**QUESTION BANK PROGRAMS**

**Definition 1 : Write a program to define abstract class, with two methods addition() and subtraction(). addition() is abstract method. Implement the abstract method and call that method using a program(s). (Abstract Class Concept)**

**Program:**

import java.util.Scanner;

abstract class Calculation

{

double num1,num2;

abstract double addition(double num1,double num2);

double subtraction(double num1,double num2)

{

return num1-num2;

}

}

class Add extends Calculation

{

public double addition(double n1,double n2)

{

return n1+n2;

}

}

class Abstract\_class

{

public static void main(String []args)

{

Calculation c1; // reference of abstract class

c1=new Add(); // object of an Add class which //extends abstract class Calculation

Scanner s=new Scanner(System.in);

System.out.println("Enter two numbers:");

c1.num1=s.nextDouble();

c1.num2=s.nextDouble();

double sum=c1.addition(c1.num1,c1.num2);

System.out.println("Addition is :" + sum);

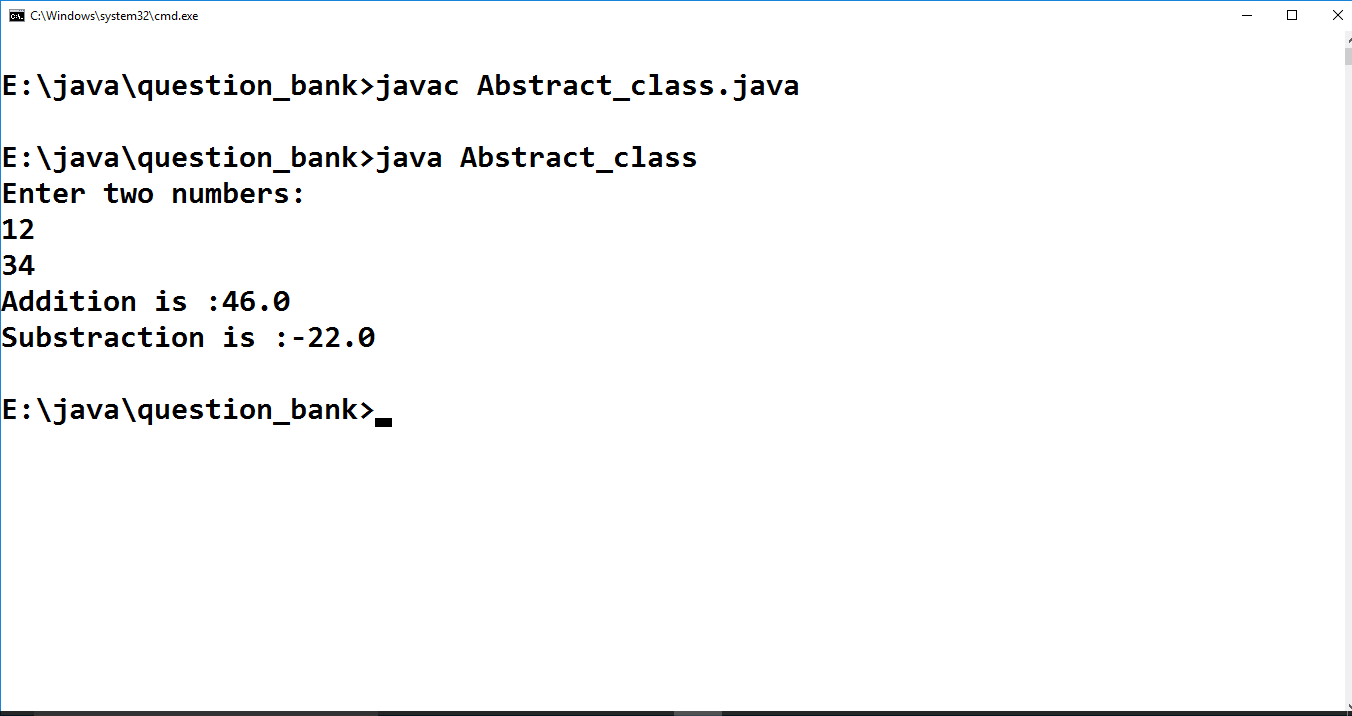
double sub=c1.subtraction(c1.num1,c1.num2);

System.out.println( "Substraction is :" + sub);

}

}

**OUTPUT:**



**Definition 2 :** **Write a program that divides two numbers. Handle all exceptions that can be generated in this program. (Exception Handling)**

**Program:**

import java.util.\*;

import java.io.\*;

class Exception\_Handling

{

public static void main(String []args)

