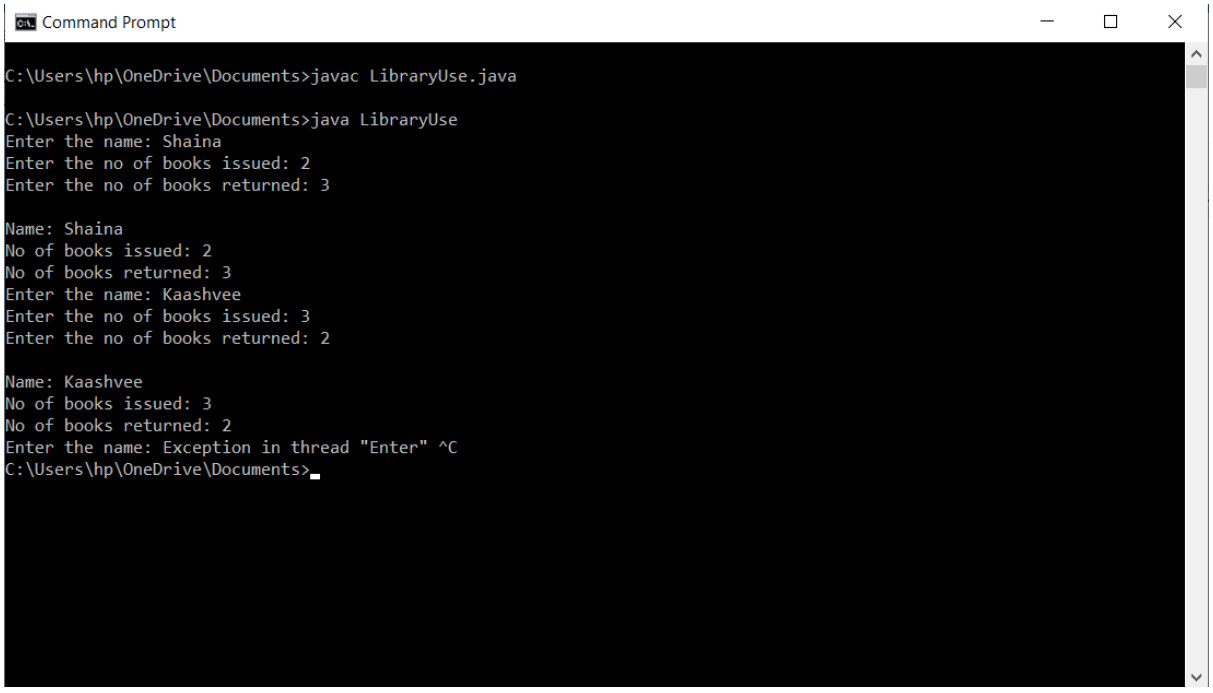
```
//LibraryUse.java

import java.util.Scanner;

class Library{

Scanner sc = new Scanner(System.in);

String name;

int iss;

int ret;

boolean val;

Library()

{

this.name="No Name";

this.iss=0;

this.ret=0;

this.val=false;

}
```

```
synchronized public void getEntry()
{
while(!val) {
try {
wait(10);
}
catch(InterruptedException e)
{
System.out.println(e);
}
System.out.print("Enter the name: ");
name=sc.next();
System.out.print("Enter the no of books issued: ");
iss=sc.nextInt();
System.out.print("Enter the no of books returned: ");
ret=sc.nextInt();
val=true;
notify();
}
}
synchronized public void showEntry()
{
while(val)
{
try {
wait(10);
}
catch(InterruptedException e)
{
System.out.println(e);
}
```

```
System.out.print("\nName: " + name);
System.out.print("\nNo of books issued: " + iss);
System.out.print("\nNo of books returned: " + ret);
System.out.print("\n");
val=false;
notify();
}
}
}
class Entry implements Runnable{
Library l;
Thread t;
Entry(Library l)
{
this.l=l;
t = new Thread (this,"Enter");
}
public void run()
{
while(true) {
l.getEntry();
}
}
}
class Shows implements Runnable{
Library l;
Thread t;
Shows(Library l)
{
this.l=l;
t = new Thread (this,"Show");
```

```
}
public void run()
{
while(true) {
l.showEntry();
}
}
}
public class LibraryUse {
public static void main(String[] args) {
Library l = new Library();
Entry ent = new Entry(l);
Shows sh = new Shows(l);
ent.t.start();
sh.t.start();
}
}
```

Output:



Q29) Write a program to do the following:

- **To store the text file contents line by line into an array.**
- **To find the longest word in the text file.**
- **To append the text to an existing file.**

Code:

1st and 2nd Part:

```
//FileHandling.java
import java.util.Scanner;
import java.util.ArrayList;
import java.io.IOException;
import java.io.File;
import java.io.FileReader;
public class FileHandling {
public static void main(String[] args) {
try {
System.out.println("1st Part: To Store The Contents Of The File In The Array:");
Scanner sc=new Scanner (System.in);
System.out.println("The file is reading the contents.");
FileReader f3 = new FileReader("D:/MyFile.txt");
Scanner fr = new Scanner(f3);
ArrayList <String> a = new ArrayList<>();
while(fr.hasNext()) {
String data= fr.next();
a.add(data);
}
System.out.println("The file has read the contents sucessfully");
fr.close();
System.out.println("The contents of the file are: ");
for(int i=0;i<a.size();i++)
{
System.out.println(a.get(i));
```

```
System.out.print("\n");
}
System.out.println("2nd Part: Find The longest String In The File:");
String s=a.get(0);
int len=a.get(0).length();
for(int i=1;i<a.size();i++)
{
if(len<a.get(i).length()) {
s=a.get(i);
len=a.get(i).length();
}
}
System.out.println("The longest string in the file is: "+ s);
}
catch(IOException e1) {
System.out.println("Exception Occured 1");
e1.printStackTrace();
}
}
}
```

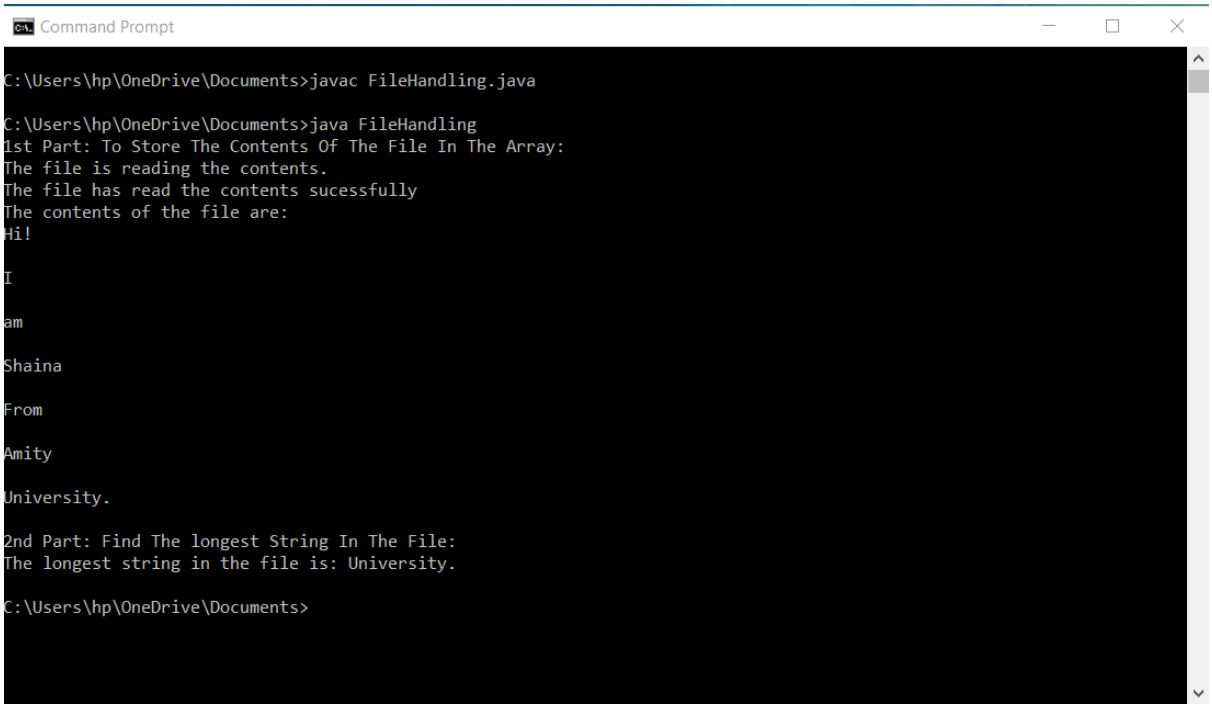
3rd Part:

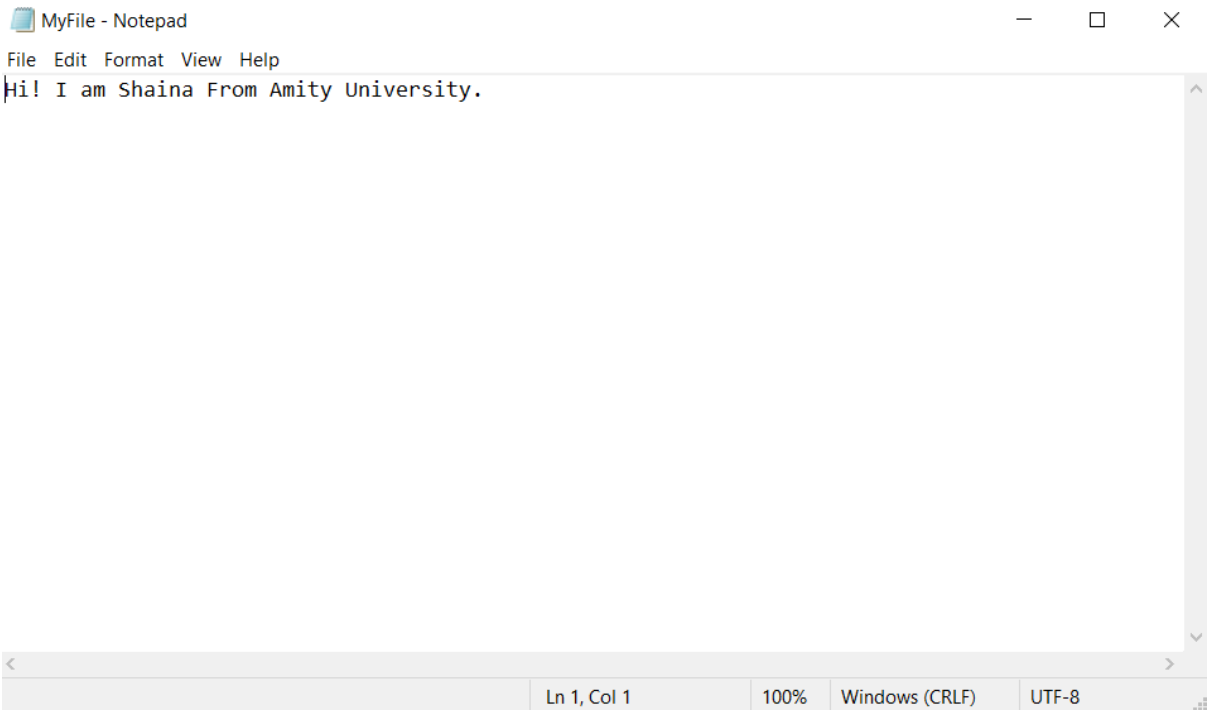
```
//FileHandling1.java
import java.util.Scanner;
import java.io.FileWriter;
import java.io.IOException;
public class FileHandling1{
public static void main(String args[]){
try{
Scanner sc = new Scanner(System.in);
System.out.println("3rd Part: To Append The Text In The File:");
FileWriter f4 = new FileWriter("D:/MyFile.txt",true);
```

```
System.out.println("Enter the string to be added: ");
String s1=sc.nextLine();
f4.write(s1);
f4.close();
System.out.println("The file has been appended sucessfully.");
}
catch(IOException e){
System.out.println("Exception Occured 1");
e.printStackTrace();
}
}
}
```

