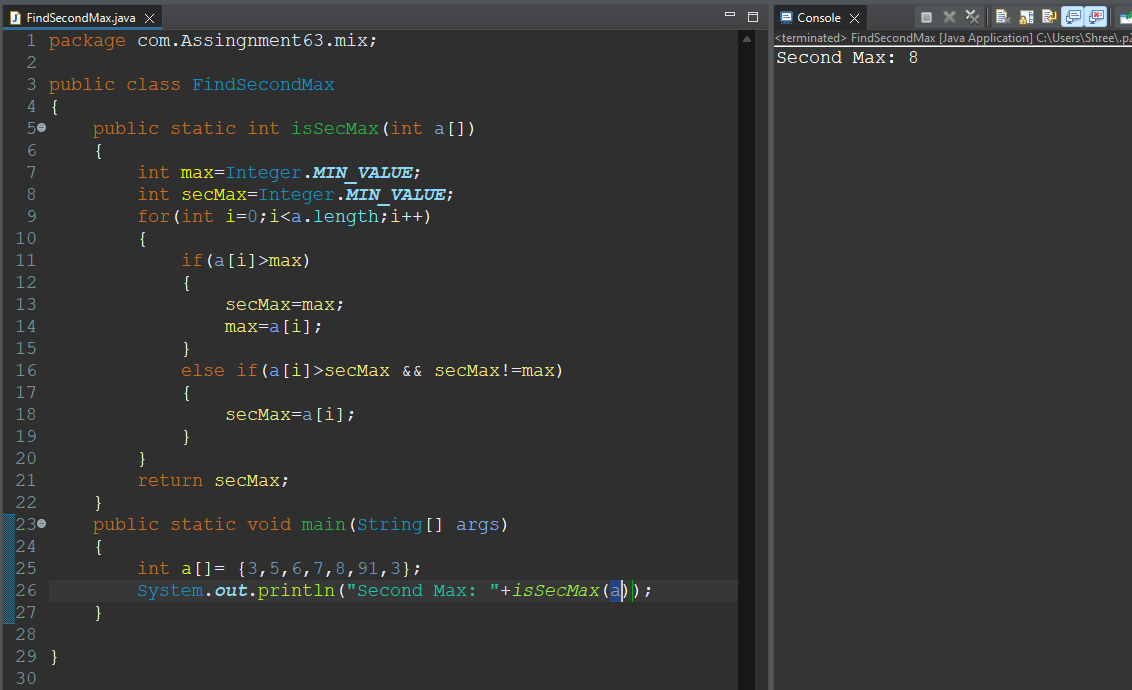
**Assignment No:-63**

Name:-Suryawanshi Sangramsingh Sambhaji

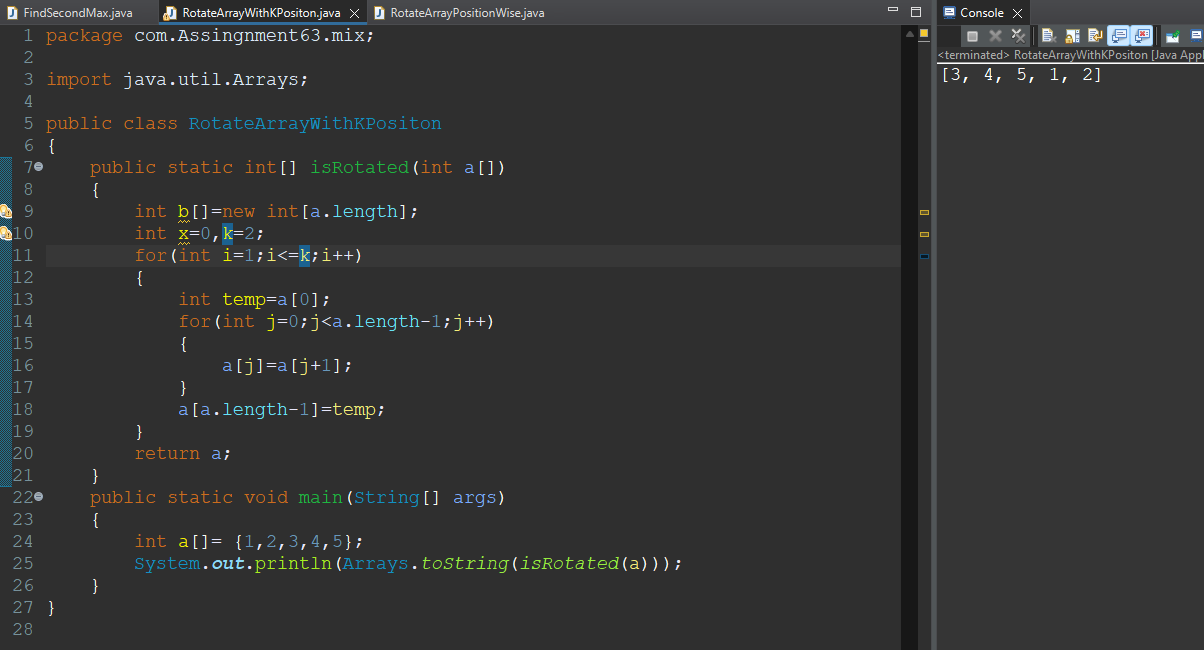
Batch: - Delta - DCA (Java) 2024 Date:-14/8/2024

**Array:**

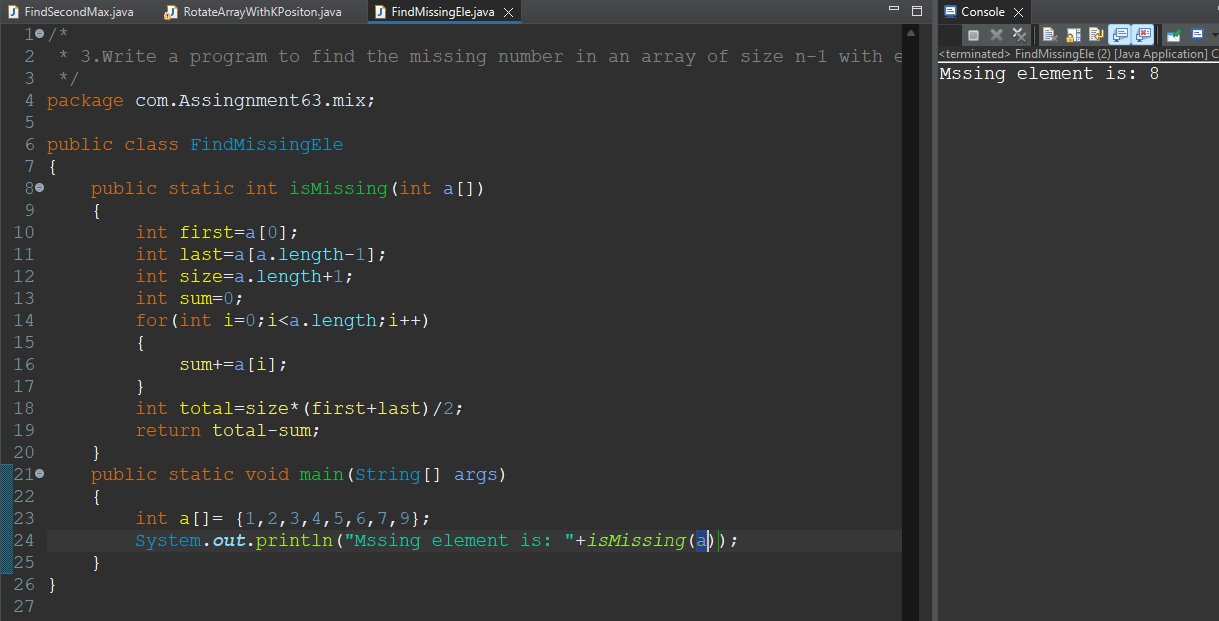
**1. Write a program to find the second largest element in an array.**

****

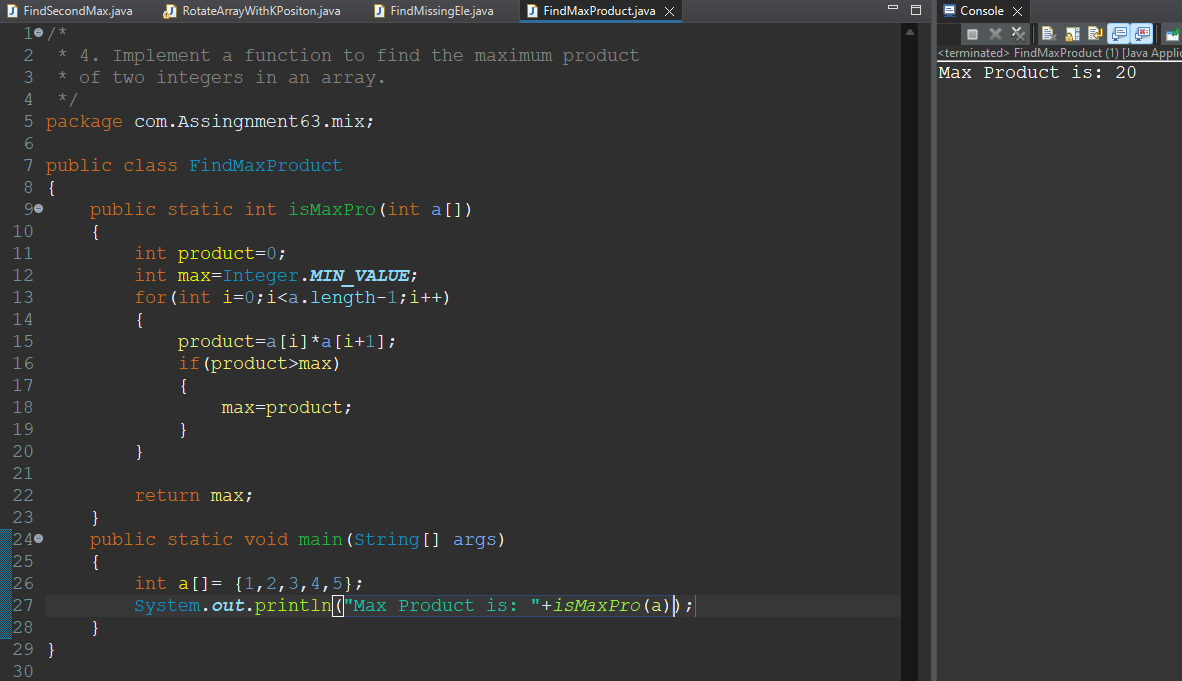
**2.Implement a function to rotate an array by k positions.**

****

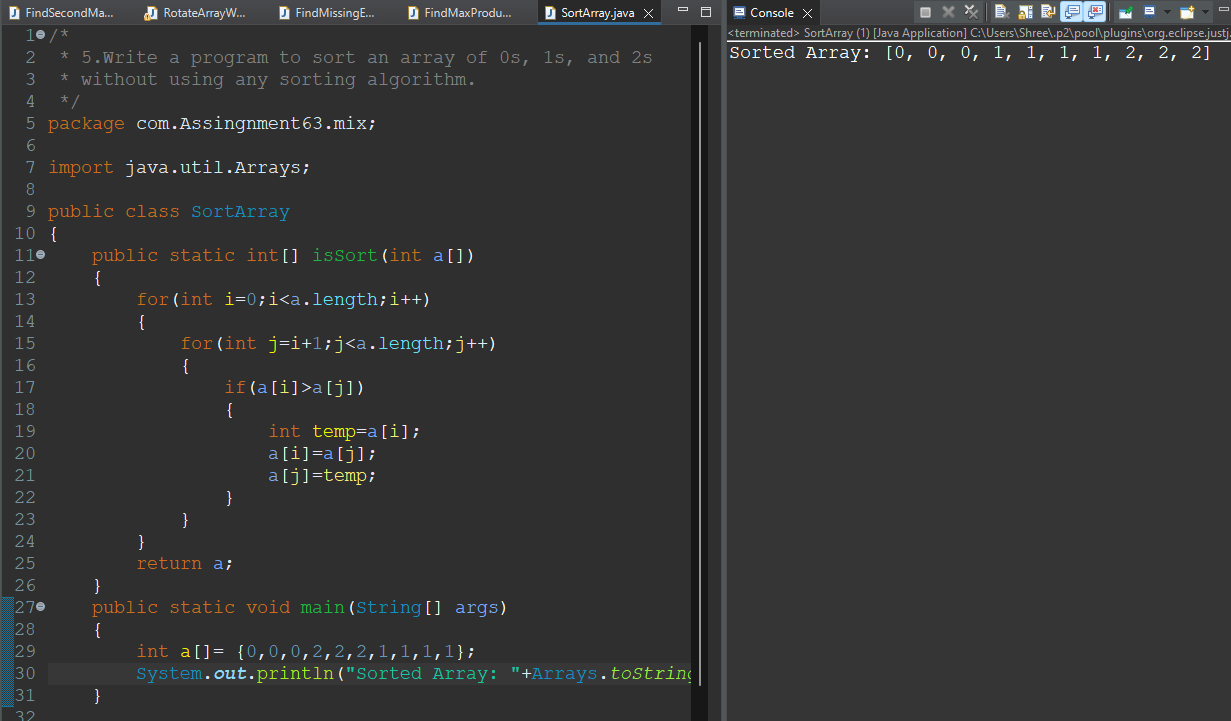
**3.Write a program to find the missing number in an array of size n-1 with elements from 1 to n.**

****

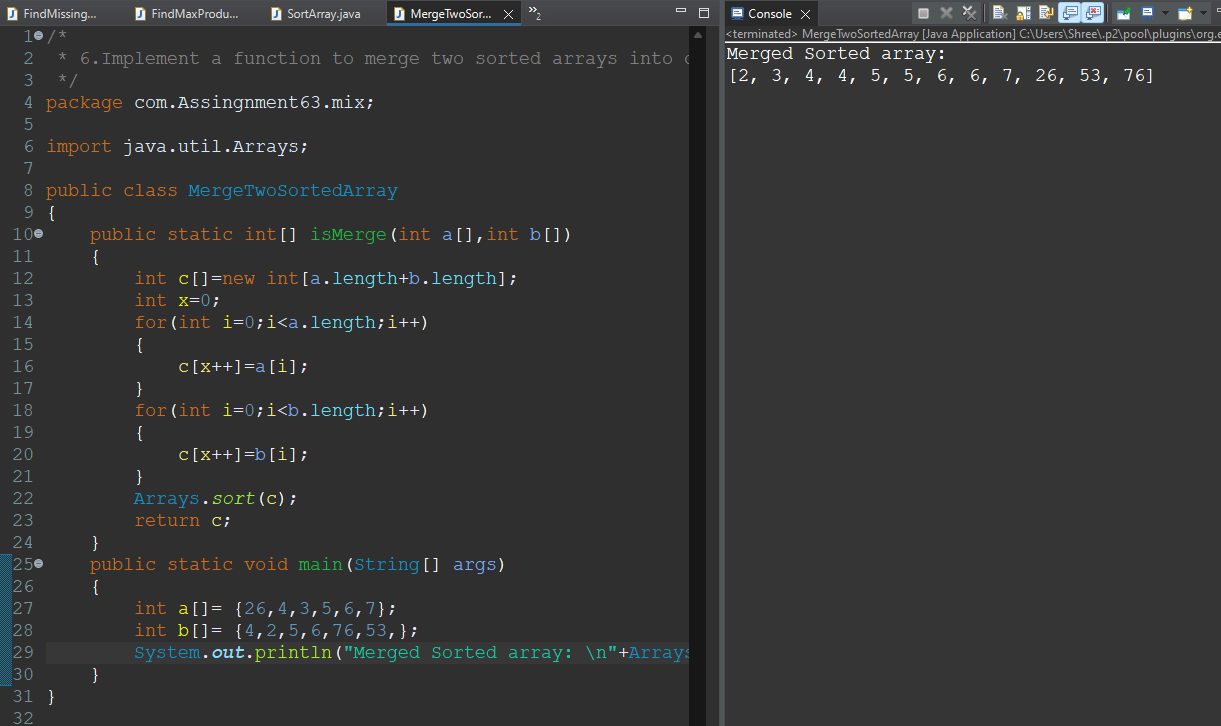
**4. Implement a function to find the maximum product of two integers in an array.**