{

float i,j,k=0;

i=45;

Scanner s=new Scanner(System.in);

try

{

System.out.println("Enter a number :");

j=s.nextFloat();

k=i/j;

System.out.println("Division is :" +k);

}

catch(ArithmeticException ae)

{

System.out.println("Arithmatic Exception" +ae);

}

catch(NumberFormatException ne)

{

System.out.println("Number Format Exception" +ne);

}

catch(InputMismatchException ime)

{

System.out.println("Input Mismatch Exception" +ime);

}

finally // this block execute atleast once // optional no need to implement

{

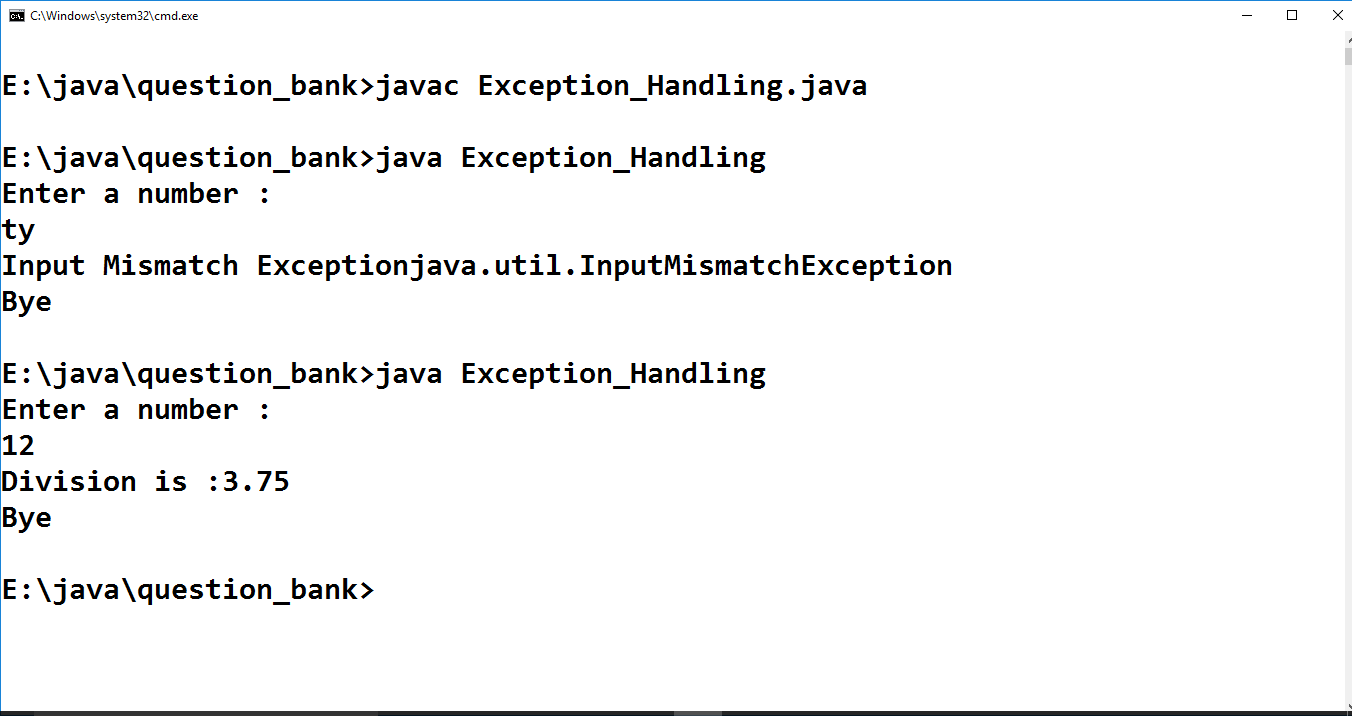
System.out.println("Bye");

}

}

}

**OUTPUT:**



**Definition 4: Define time class with hour and minute. Also define addition method to add two time objects. (Class and Object Concept)**

**Program:**

import java.util.Scanner;

class Time

{

int hour,minute;

Time(int hour,int minute)

{

this.hour=hour;

this.minute=minute;

}

void addition(Time obj1,Time obj2)

{

int min=obj1.minute+obj2.minute;

int hr=obj1.hour+obj2.hour;

if(min>=60)

{

hr=hr+(min/60);

min=min%60;

}

System.out.println( hr + " Hours " + " : " + min + " Minutes") ;

}

}

class Time\_Hour

{

public static void main(String []args)

{

int h,m;

Scanner s=new Scanner(System.in);