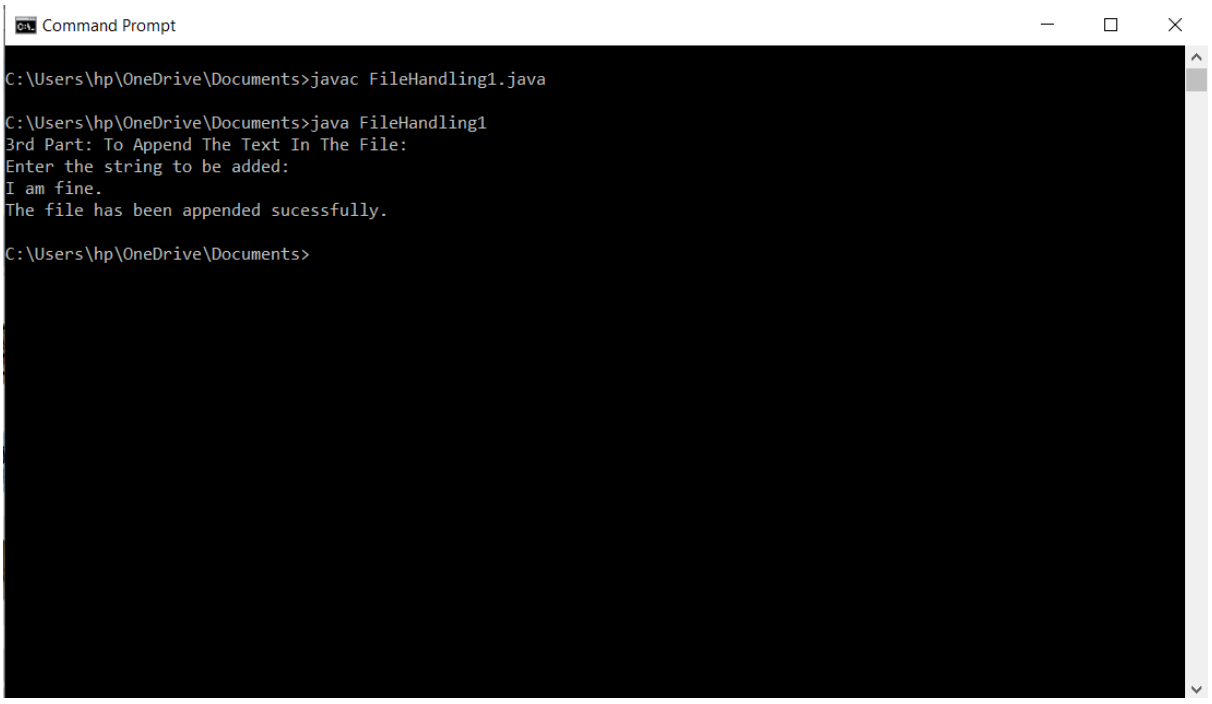
Output:

1st and 2nd Part:




MyFile - Notepad
File Edit Format View Help
Hi! I am Shaina From Amity University.
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

3rd Part:


Command Prompt
C:\Users\hp\OneDrive\Documents>javac FileHandling1.java
C:\Users\hp\OneDrive\Documents>java FileHandling1
3rd Part: To Append The Text In The File:
Enter the string to be added:
I am fine.
The file has been appended sucessfully.
C:\Users\hp\OneDrive\Documents>

Q 30) Write a simple calculator using AWT.

Code:

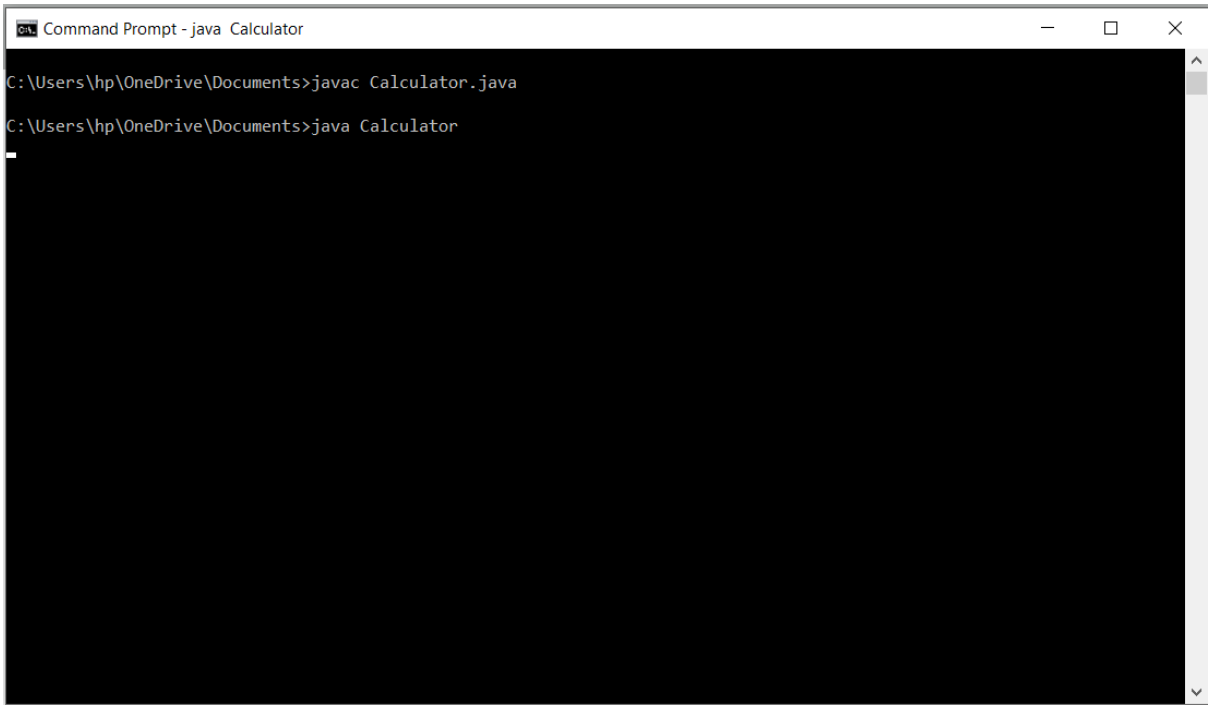
```
import java.awt.*;
import java.awt.event.*;
public class Calculator extends Frame implements ActionListener{

/**
*
*/
private static final long serialVersionUID = 1L;
TextField t;
Panel p;
String str[]= {"7","8","9","+",
"4","5","6","-",
"1", "2", "3","*",
"C","0","=","/"};
Button b[] = new Button[16];
```