****

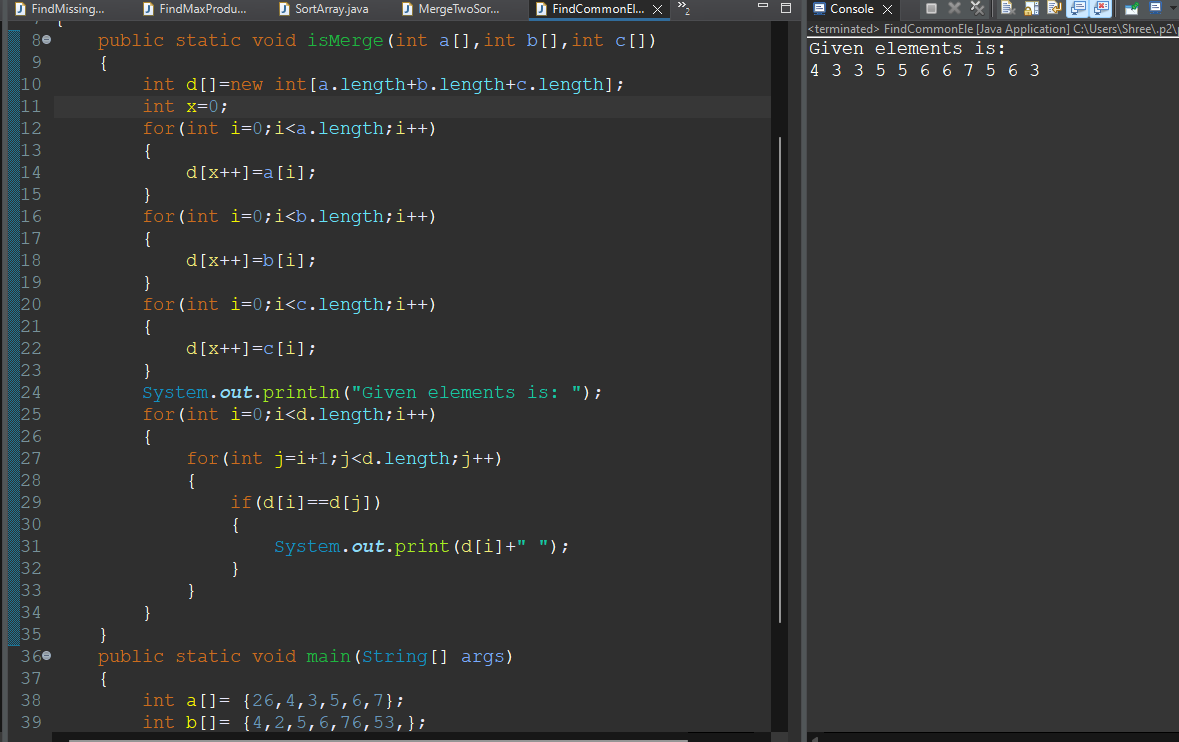
**5.Write a program to sort an array of 0s, 1s, and 2s without using any sorting algorithm.**

****

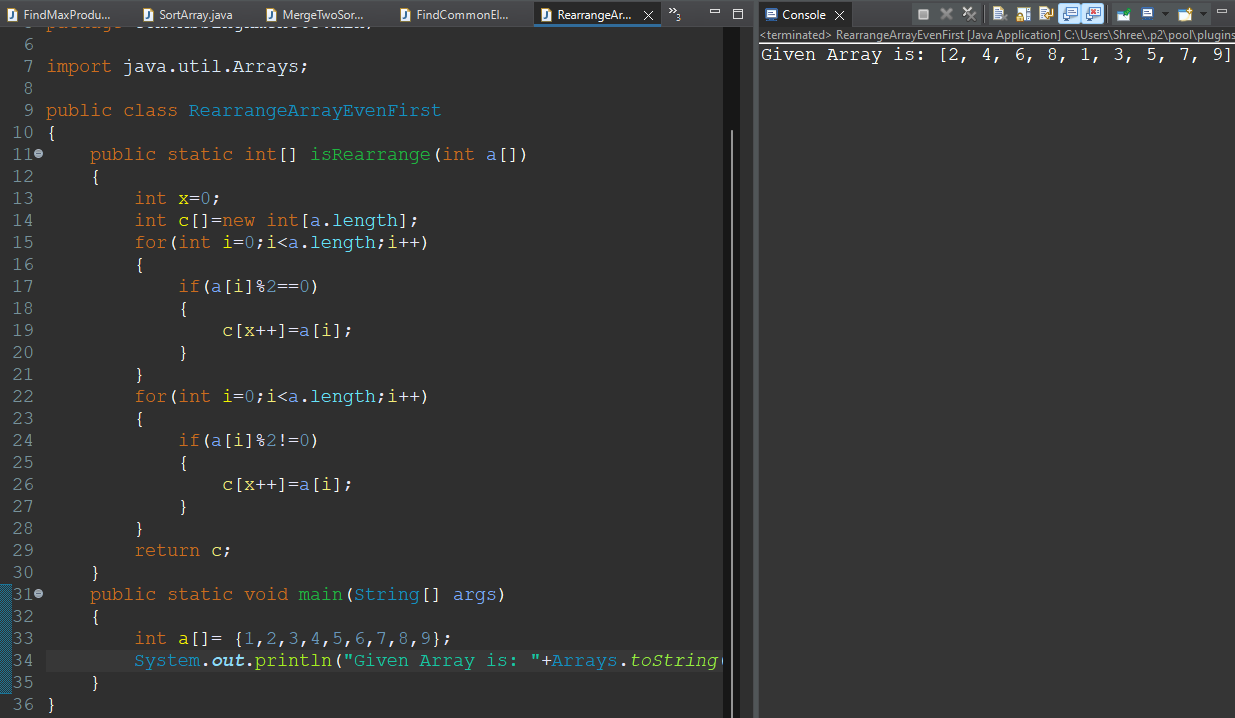
**6.Implement a function to merge two sorted arrays into one sorted array.**

****

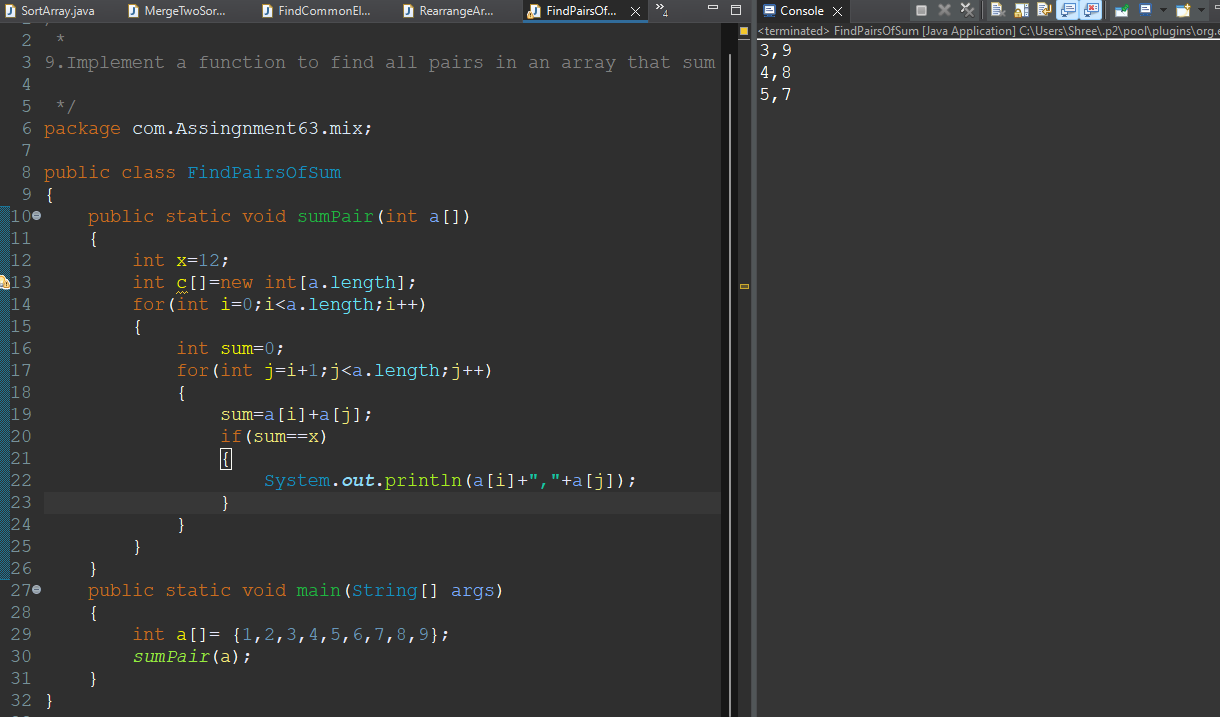
**7.Write a program to find the common elements in three sorted arrays.**

****

**8.Write a program to rearrange the array such that all even numbers appear before odd numbers.**

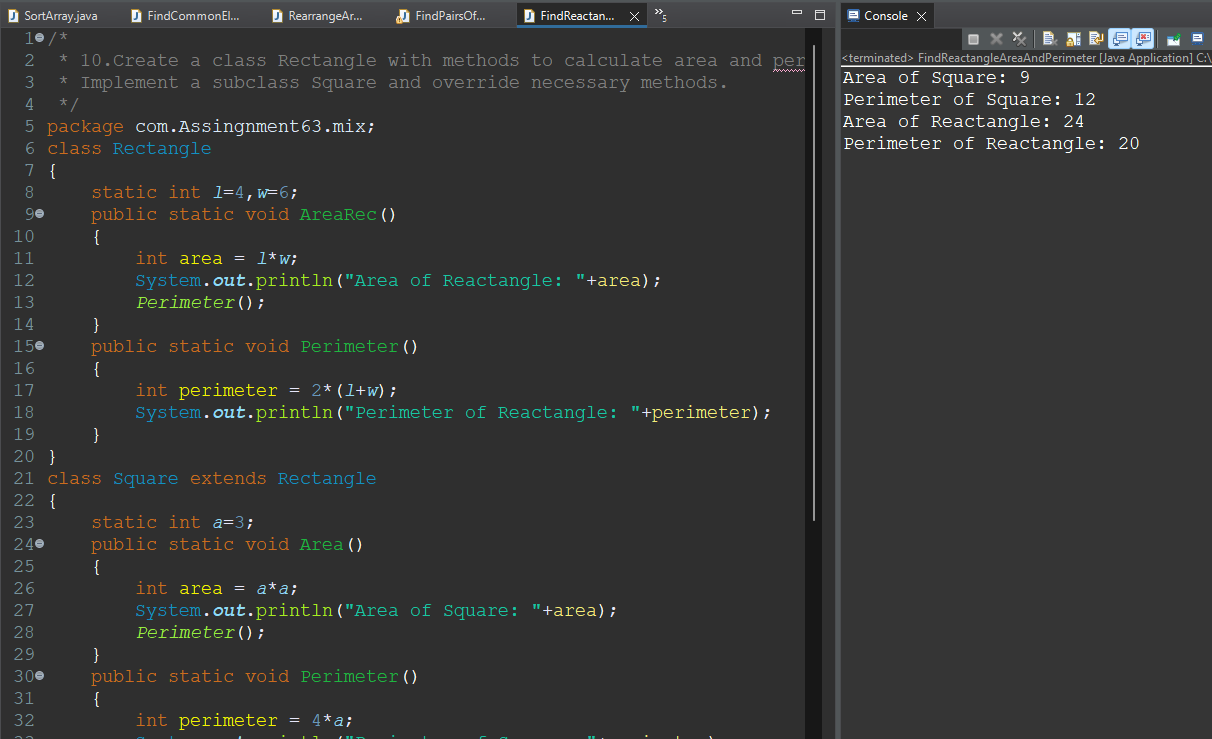
****

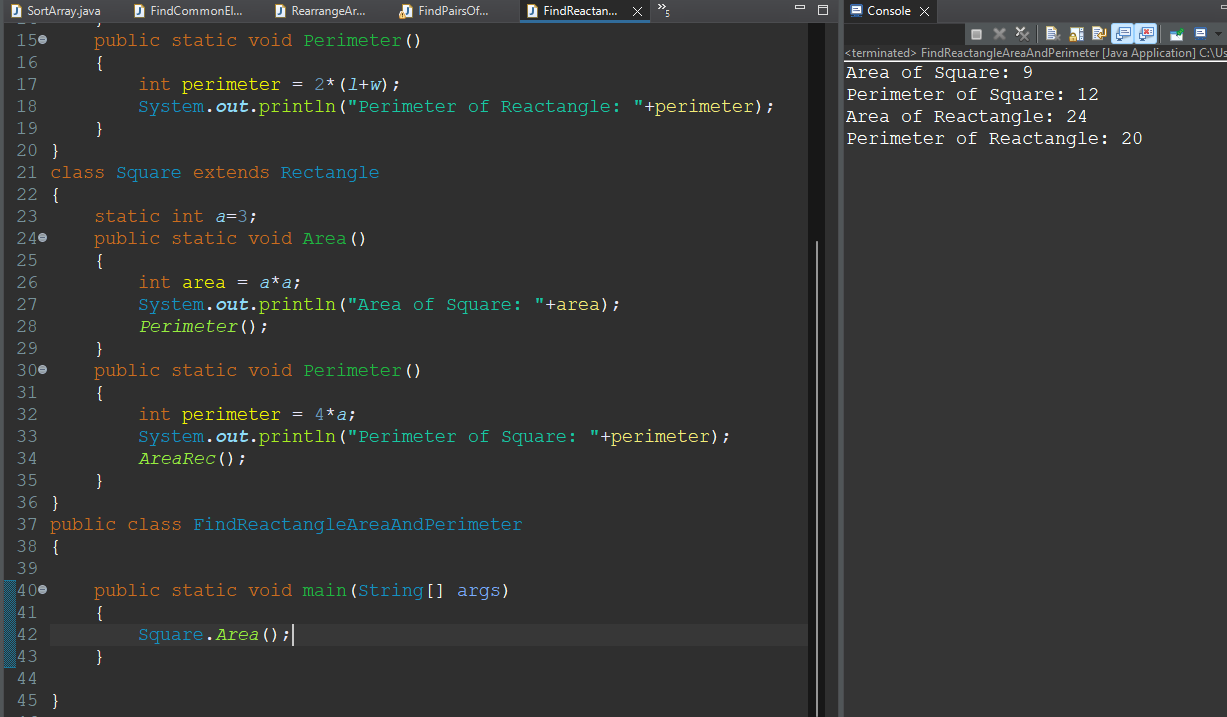
**9.Implement a function to find all pairs in an array that sum up to a specific target.**

****

**OOPs**

**10.Create a class Rectangle with methods to calculate area and perimeter. Implement a subclass Square and override necessary methods.**

****

****

**11.Design a class BankAccount with methods to deposit, withdraw, and check balance. Implement subclasses SavingsAccount and CheckingAccount.**

/\*

\* 11.Design a class BankAccount with methods to deposit, withdraw, and check balance.

\* Implement subclasses SavingsAccount and CheckingAccount.

\*/

package com.Assingnment63.mix;

import java.util.Scanner;

class BankAccount

{

static Scanner *sc*;

static long *depo*=0;

static long *withd*=0;

static long *bal*=0;

static long *cnt*=0;

public static void deposite()

{

*sc*= new Scanner(System.***in***);

*cnt*++;

System.***out***.println("Enter Deposite Money: ");

*depo* = *sc*.nextInt();

System.***out***.println("Amount Deposited Suceessfully: "+*depo*);

System.***out***.println("--------------------<>-------------------");

}

public static void withdraw()

{

if(*bal*>0)

{

*sc*= new Scanner(System.***in***);

*cnt*=0;

System.***out***.println("Enter Withdrawal Money: ");

*withd* = *sc*.nextInt();

System.***out***.println("Amount Withdrawal Suceessfully: "+*withd*);

System.***out***.println("--------------------<>-------------------");

}

else

{

System.***out***.println("Insufficient Balance: "+*bal*);

}

}

public static void checkBalance()

{

if(*cnt*>=1)

{

long c=*bal*+=*depo*;

System.***out***.println("Your Balance is: "+(c));

}

else

{

long w=*bal*-=*withd*;

System.***out***.println("Your Balance is: "+(w));

}

}

}

public class CheckBalance {

public static void main(String[] args)

{

Scanner sc = new Scanner(System.***in***);

int ch =-1;

while(ch!=0)

{

System.***out***.println("Enter your choice\n1.Deposite\n2.Withdraw\n3.Check Balance\n4.Exit");

ch=sc.nextInt();

System.***out***.println("---------------<>-----------------");

switch(ch)

{

case 1: BankAccount.*deposite*();

break;

case 2: BankAccount.*withdraw*();;

break;

case 3: BankAccount.*checkBalance*();;

break;

case 4: ch=0;

System.***out***.println("Processes Terminated.....!");

break;

}

}

System.***out***.println("-------------------<>------------------");

}

}

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

3

---------------<>-----------------

Your Balance is: 0

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

2

---------------<>-----------------

Insufficient Balance: 0

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

1

---------------<>-----------------

Enter Deposite Money:

10000000

Amount Deposited Suceessfully: 10000000

--------------------<>-------------------

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

3

---------------<>-----------------

Your Balance is: 10000000

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

2

---------------<>-----------------

Enter Withdrawal Money:

780000

Amount Withdrawal Suceessfully: 780000

--------------------<>-------------------

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

3

---------------<>-----------------

Your Balance is: 9220000

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

2

---------------<>-----------------

Enter Withdrawal Money:

7464683

Amount Withdrawal Suceessfully: 7464683

--------------------<>-------------------

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

3

---------------<>-----------------

Your Balance is: 1755317

Enter your choice

1.Deposite

2.Withdraw

3.Check Balance

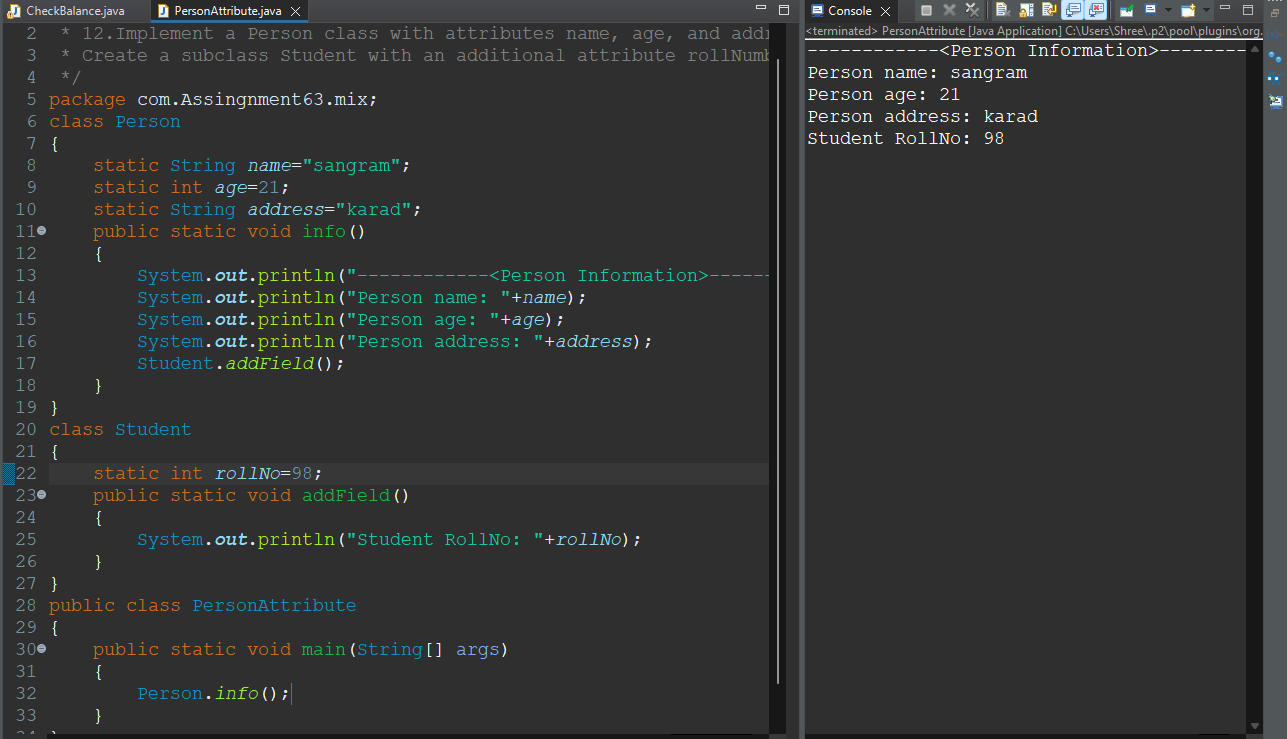
4

---------------<>-----------------

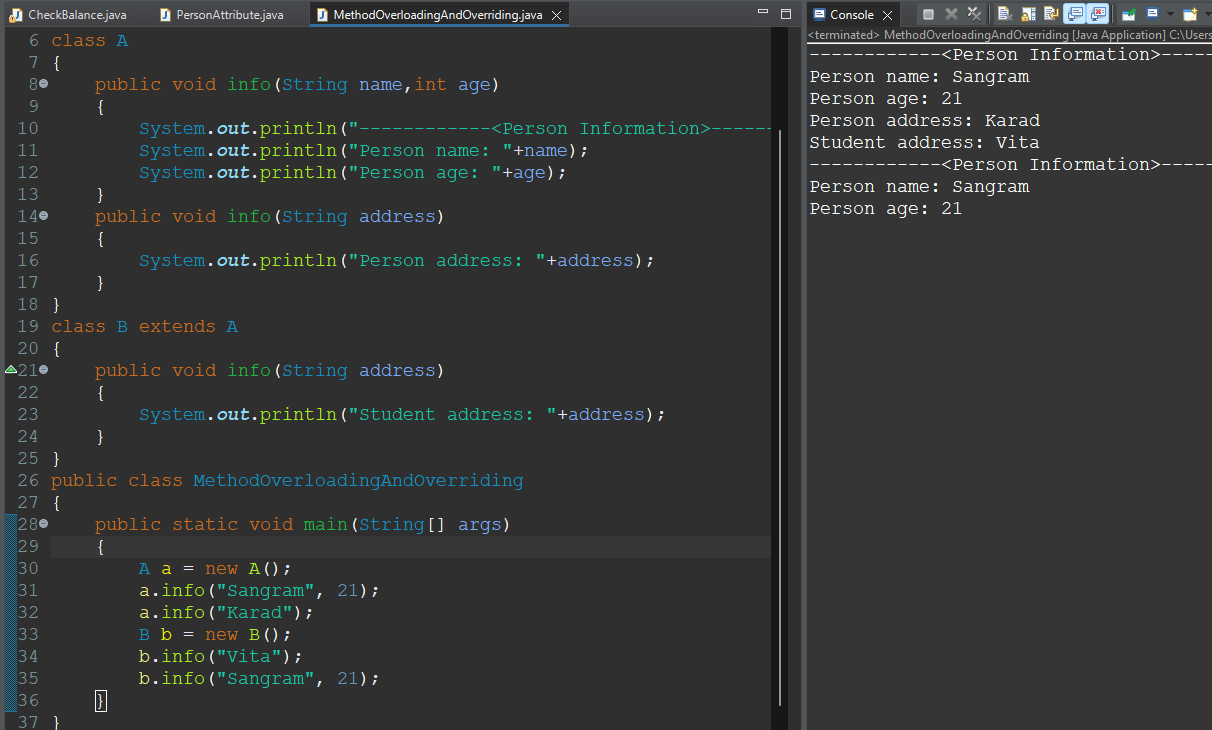
Processes Terminated.....!

-------------------<>------------------

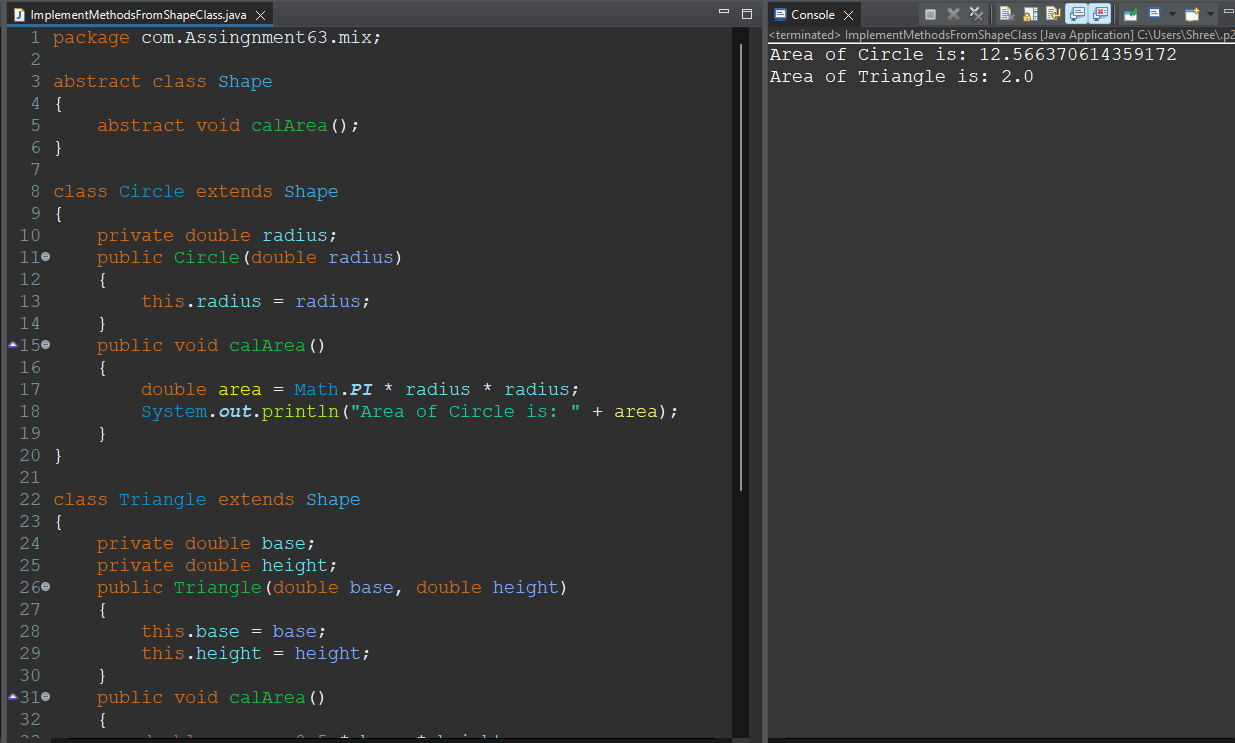
**12.­­Implement a Person class with attributes name, age, and address. Create a subclass Student with an additional attribute rollNumber.**

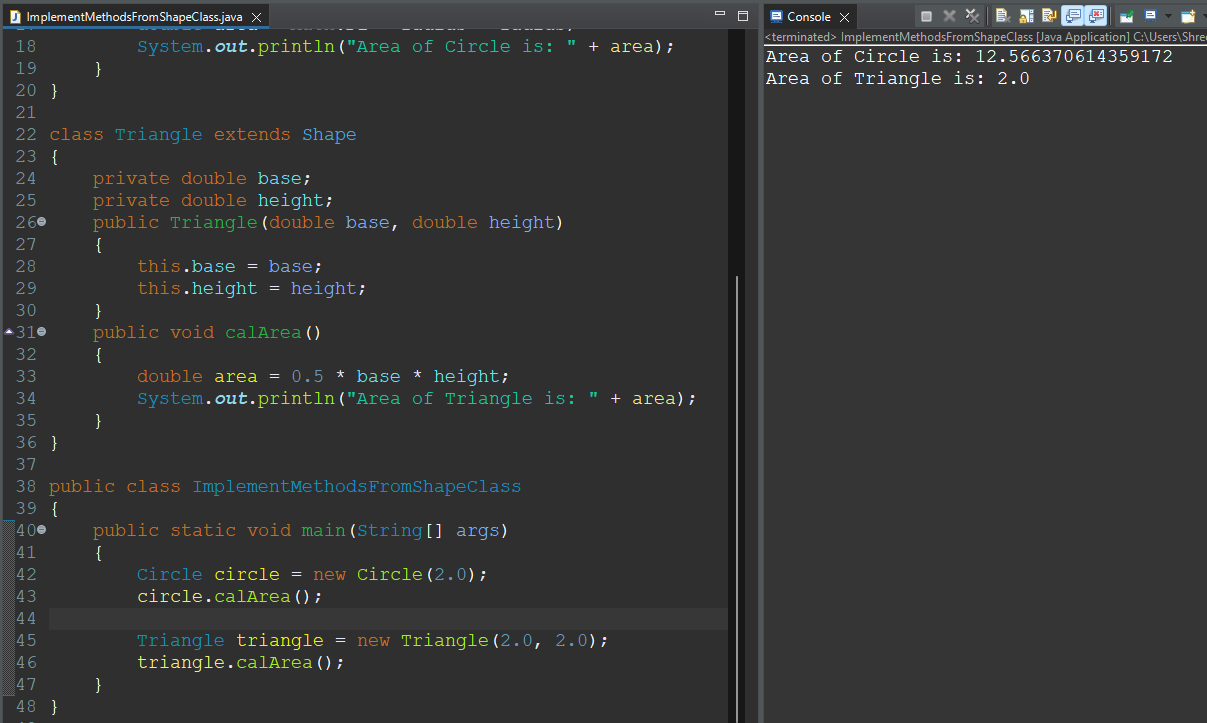
****

**13.Write a program demonstrating the concept of method overloading and method overriding**

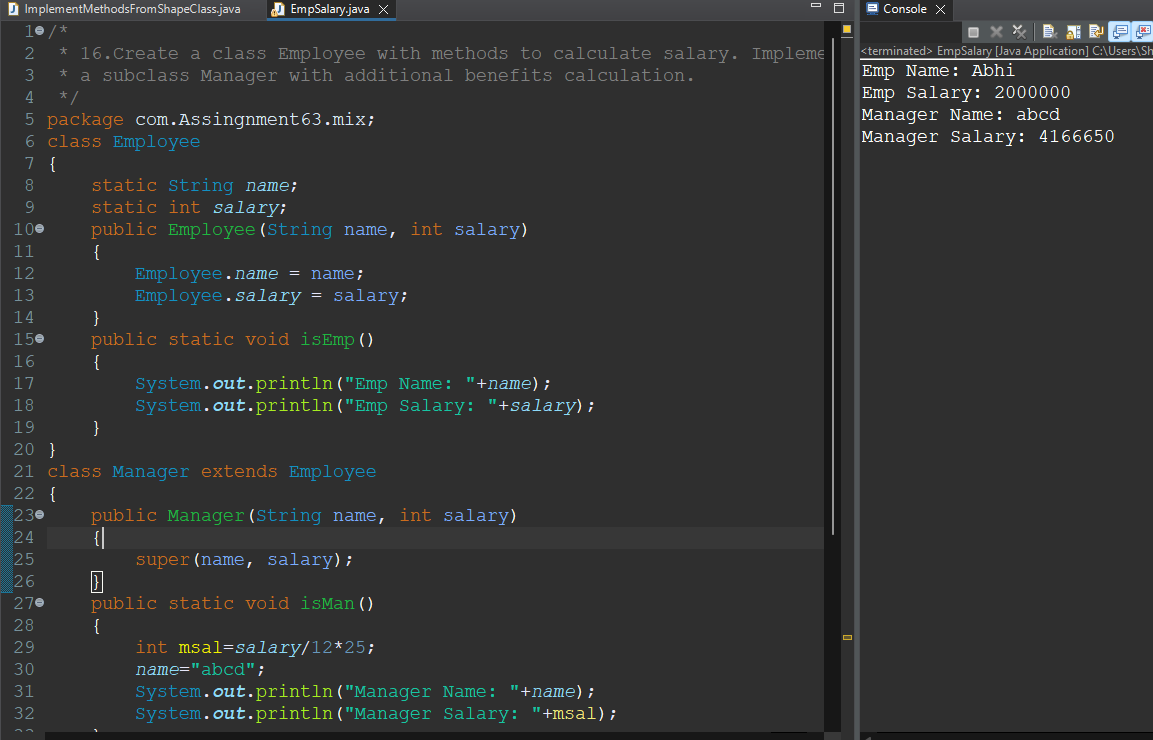
****

**15.Implement a Shape class with methods to calculate area. Create subclasses Circle and Triangle with specific implementations.**

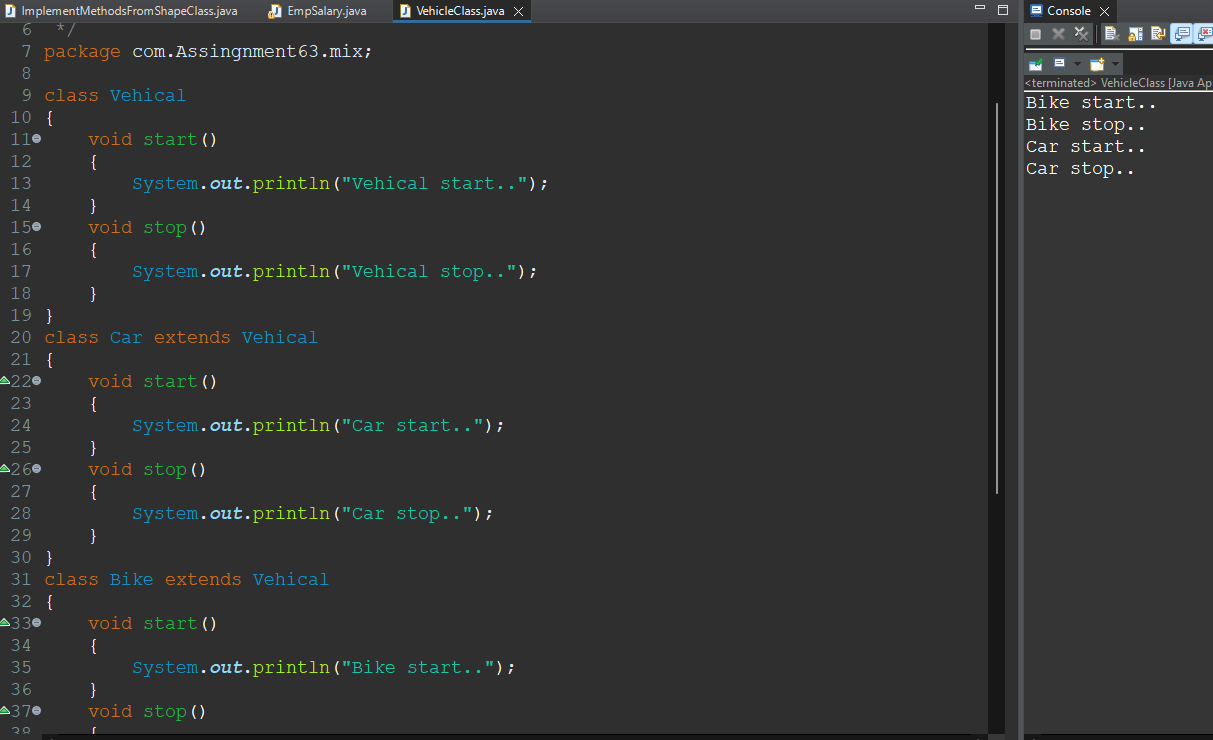
****

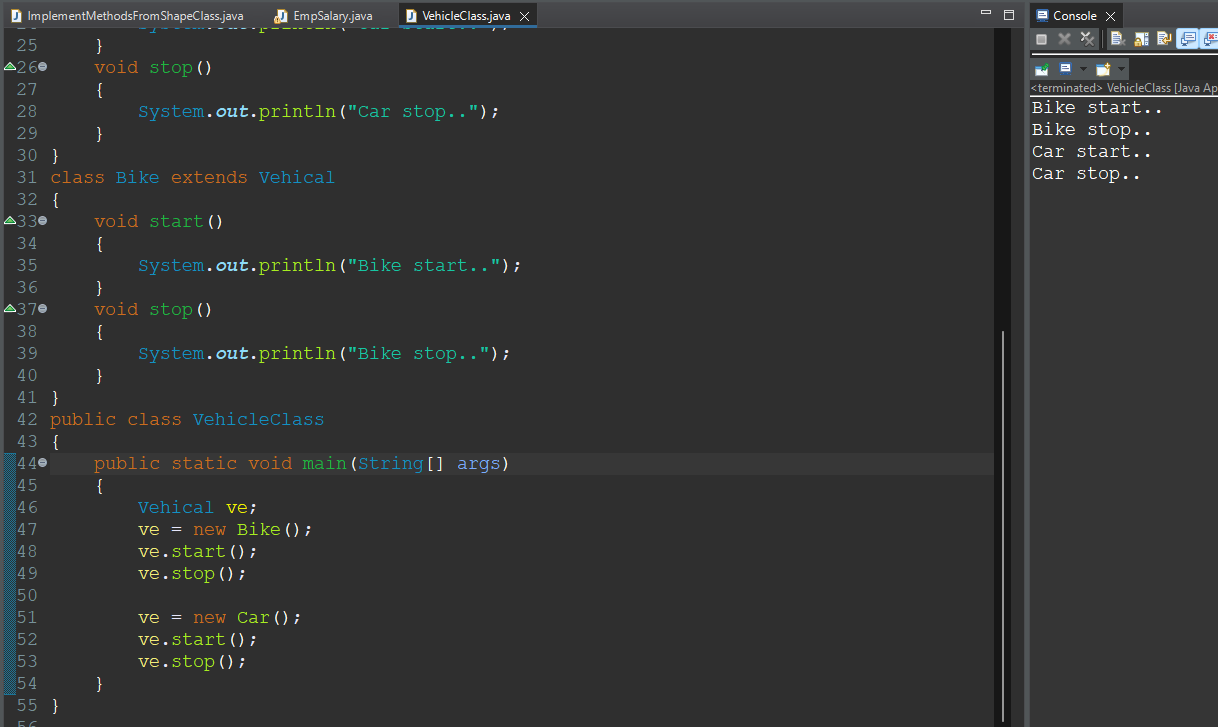
****

**16.Create a class Employee with methods to calculate salary. Implement a subclass Manager with additional benefits calculation.**

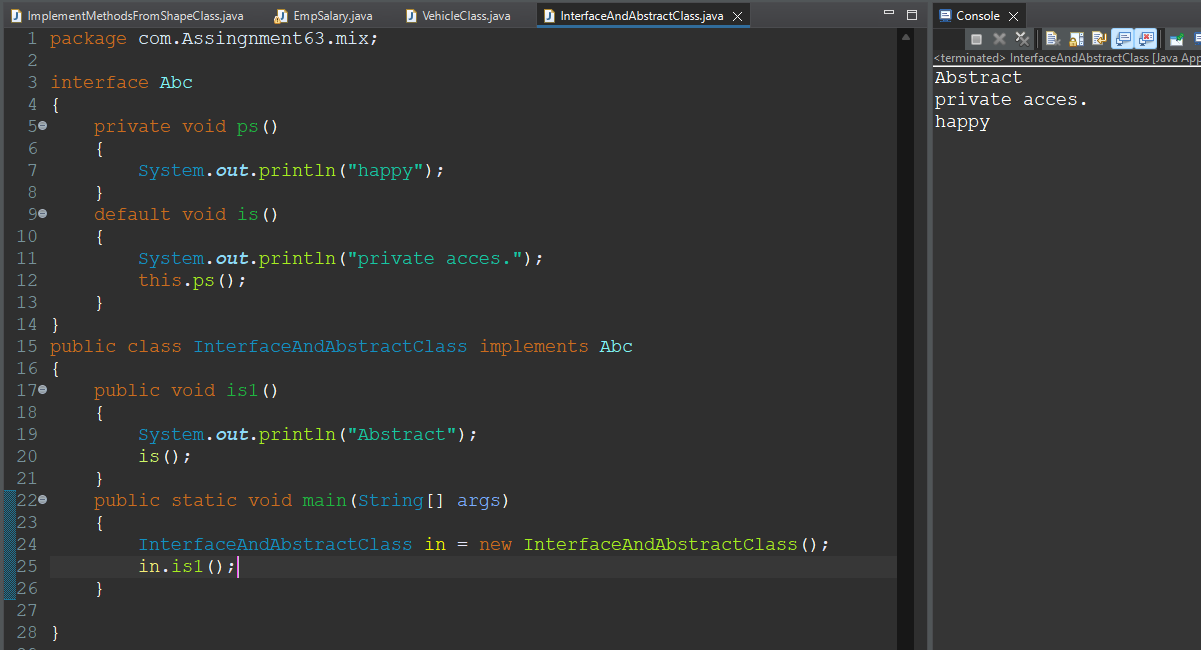


**17.Implement a Vehicle class with methods to start and stop. Create subclasses Car and Bike and demonstrate polymorphism.**

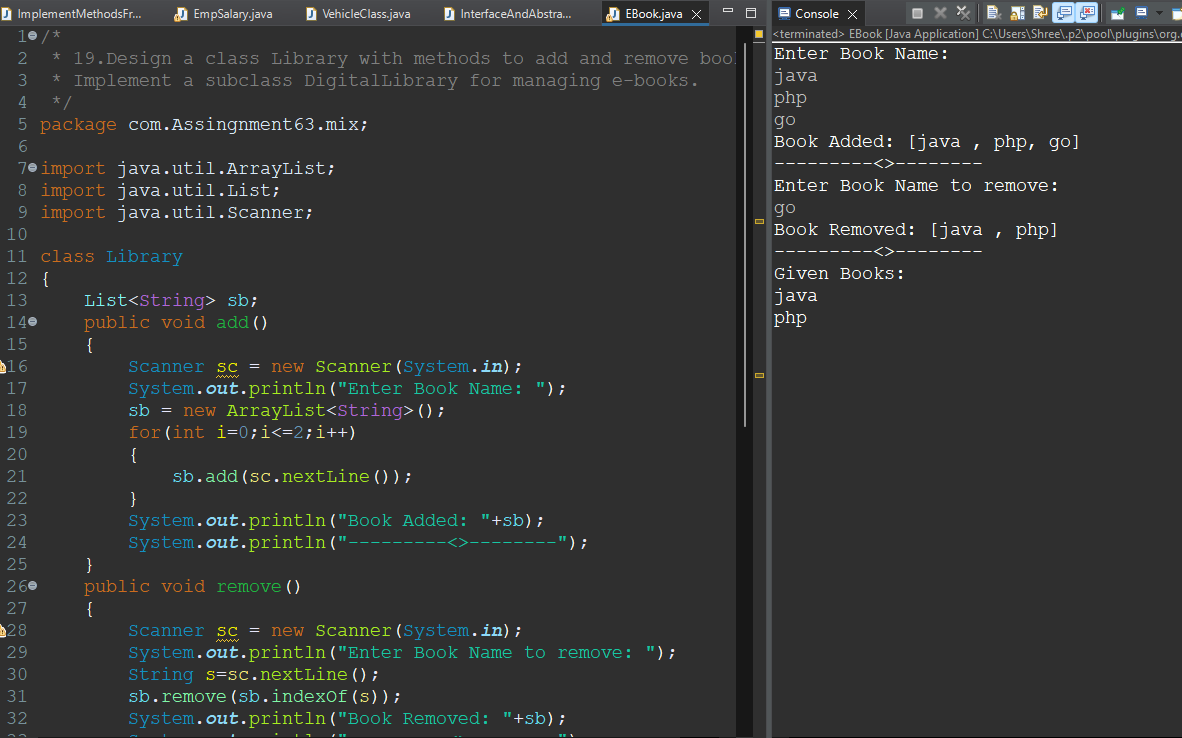
****

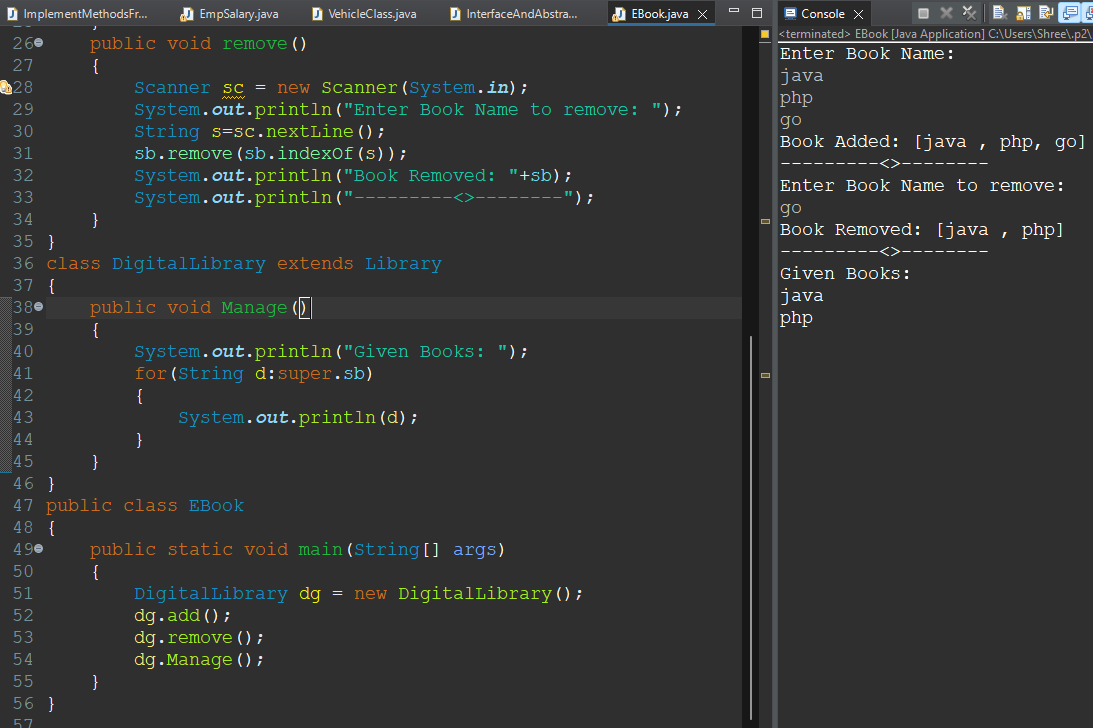
****

**18.Write a program to demonstrate the use of abstract classes and interfaces.**

****

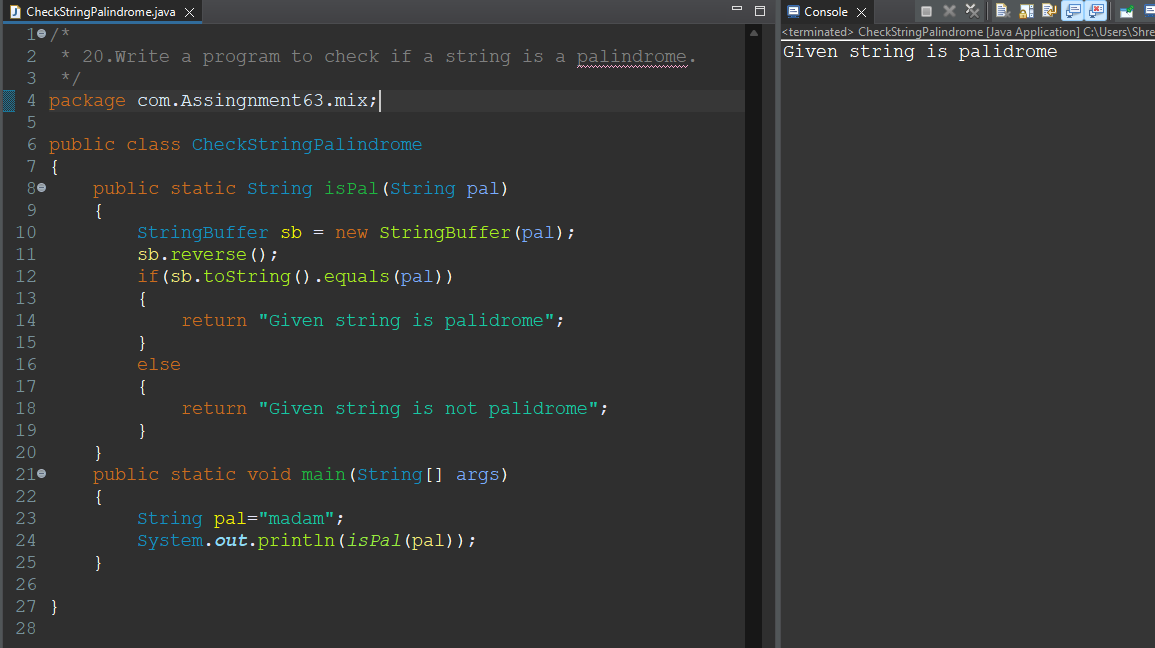
**19.Design a class Library with methods to add and remove books. Implement a subclass DigitalLibrary for managing e-books.**

****

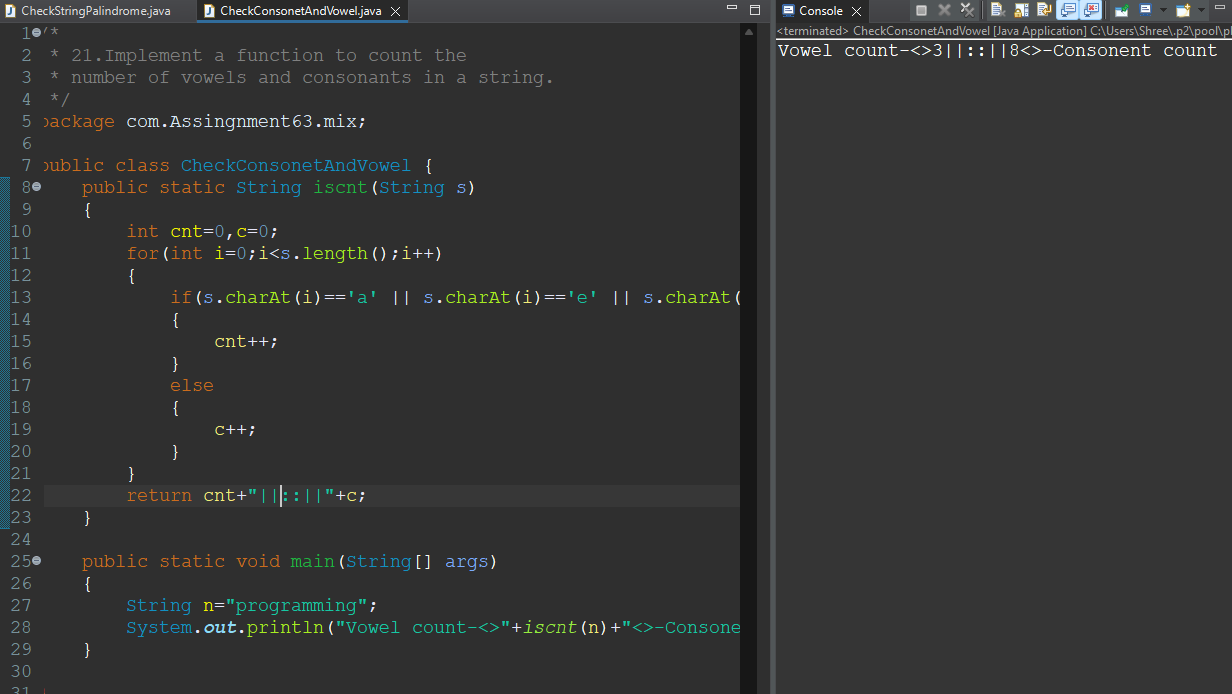
****

**String**

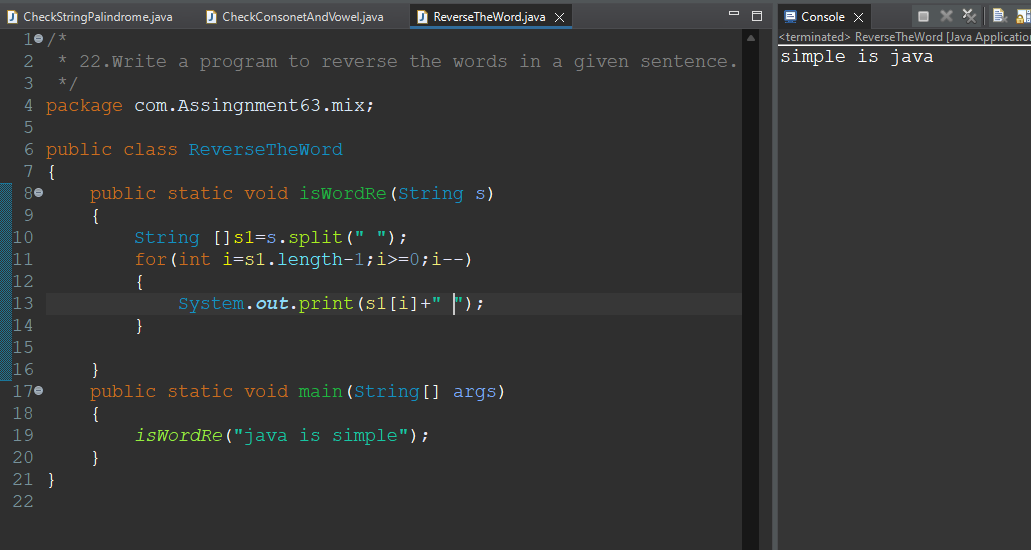
**20.Write a program to check if a string is a palindrome.**

