**Assignment - 1**

1. **Create a class car that has Model no, name , color as data member and create void getCarDetails( ) and void showCarDetails( ) as member method. Read details for 3 cars and display it.**

import java.util.Scanner;

class Car{

int model\_no;

String name, color;

Scanner inp = new Scanner(System.in);

public void getCarDetials(){

System.out.print("\nEnter Model Number:- ");

model\_no = inp.nextInt();

System.out.print("Enter Name:- ");

inp.nextLine(); // This is just to catch the \n from previous input value

name = inp.nextLine();

System.out.print("Enter Color:- "); // no Need of black inp.nextLine() as upper value takes last \n char

name = inp.nextLine();

System.out.println("\nDetails Stored Successfully!!!!");

}

public void showCarDetials(){

System.out.println("\nModel No:- " + model\_no);

System.out.println("Name:- " + name);

System.out.println("Color:- " + color);

}

}

class MainClass{

public static void pattern(int limit, char ch){

System.out.println();

for(int i = 0; i < limit; i++) System.out.print(ch);

System.out.println();

}

public static void main(String args[]){

int i;

int LIMIT = 30;

System.out.print("Enter Car details");

Car cars[] = new Car[3];

pattern(LIMIT, '=');

for(i = 0; i < 3; i++){

System.out.println("Enter info of car " + (i+1));

cars[i] = new Car();

cars[i].getCarDetials();

}

System.out.println("\nPrinting Car Details");

pattern(LIMIT, '=');

for(i = 0; i < 3; i++){

System.out.println("Detial for car No:- " + (i+1));

cars[i].showCarDetials();

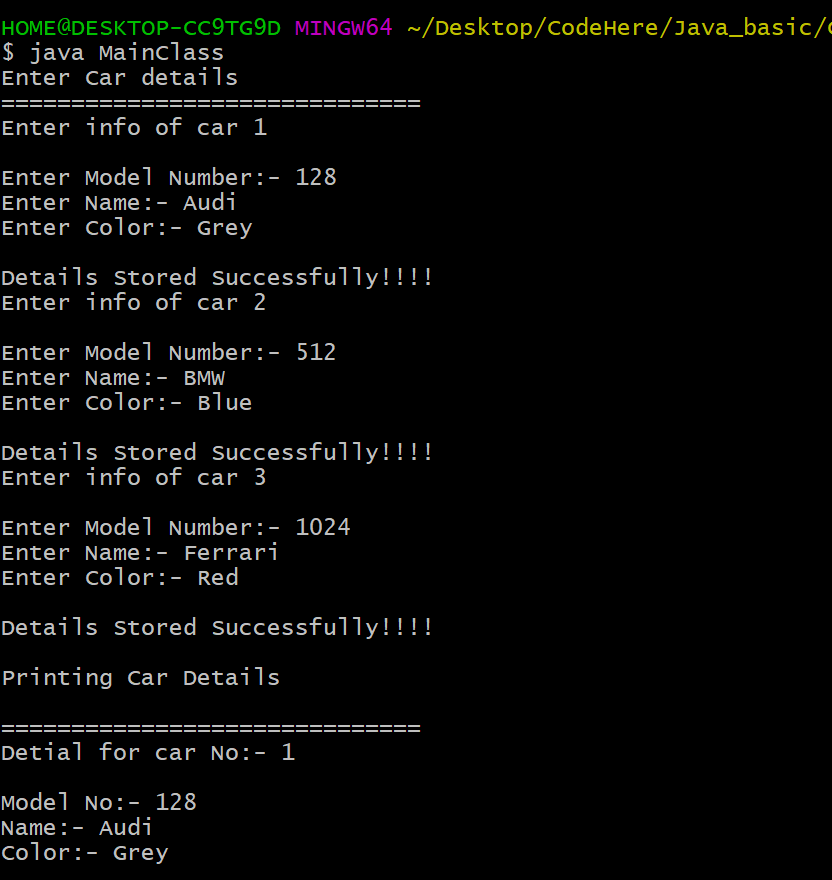
pattern(LIMIT, '-');