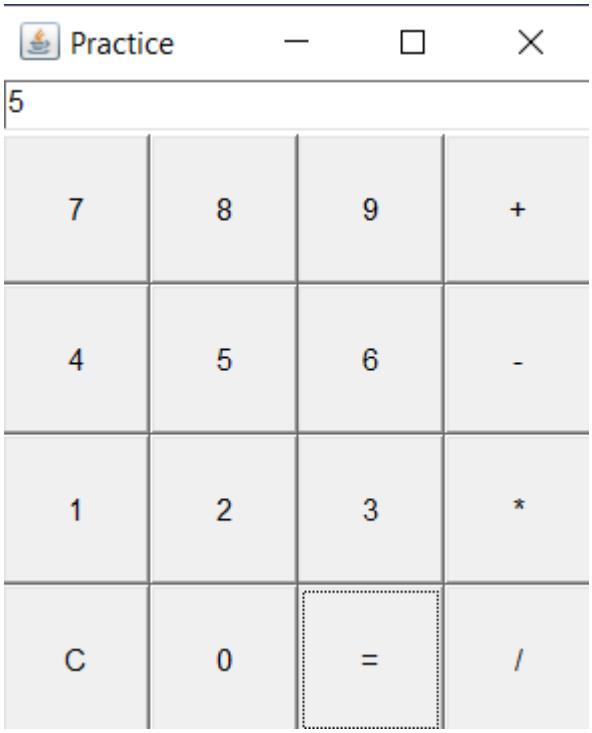
```
int num1=0,num2=0,result=0;
char op;
public Calculator() {
t= new TextField(10);
p= new Panel();
add(t,"North");
add(p,"Center");
p.setLayout(new GridLayout(4,4));
for(int i=0;i<16;i++) {
b[i]=new Button(str[i]);
b[i].addActionListener(this);
p.add(b[i]);
}
addWindowListener(new Window());
}
public void actionPerformed(ActionEvent e) {
// TODO Auto-generated method stub
String s=e.getActionCommand();
if(s.equals("+")) {
op='+';
num1=Integer.parseInt(t.getText());
t.setText("");
}
else if(s.equals("-")) {
op='-';
num1=Integer.parseInt(t.getText());
t.setText("");
}
else if(s.equals("*")) {
op='*';
num1=Integer.parseInt(t.getText());
```

```
t.setText("");
}
else if(s.equals("/")) {
op='/';
num1=Integer.parseInt(t.getText());
t.setText("");
}
else if(s.equals("=")) {
num2=Integer.parseInt(t.getText());
switch(op) {
case '+': result=num1+num2;
break;
case '-': result=num1-num2;
break;
case '*': result=num1*num2;
break;
case '/': result=num1/num2;
break;
}
t.setText(result+"");
result=0;
}
else if(s.equals("C")) {
t.setText("");
num1=num2=result=0;
}
else {
t.setText(t.getText()+s);
}

}
```

```
public static void main(String[] args) {
// TODO Auto-generated method stub
Calculator c= new Calculator();
c.setTitle("Practice");
c.setSize(new Dimension(250,300));
c.setVisible(true);
}
}
```

Output:




Practice
5
7
8
9
+
4
5
6
-
1
2
3
*
C
0
=
/