****

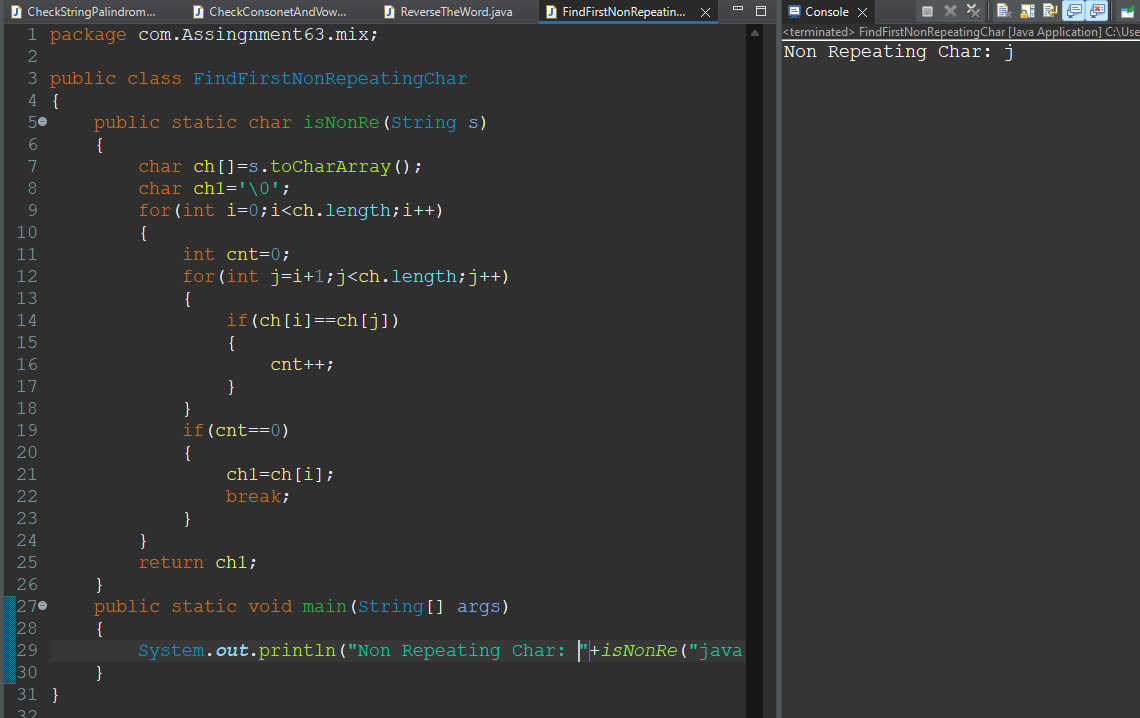
**21.Implement a function to count the number of vowels and consonants in a string.**

****

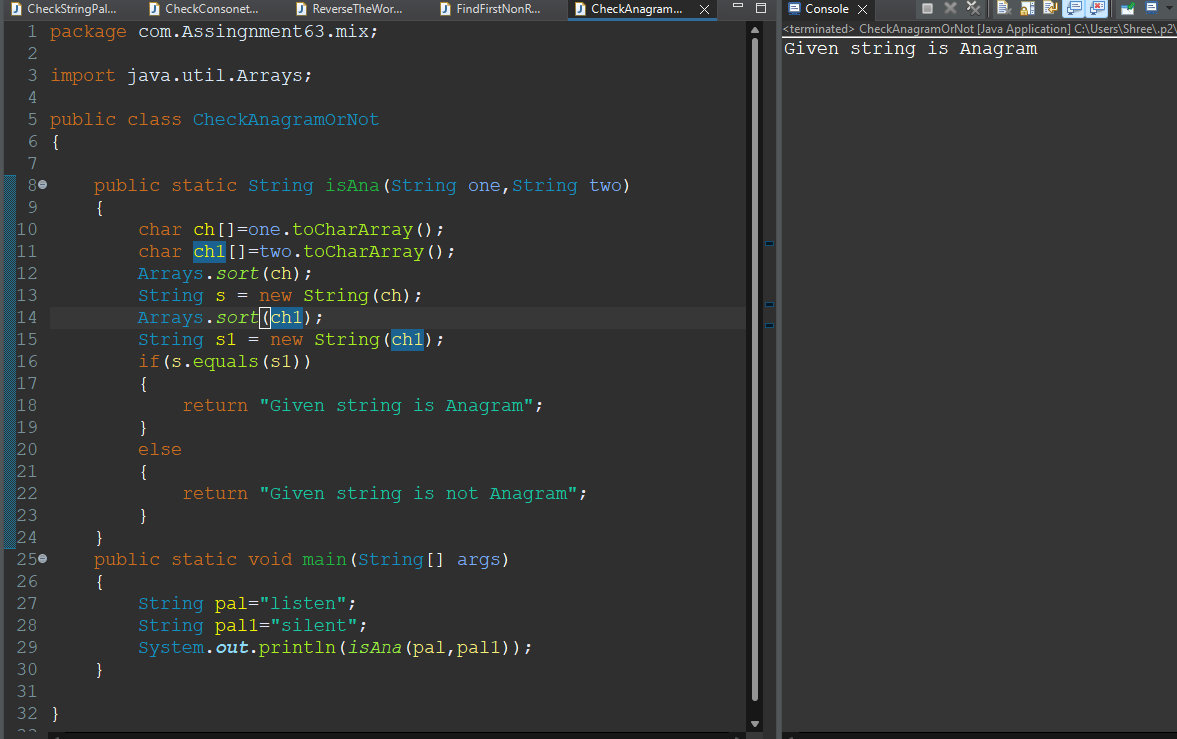
**22.Write a program to reverse the words in a given sentence.**

****

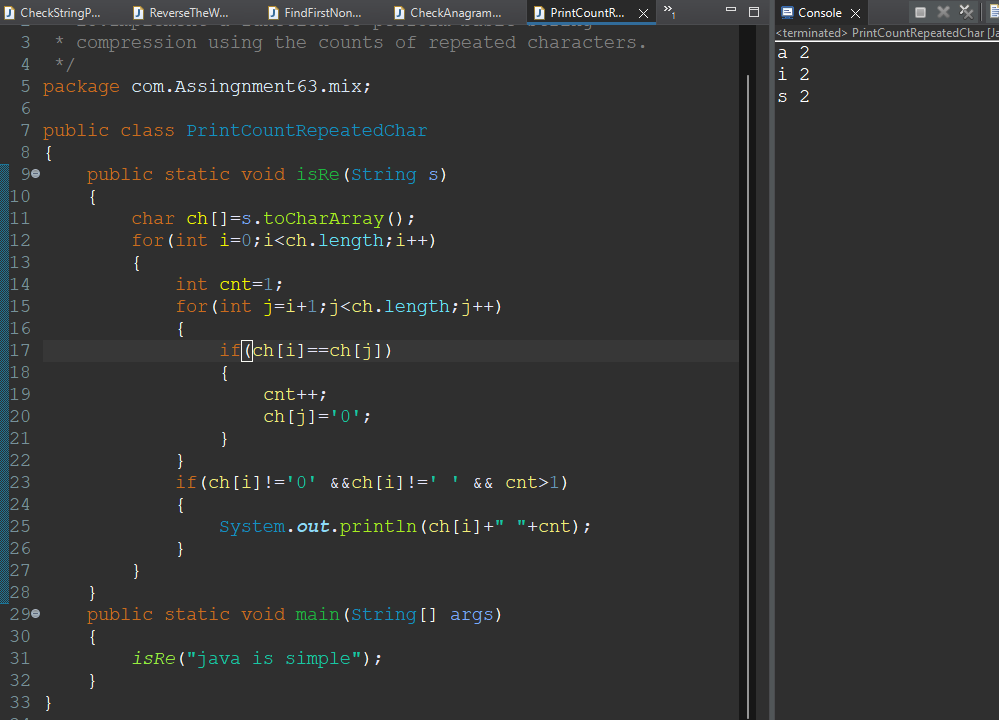
**23.Implement a function to find the first non-repeating character in a string.**

****

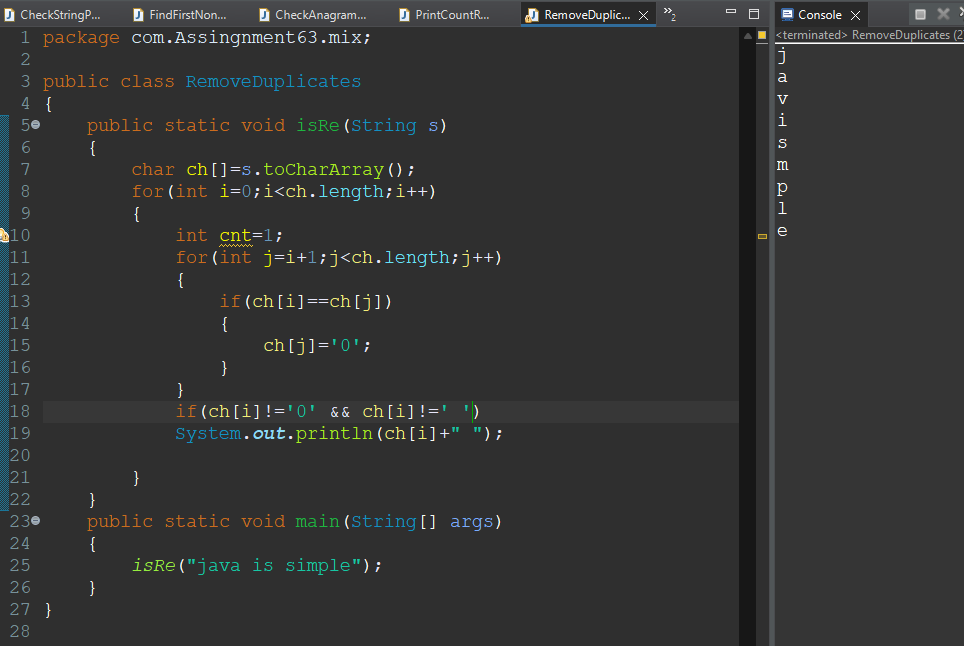
**24.Write a program to check if two strings are anagrams of each other.**

****

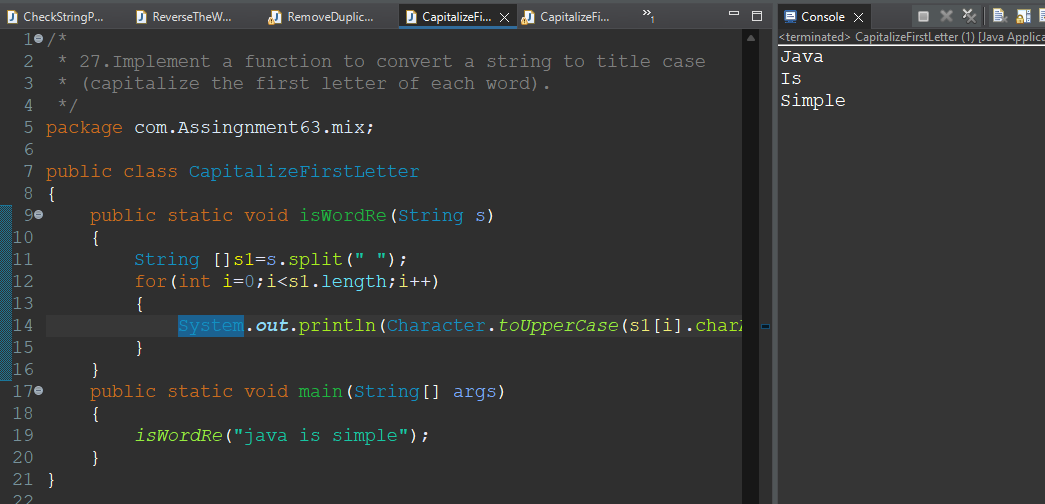
**25.Implement a function to perform basic string compression using the counts of repeated characters.**

****

**26.Write a program to remove duplicate characters from a string.**

****

**27.Implement a function to convert a string to title case (capitalize the first letter of each word).**

****

**28.Write a program to find the longest substring without repeating characters.**

/\*

\*

\*/

package com.Assingnment63.mix;

public class LongestSubstringWithoutRepeating

{

public static void issub(String d)

{

int max=0;

String temp=null;

for(int i=0;i<d.length();i++)

{

for(int j=i+1;j<d.length();j++)

{

String str = d.substring(i, j);

int cnt=1;

for(int k=0;k<str.length();k++)

{

for(int y=k+1;y<str.length();y++)

{

if(str.charAt(k)==str.charAt(y))

{

cnt++;

break;

}

}

}

if(cnt==1)

{

if(str.length()>max)

{

max=str.length();

temp=str;

}

}

}

}

System.***out***.println(temp);

}

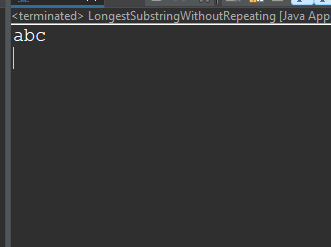
public static void main(String[] args)

{

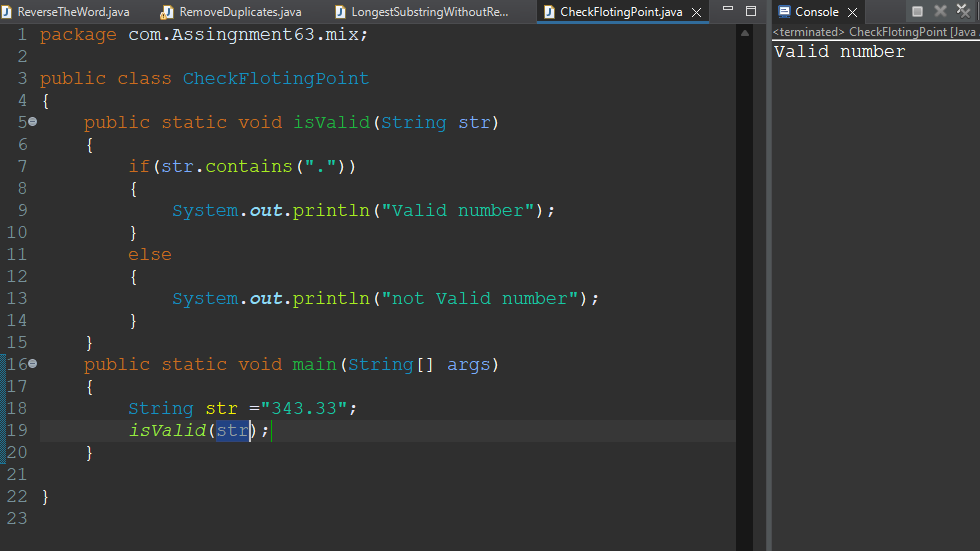
*issub*("abcabcabc");