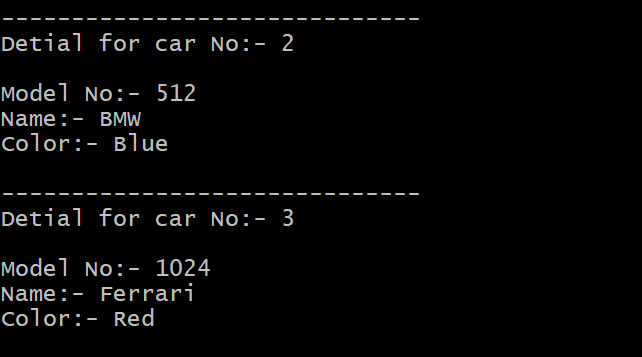
}

}

}

Output:





|  |
| --- |
| 1. **Create a class Time that has hour, minute and second as data members. Create a parameterized constructor to initialize Time objects. Create a member function to sum two time objects.(for example Time Sum(Time obj1,Time obj2) or Time Sum(Time obj1) ).** |

class Time{

int hh, mm, ss;

Time(int d\_h, int d\_m, int d\_s){

hh = d\_h;

mm = d\_m;

ss = d\_s;

setTime();

}

private void setTime(){

hh += (mm/24);

mm = (mm%24);

mm += (ss/60);

ss = (ss % 60);

}

public Time sum(Time obj){

Time new\_time = new Time(hh+obj.hh, mm+obj.mm, ss+obj.ss);

return new\_time;

}

public void showTime(String name){

System.out.println("\nTime of Obj" + name + " :- " + hh + ":" + mm + ":" + ss);

} }

class MainClass{

public static void main(String args[]){

Time obj1 = new Time(10, 50, 30);

Time obj2 = new Time(5, 70, 50);

Time obj3 = obj1.sum(obj2);

obj1.showTime("1");

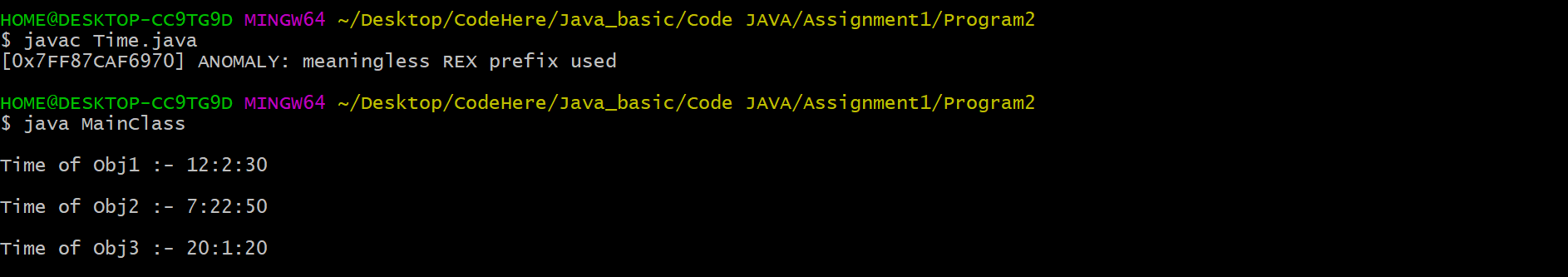
obj2.showTime("2");

obj3.showTime("3");

}

}

Output:



|  |
| --- |
| 1. **Write a program to implement singly linklist with member methods append, Insert, display, search and delete.** |

import java.util.\*;

class nodell{

int data;

nodell next;

nodell(){

data = -1;

next = null;

}

nodell(int d\_data){

data = d\_data;

next = null;

}

void append(nodell new\_node){

nodell temp = this;

while(temp.next != null) temp = temp.next;

temp.next = new\_node;

}

void insert(int d\_data, nodell new\_node){

nodell temp = this;

while(temp.data != d\_data) temp = temp.next;

new\_node.next = temp.next;

temp.next = new\_node;

}

int delete(int d\_data){

if(this.next == null){

int val = this.data;

this.data = -1;

return val;

}

else{

nodell prev\_node = this, next\_node = this.next;

while(next\_node.data != d\_data){

prev\_node = next\_node;

next\_node = next\_node.next;

}

prev\_node.next = next\_node.next;

return next\_node.data;

}

}

public int search(int val){

nodell temp = this;

int cnt = 0;

while(temp.next != null && temp.data != val) {

temp = temp.next;

cnt += 1;