System.out.println("Enter hour and minute for object 1: ");

h=s.nextInt();

m=s.nextInt();

Time t1=new Time(h,m);

System.out.println("Enter hour and minute for object 2: ");

h=s.nextInt();

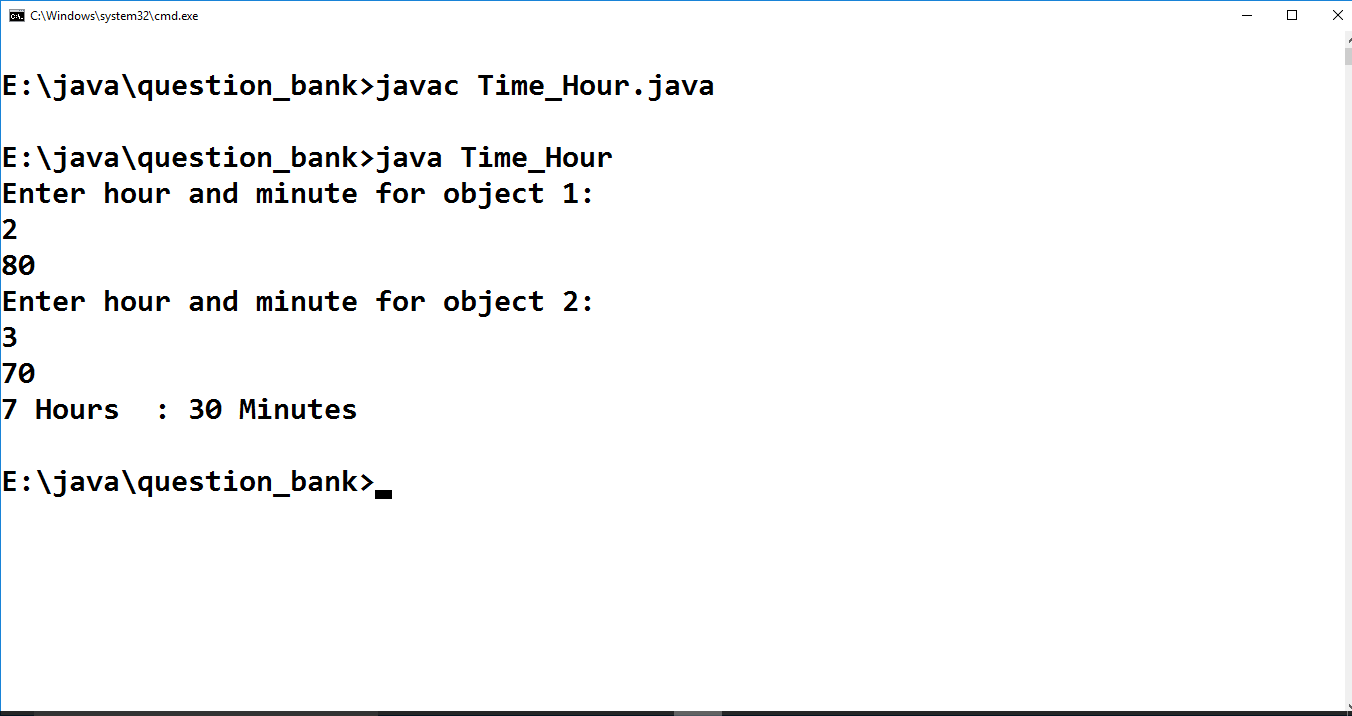
m=s.nextInt();

Time t2=new Time(h,m);

t1.addition(t1,t2);

}}

**OUTPUT:**



**Definition 5: Write an application that takes input from command-line argument. If an argument is found that does not begin with an upper case letter,display error message and terminate. (Command line argument and String class Concept)**

**Program:**

class Command\_Line

{

public static void main(String []args)

{

String str;

if(Character.isUpperCase(args[0].charAt(0)))

{

System.out.println("No error: " +args[0]);