}

}

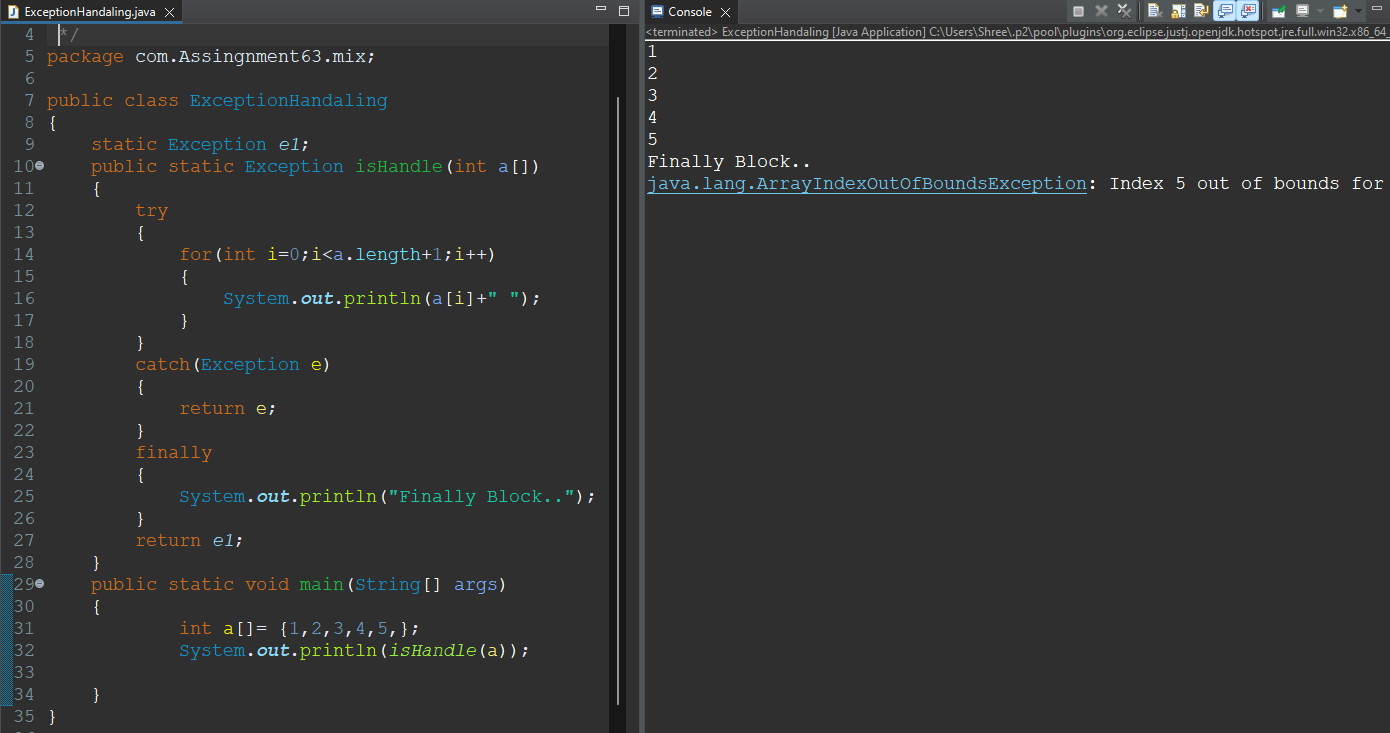
****

**29.Implement a function to check if a given string is a valid number (integer or floating-point).**

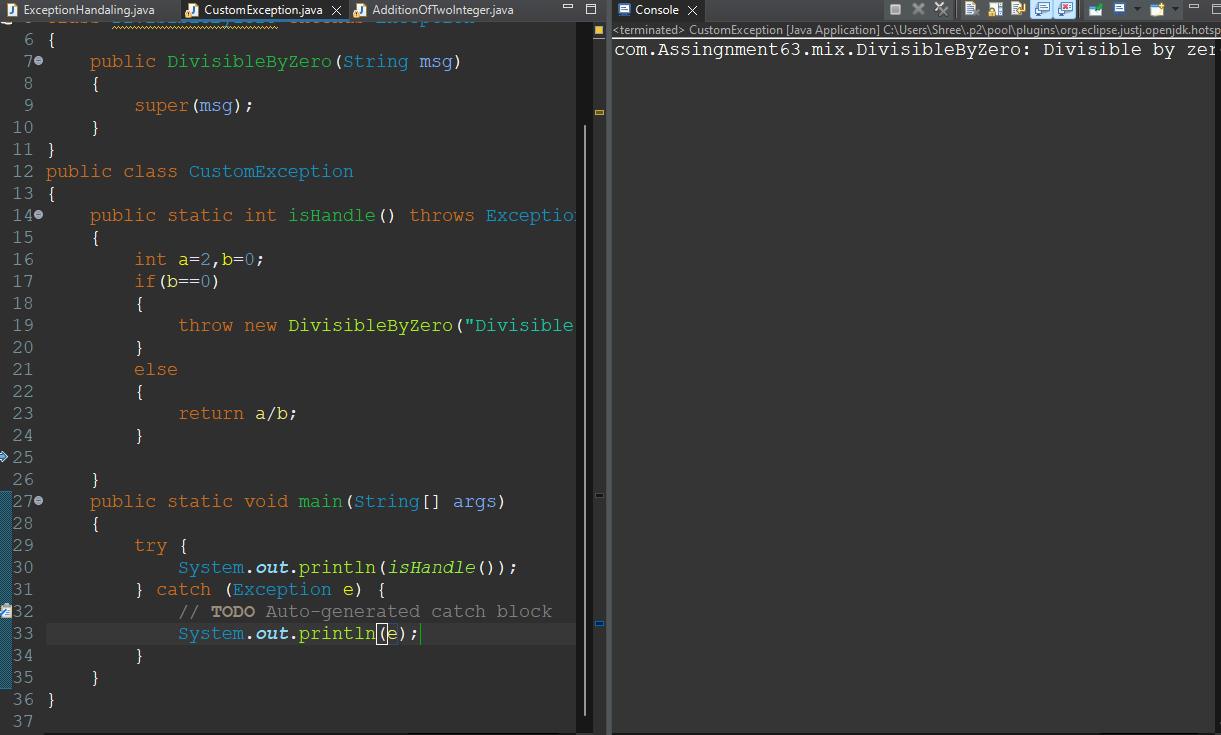
****

**Exception Handling**

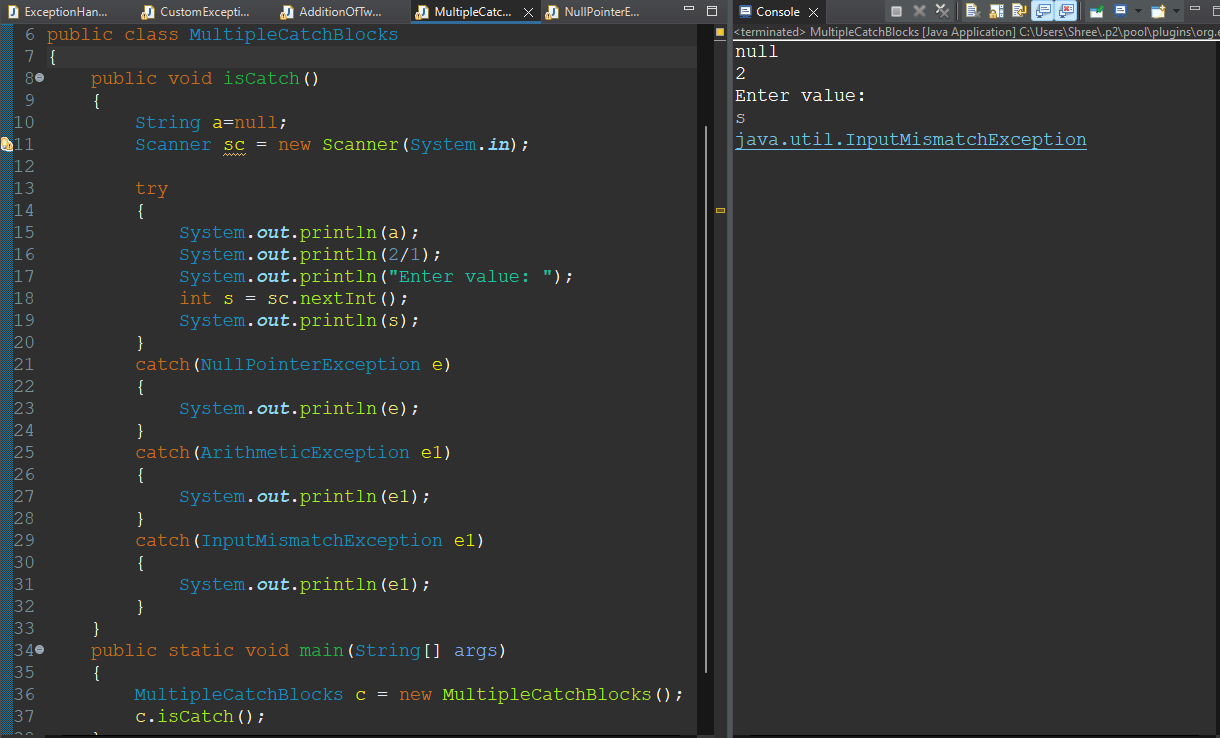
**30.Write a program to demonstrate the use of try, catch, and finally blocks.**

****

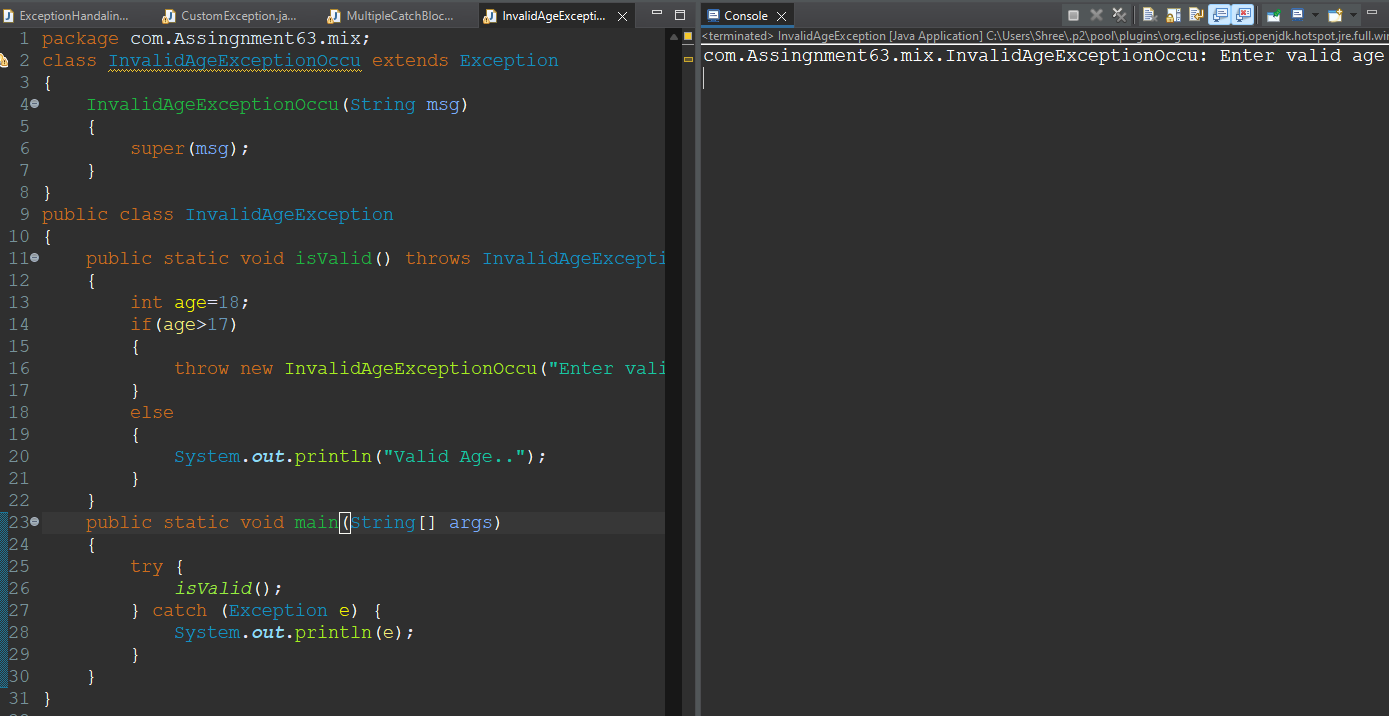
**31.Implement a function to handle division by zero and throw a custom exception if it occurs.**

****

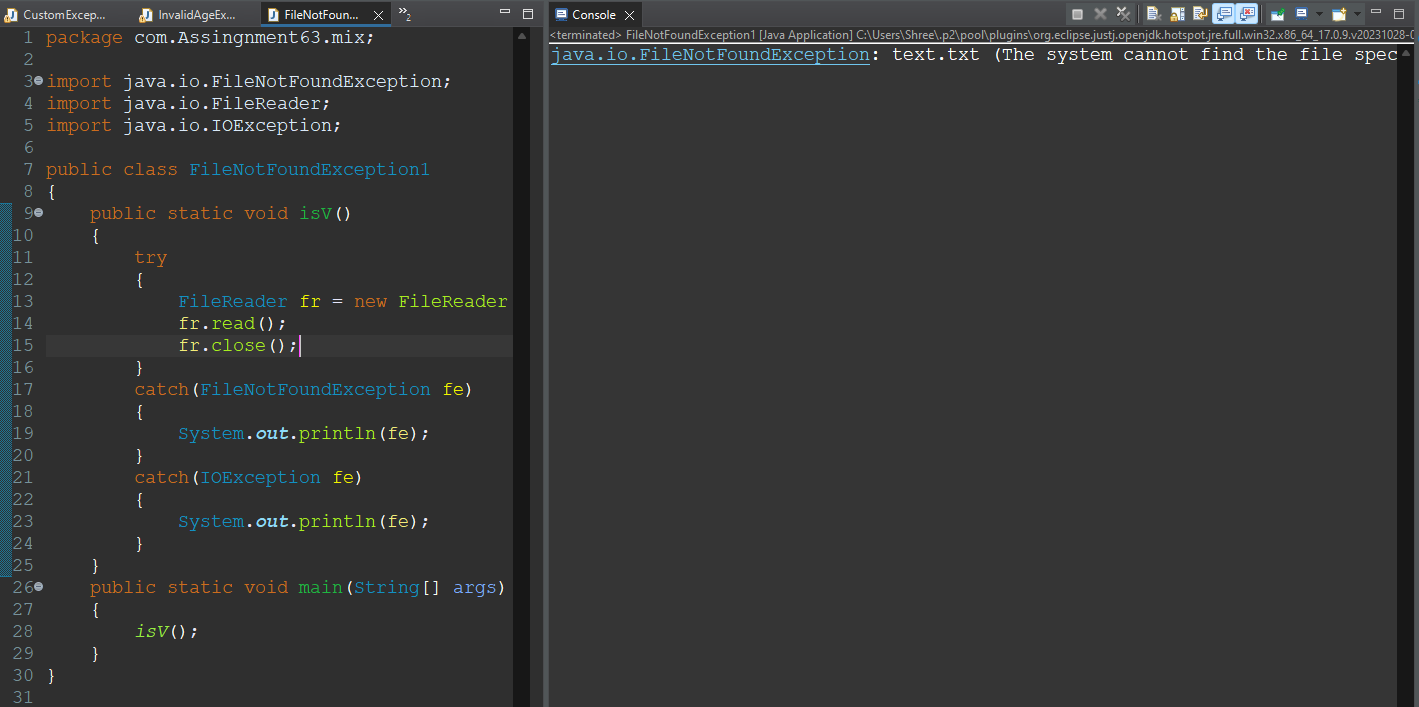
**32.Write a program to demonstrate multiple catch blocks for different types of exceptions.**

****

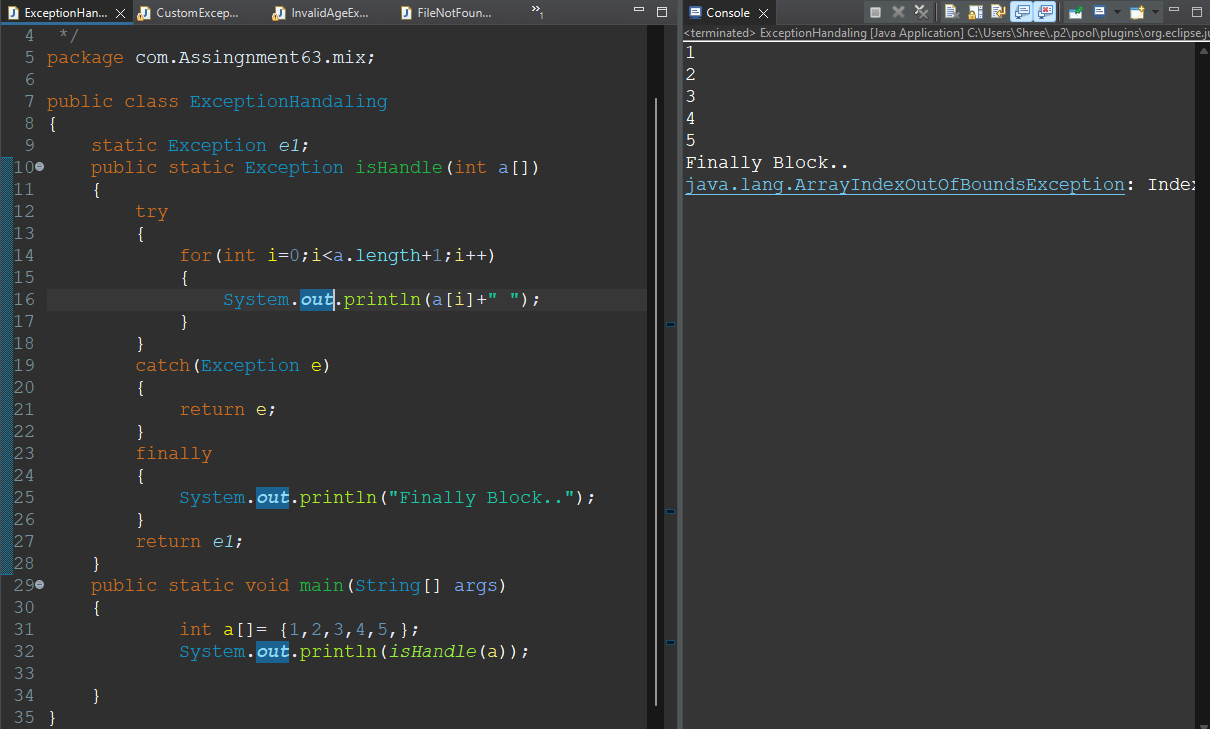
**33.Implement a function to check the validity of a user’s age and throw an InvalidAgeException if it’s not within a certain range.**

****

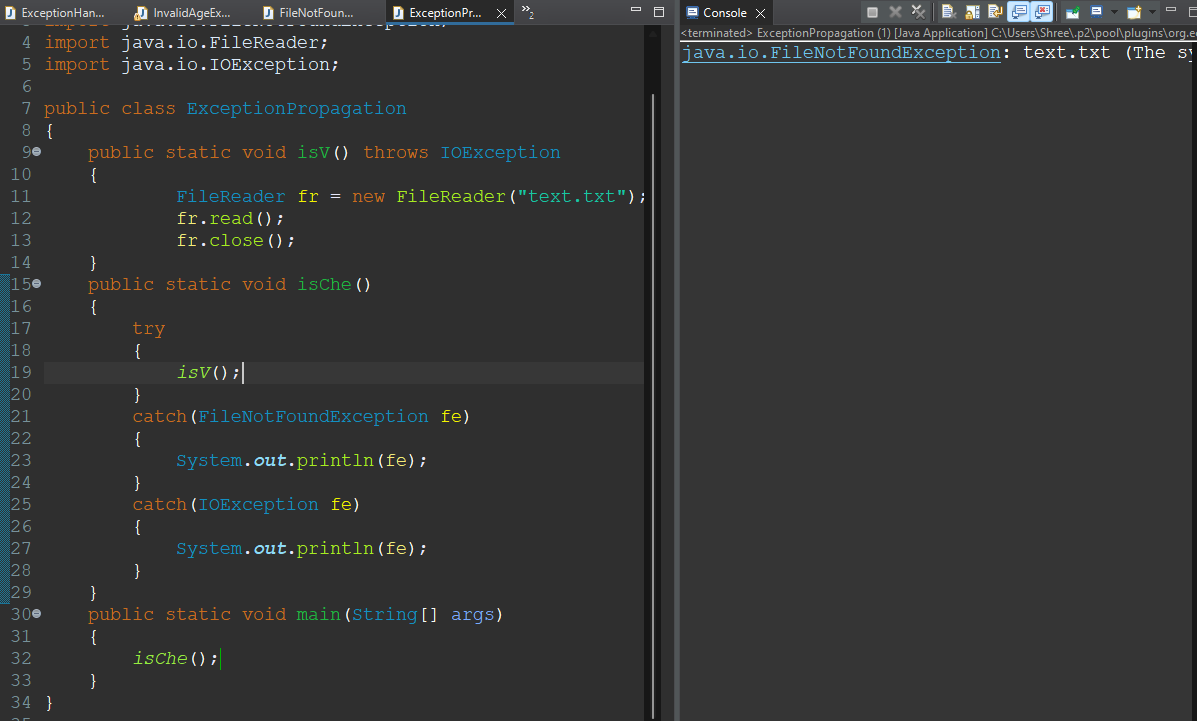
**34.Write a program to read from a file and handle FileNotFoundException and IOException.**

****

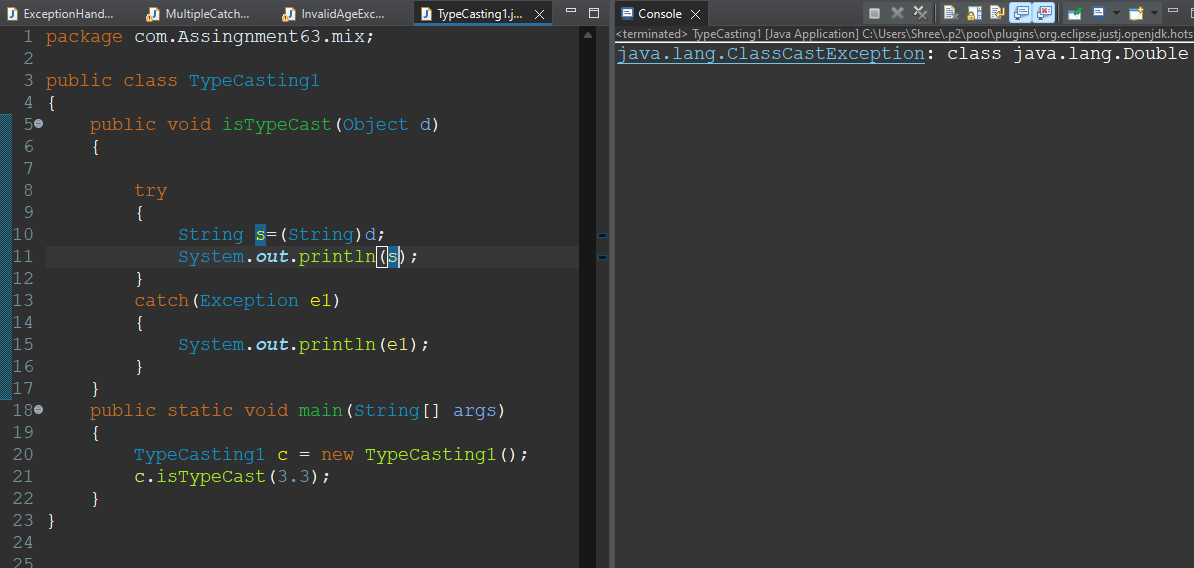
**35.Implement a function that takes an array index as input and handles ArrayIndexOutOfBoundsException.**

****

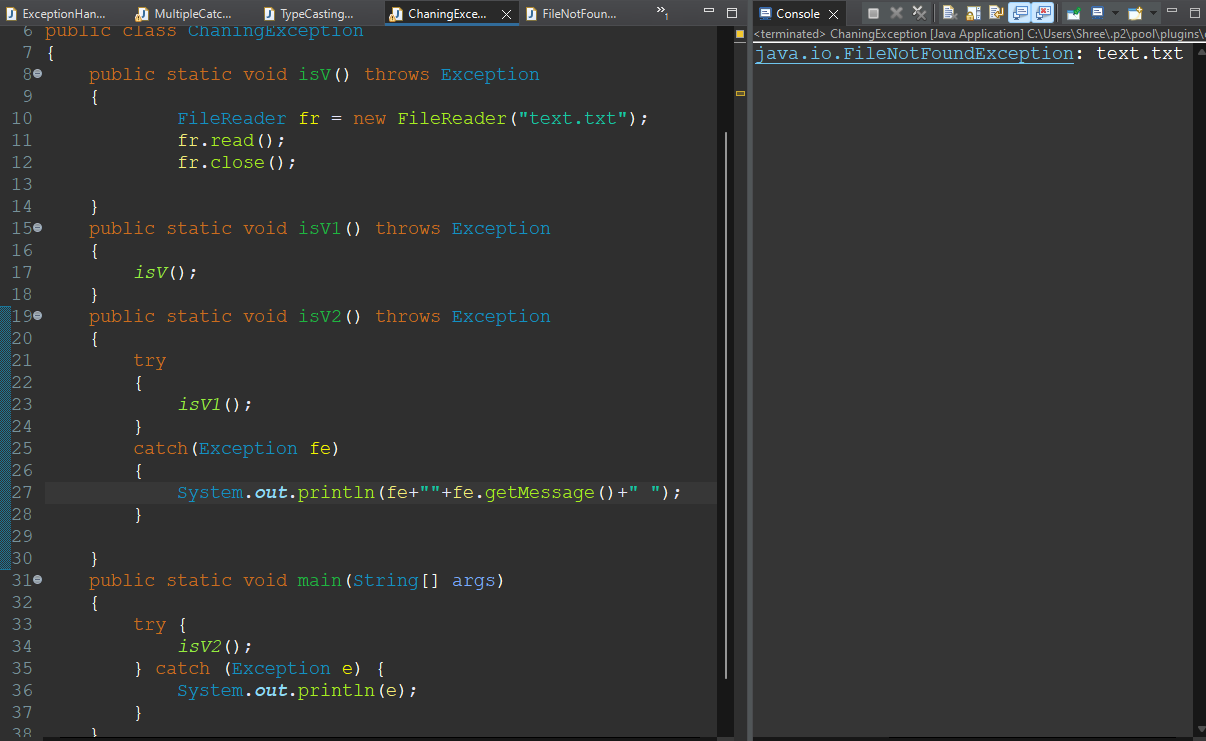
**36.Write a program to demonstrate the use of throws keyword and propagate an exception to the calling method.**

****

**37.Implement a function that performs typecasting and handles ClassCastException.**

****

**38.Write a program to demonstrate the chaining of exceptions.**

****

**39.Implement a function to simulate a simple banking system and throw an InsufficientFundsException if withdrawal exceeds balance.**