}

if(temp.next == null) return -1;

else return cnt;

}

void display(){

nodell temp = this;

while(temp != null){

System.out.println("Value = " + temp.data);

temp = temp.next;

}

}

}

class LinkedList{

static void print(String str){

System.out.println(str);

}

public static void main(String arg[]){

Scanner inp = new Scanner(System.in);

print("Enter '-1' When done !!");

print("\nEnter values:- ");

nodell obj = new nodell(inp.nextInt());

while(true){

nodell new\_obj = new nodell(inp.nextInt());

if(new\_obj.data == -1) break;

obj.append(new\_obj);

}

print("\nDisplaying Elements");

obj.display();

nodell temp = new nodell(15);

obj.insert(4, temp);

print("\nDisplaying Elements after insertion at Nth posisition");

obj.display();

obj.delete(15);

print("\nDisplaying Elements after Deletion at Nth posisition");

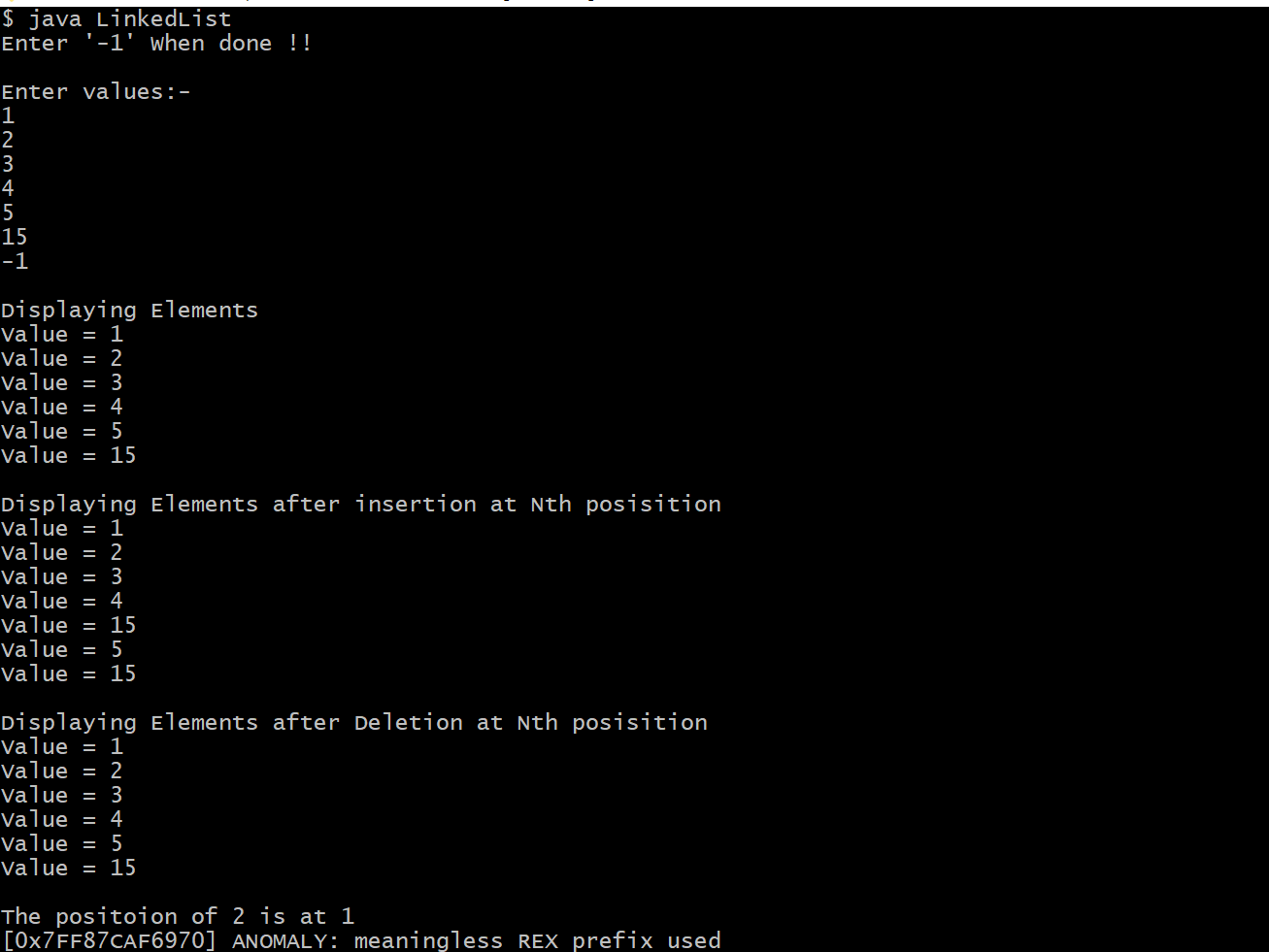
obj.display();