}

else

{

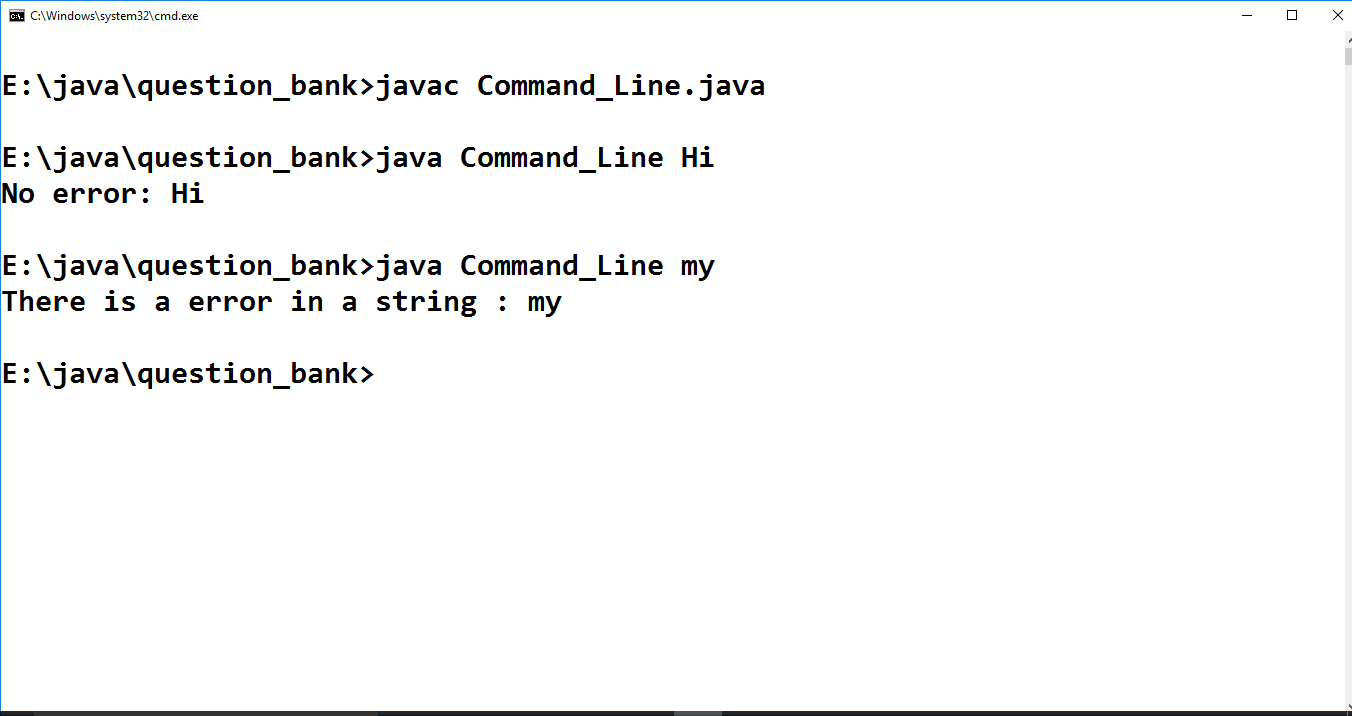
System.out.println("There is a error in a string : " +args[0]);

}

}

}

**OUTPUT:**



**Definition 6: Write a program to demonstrate the multipath inheritance for the classes having relations as shown in figure. (Interface Inheritance Concept)**

**Program:**

interface A

{

public void display\_A();

}

interface B extends A

{

public void display\_B();

}

interface C extends A

{

public void display\_C();

}

class D implements B,C

{

public void display\_A()

{

System.out.println("This is a class A which is a super class");

}

public void display\_B()

{

System.out.println("This is a class B which implements interface A");

}

public void display\_C()

{

System.out.println("This is a class C which implements interface A");

}

public void display\_D()

{

System.out.println("This is a class D which implements interface B and C");

}

}

class Multipath\_Inheritence

{

public static void main(String []args)

{

D d1=new D();

d1.display\_A();