**/\***

**\* 11.Design a class BankAccount with methods to deposit, withdraw, and check balance.**

**\* Implement subclasses SavingsAccount and CheckingAccount.**

**\*/**

**package com.Assingnment63.mix;**

**import java.util.Scanner;**

**class InsufficientBalance extends Exception**

**{**

**InsufficientBalance(String msg)**

**{**

**super(msg);**

**}**

**}**

**class BankAccount**

**{**

**static Scanner sc;**

**static long depo=0;**

**static long withd=0;**

**static long bal=0;**

**static long cnt=0;**

**public static void deposite()**

**{**

**sc= new Scanner(System.in);**

**cnt++;**

**System.out.println("Enter Deposite Money: ");**

**depo = sc.nextInt();**

**System.out.println("Amount Deposited Suceessfully: "+depo);**

**System.out.println("--------------------<>-------------------");**

**}**

**public static void withdraw()throws InsufficientBalance**

**{**

**if(bal>0)**

**{**

**sc= new Scanner(System.in);**

**cnt=0;**

**System.out.println("Enter Withdrawal Money: ");**

**withd = sc.nextInt();**

**System.out.println("Amount Withdrawal Suceessfully: "+withd);**

**System.out.println("--------------------<>-------------------");**

**}**

**else**

**{**

**throw new InsufficientBalance("Insufficient Balance: "+bal);**

**}**

**}**

**public static void checkBalance()**

**{**

**if(cnt>=1)**

**{**

**long c=bal+=depo;**

**System.out.println("Your Balance is: "+(c));**

**}**

**else**

**{**

**long w=bal-=withd;**

**System.out.println("Your Balance is: "+(w));**

**}**

**}**

**}**

**public class CheckBalance {**

**public static void main(String[] args)**

**{**

**Scanner sc = new Scanner(System.in);**

**int ch =-1;**

**while(ch!=0)**

**{**

**System.out.println("Enter your choice\n1.Deposite\n2.Withdraw\n3.Check Balance\n4.Exit");**

**ch=sc.nextInt();**

**System.out.println("---------------<>-----------------");**

**switch(ch)**

**{**

**case 1: BankAccount.deposite();**

**break;**

**case 2: try {**

**BankAccount.withdraw();**

**} catch (InsufficientBalance e) {**

**// TODO Auto-generated catch block**

**System.out.println(e);**

**};**

**break;**

**case 3: BankAccount.checkBalance();;**

**break;**

**case 4: ch=0;**

**System.out.println("Processes Terminated.....!");**

**break;**

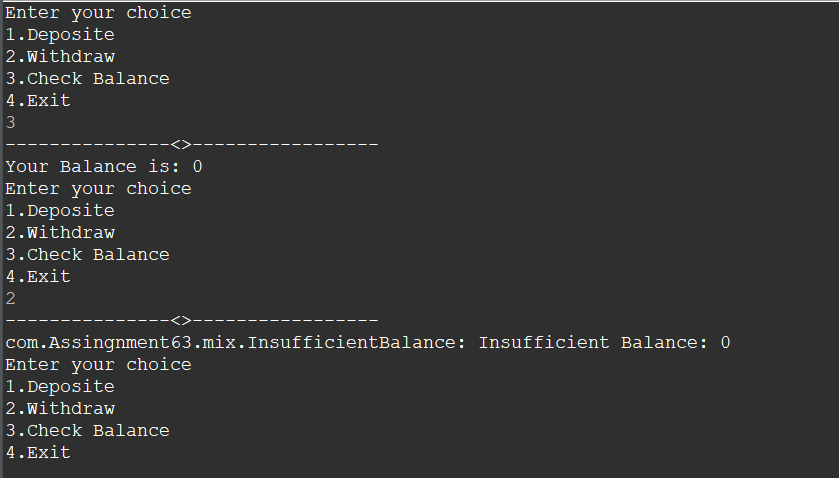
**}**

**}**

**System.out.println("-------------------<>------------------");**

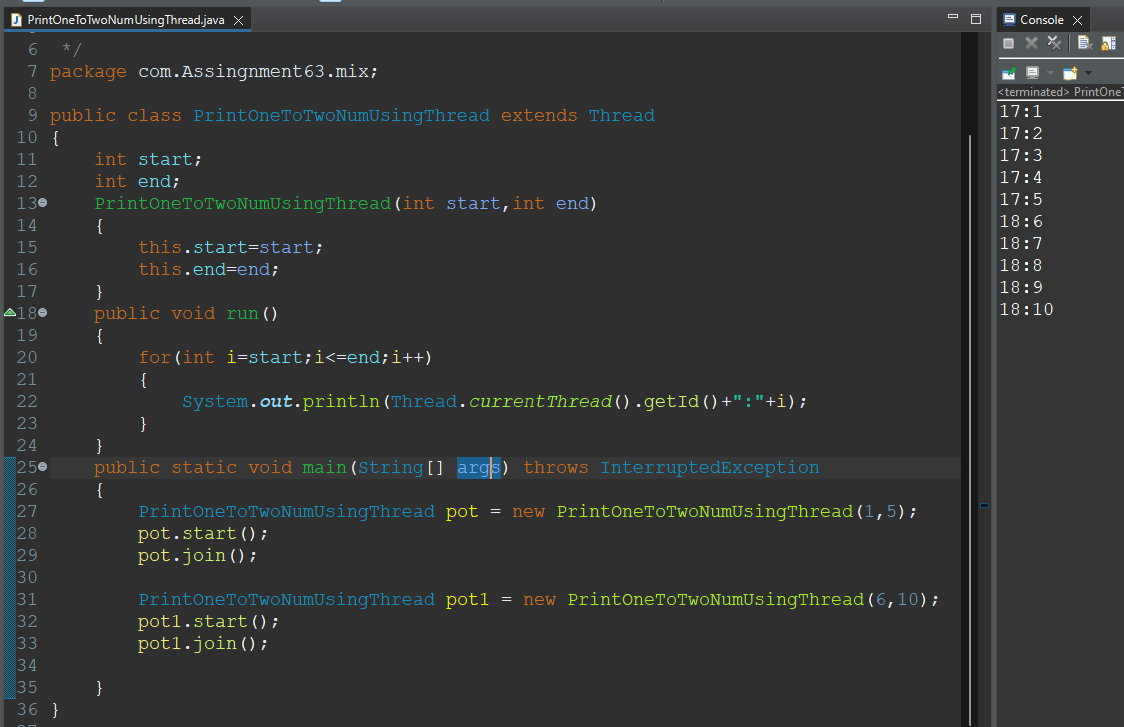
**}**

**}**

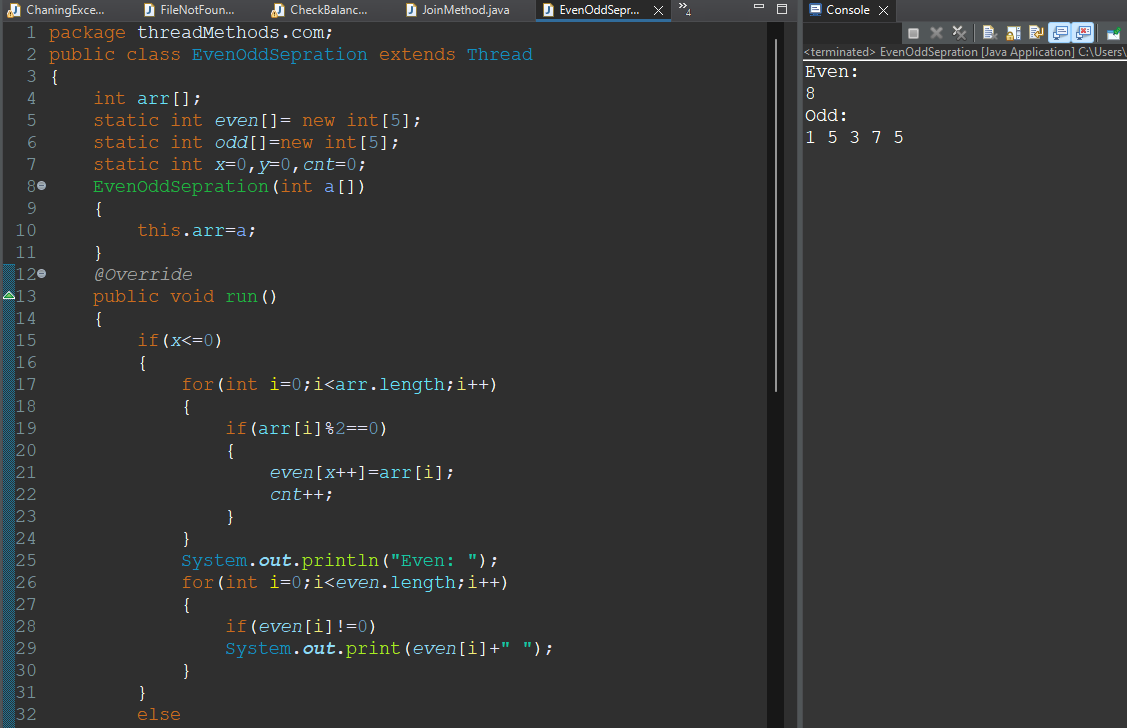
****

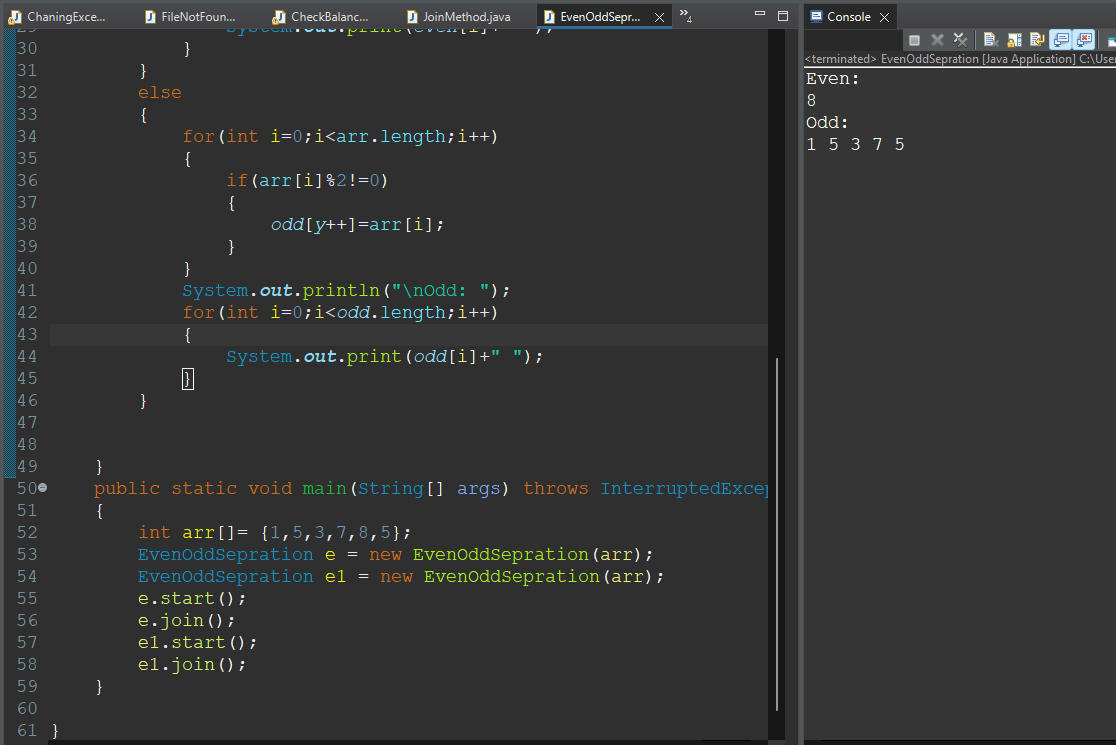
**Multithreading**

**40.Write a program to create and start two threads that print numbers from 1 to 10.**

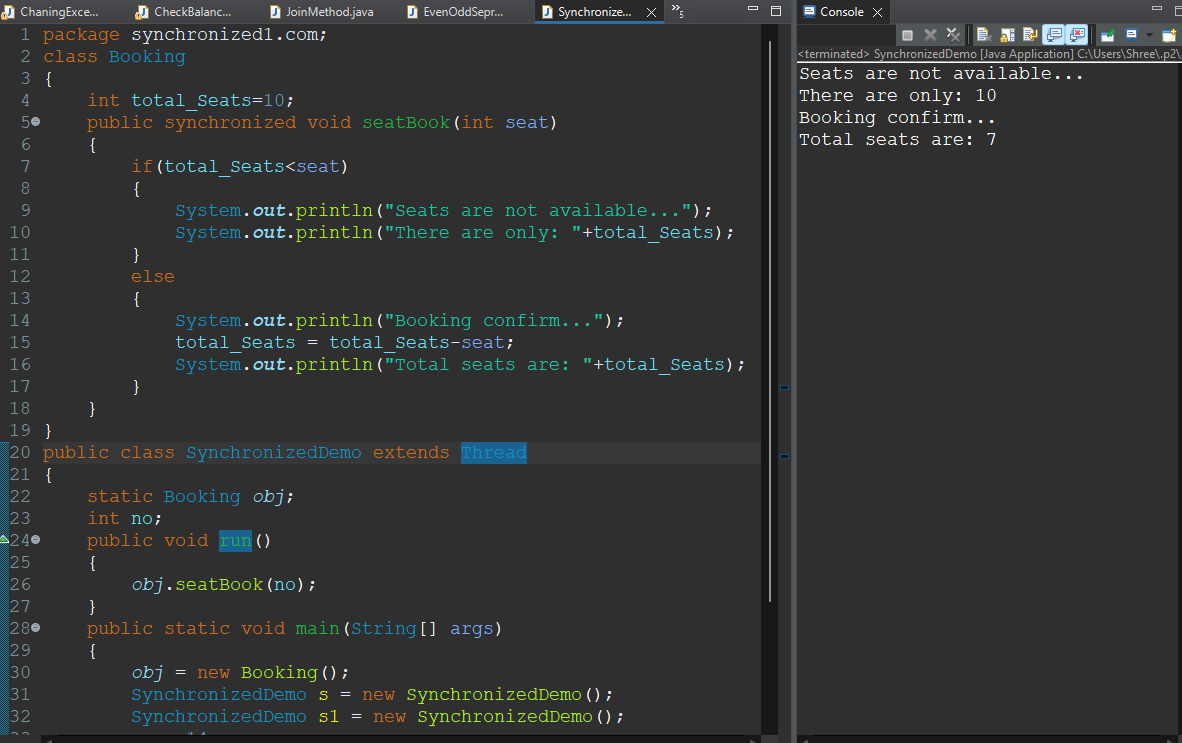
****

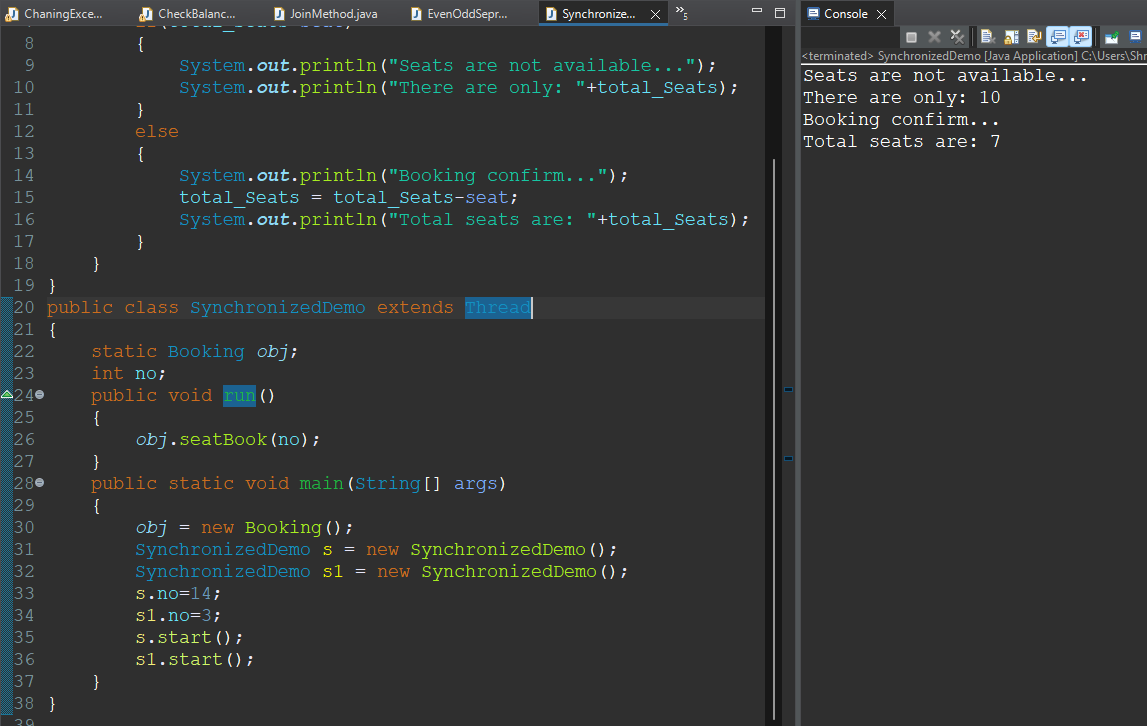
**41.Implement a program where one thread prints even numbers and another thread prints odd numbers.**

****

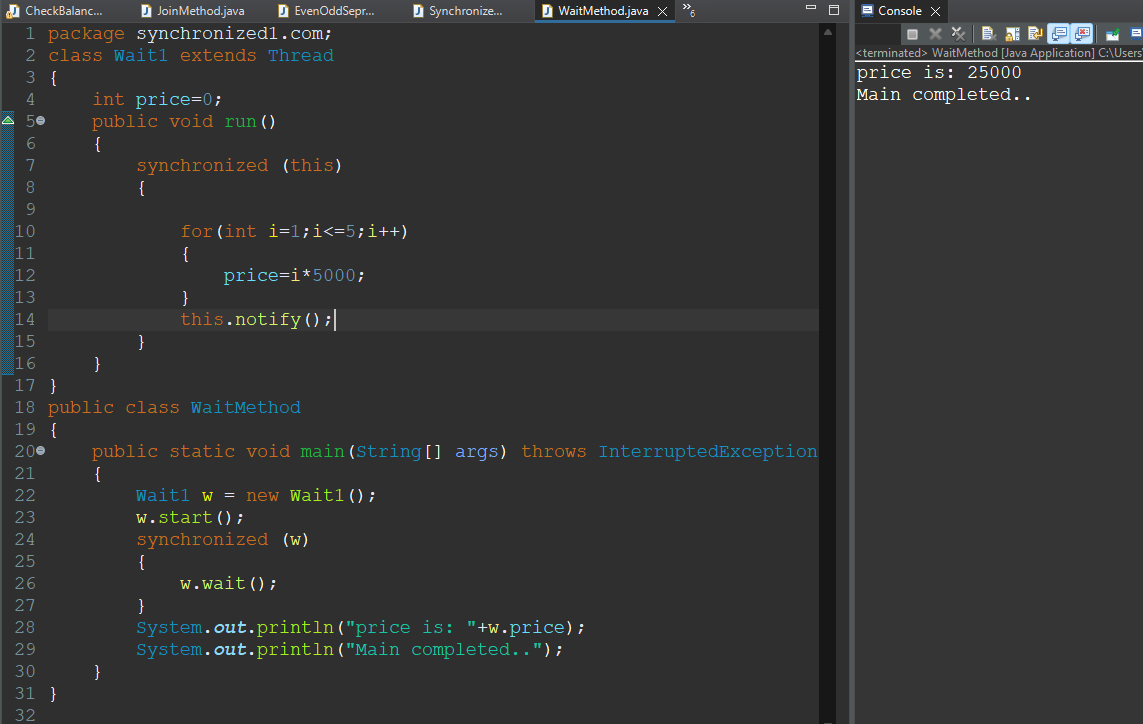
****

**42.Write a program to demonstrate the use of synchronized keyword for thread safety.**

****

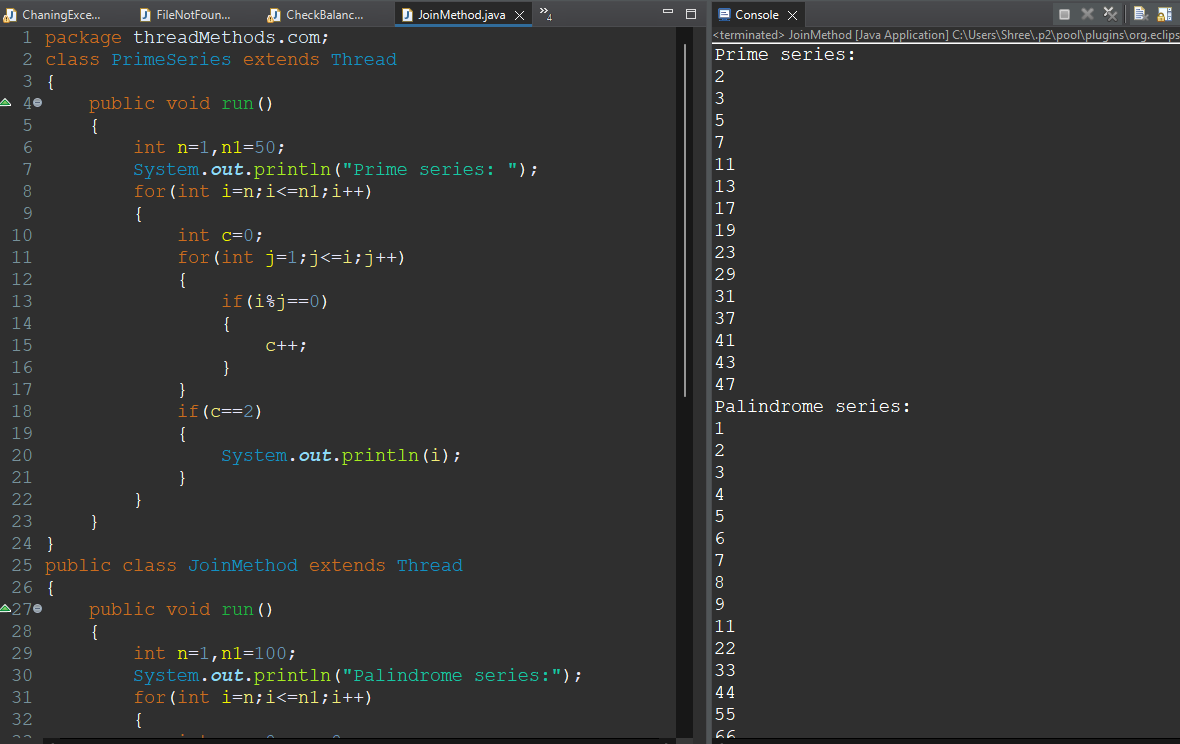
****

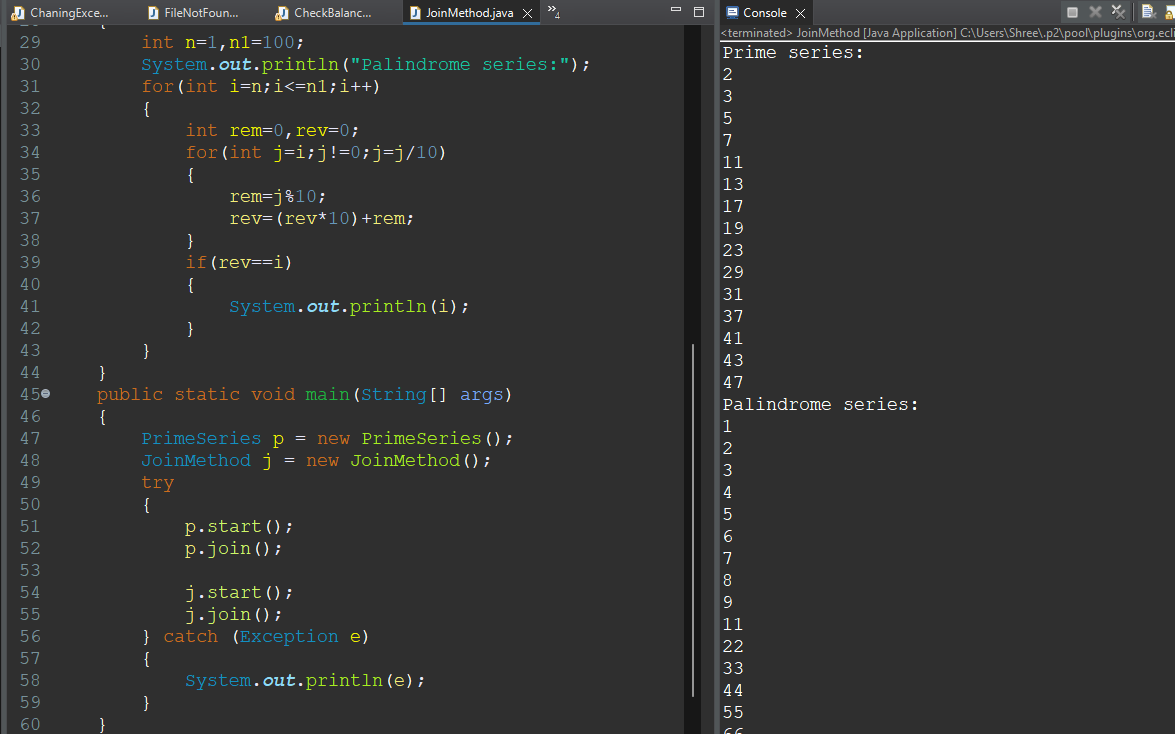
**43.Implement a producer-consumer problem using wait() and notify().**

****

**44.Write a program to demonstrate the use of Thread.sleep() and handle InterruptedException.**

**45. Implement a program to demonstrate the use of Thread.join() to wait for a thread to finish.**

****

****