System.out.println("\nThe positoion of 2 is at " + obj.search(2));

}

}

Output:



1. **Write a program to create an abstractclass Shape havinginstance variables dimension1 in double and color and two member methods:void display(): displays the color of the shape defined.abstract void area(): returns the area for given shape.Inherit class Triangle, Square, rectangle and circle. Declare the other dimension as dim2 if needed for particular class.**

import java.util.Scanner;

abstract class Shape{

double dim1, dim2;

String color;

Shape(int d\_dim1, int d\_dim2, String d\_clr){

dim1 = d\_dim1;

dim2 = d\_dim2;

color = d\_clr;

}

Shape(int d\_dim1, String d\_clr){

dim1 = d\_dim1;

dim2 = 0;

color = d\_clr;

}

public void display(){

System.out.print("\nDisplaying Property of ");

}

public abstract void area();

}

class Triangle extends Shape{

Triangle(int dim1, int dim2, String clr){

super(dim1, dim2, clr);

}

public void display(){

super.display();

System.out.println("Triangle");

System.out.println("Length = " + dim1);

System.out.println("Height = " + dim2);

System.out.println("Color = " + color);

area();

}

public void area(){

System.out.println("Area of Triangle = " + (0.5\*dim1\*dim2));

}

}

class Square extends Shape{

Square(int dim1, int dim2, String clr){

super(dim1, dim2, clr);

}

public void display(){

super.display();

System.out.println("Square");

System.out.println("Length = " + dim1);

System.out.println("Breadth = " + dim2);

System.out.println("Color = " + color);

area();

}

public void area(){

System.out.println("Area of Square = " + (dim1\*dim2));

}

}

class Rectangle extends Shape{

Rectangle(int dim1, int dim2, String clr){

super(dim1, dim2, clr);

}

public void display(){

super.display();

System.out.println("Rectangle");

System.out.println("Length = " + dim1);

System.out.println("Breadth = " + dim2);

System.out.println("Color = " + color);

area();

}

public void area(){

System.out.println("Area of Rectangle = " + (2\*(dim1+dim2)));

}

}

class Circle extends Shape{

Circle(int dim1, String clr){

super(dim1, clr);

}

public void display(){

super.display();

System.out.println("Circle");

System.out.println("Radius = " + dim1);

System.out.println("Color = " + color);

area();

}

public void area(){

System.out.println("Area of Circle = " + (2\*3.14\*dim1\*dim1));

}

}

class MainCaller{

public static void main(String arg[]){

Triangle triangle = new Triangle(5, 4, "Red");

Rectangle rectangle = new Rectangle(4, 6, "Orange");

Square square = new Square(5, 5, "Black");

Circle circle = new Circle(7, "Cyan");

triangle.display();

rectangle.display();