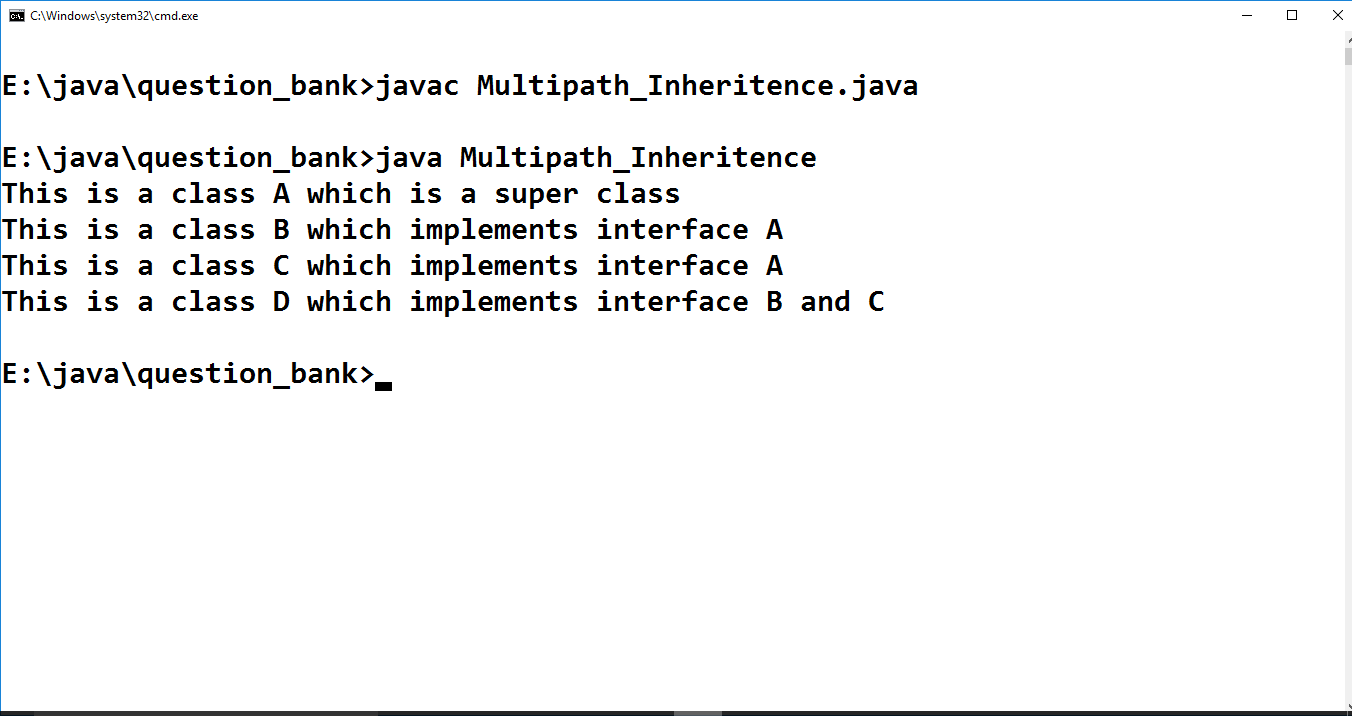
d1.display\_B();

d1.display\_C();

d1.display\_D();

} }

**OUTPUT:**



**Definition 7: Design a class named Fan to represent a fan. The class contains:**

**- Three constants named SLOW, MEDIUM and FAST with values 1,2 and 3 to denote the fan speed.**

**- An int data field named speed that specifies the speed of the fan (default SLOW).**

**- A boolean data field named f\_on that specifies whether the fan is on(default false).**

**- A double data field named radius that specifies the radius of the fan (default 4).**

**- A data field named color that specifies the color of the fan (default blue).**

**- A no-arg constructor that creates a default fan.**

**- A parameterized constructor initializes the fan objects to given values.**

**- A method named display() will display description for the fan. If the fan is on, the display() method displays speed, color and radius.**

**If the fan is not on, the method returns fan color and radius along with the message fan is off.**

**-Write a test program that creates two Fan objects. One with default values and the other with medium speed, radius 6, color brown, and turned on status true.**

**-Display the descriptions for two created Fan objects. (Class, Object, conctructor and Method concepts)**

**Program:**

import java.util.Scanner;

class Fan

{

int SLOW=1,MEDIUM=2,FAST=3;

int speed;

boolean f\_on;

double radious;

String color;

Fan()

{

this.speed=SLOW;

this.f\_on=false;

this.radious=4;

this.color="blue";

}

Fan(int speed,boolean f\_on,double radious,String color)

{

this.speed=speed;

this.f\_on=f\_on;

this.radious=radious;

this.color=color;

}

String display(String str)

{

return "speed : " + speed +"\n"+ "radious :" + radious +"\n"+ "color :" + color +"\n"+str;

}

}

class Fan\_Demo

{

public static void main(String []args)

{

Fan f1=new Fan();

Fan f2;

Scanner s=new Scanner(System.in);

System.out.println("Enter weather the fan is on or off: " + "true=on" + " false=off");

f1.f\_on=s.nextBoolean();

if(f1.f\_on==true)

{

f2=new Fan(2,f1.f\_on,6,"brown");

System.out.println( "For object 1: " + f1.display("Fan is on"));

System.out.println( "\n\nFor object 2: " + f2.display("Fan is on"));

}

else

{

f1.speed=0;

f2=new Fan(0,f1.f\_on,6,"brown");

System.out.println("For object 1 :" + f1.display("Fan is off"));

System.out.println("\n\nFor object 2 :" + f2.display("Fan is off"));

}

}

}

**OUTPUT:**