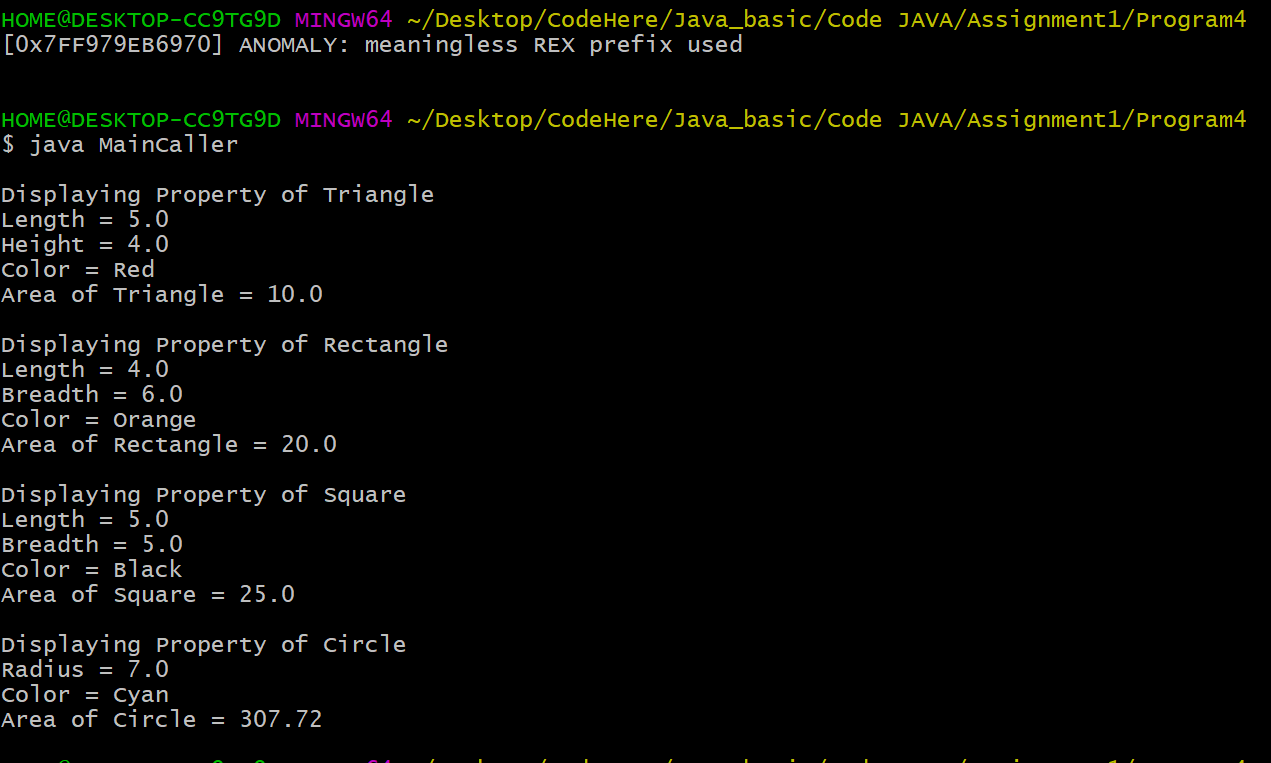
square.display();

circle.display();

}

}

Output:



|  |
| --- |
| 1. **Write a program for creating a multithreaded program with following conditions:** 2. **a. length of stack is 15 elements** 3. **b. Thread named StackInsert pushes integer data on the stack** 4. **c. Thread named StackDelete pops integer data from the stack.**   class Q{  int[] stack = new int[15];  int last\_pro = 0;  int last\_con = 0;  int stack\_len = 0;  synchronized void pop(){  if(stack\_len == 0){  try{ wait(); }  catch(InterruptedException e){ System.out.println("\nConsumer Interrupted!!!"); }  }  System.out.print("\nConsuming:- ");  while(stack\_len != 0){  stack[last\_con] = 0;  last\_con += 1;  System.out.print(last\_con + "\t");    last\_con %= stack.length;  stack\_len -= 1;  }  notify();  }  synchronized void push(){  if(stack\_len == stack.length){  try{ wait(); }  catch(InterruptedException e){ System.out.println("\nProducer Interrupted!!!"); }  }  System.out.print("\nProducing-> ");  while(stack\_len != stack.length){  stack[last\_pro] = ++last\_pro;  System.out.print(last\_pro + "\t");    last\_pro %= stack.length;  stack\_len += 1;  }  notify();  }  }  class Producer implements Runnable{  Q q;  Producer(Q new\_inst){  q = new\_inst;  new Thread(this, "StackInsert").start();  }  public void run(){  while(true){  q.push();  }  }  }  class Consumer implements Runnable{  Q q;  Consumer(Q new\_inst){  q = new\_inst;  new Thread(this, "StackDelete").start();  }  public void run(){  while(true){  q.pop();  }  }  }  class PCArray{  public static void main(String arg[]){  Q obj = new Q();  Producer p\_obj = new Producer(obj);  Consumer c\_obj = new Consumer(obj);  }  } |

Output:

