Advanced Programming Language

Assignment 2

Chinmay Dorge

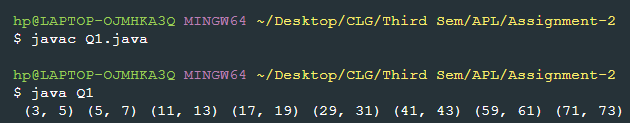
LCS2020022

Q1) Write a Java method to find all twin prime numbers less than 100 such as (3, 5), (5, 7), (11, 13)………

Code: Q1.java

public class Q1{  
 static void printTwinPrime(int n)  
 {  
 boolean prime[] = new boolean[n + 1];  
 for (int i = 0; i <= n; i++)  
 prime[i] = true;  
 for (int p = 2; p \* p <= n; p++) {  
 if (prime[p] == true) {  
 for (int i = p \* 2; i <= n; i += p)  
 prime[i] = false;  
 }  
 }  
 for (int i = 2; i <= n - 2; i++) {  
 if (prime[i] == true &&  
 prime[i + 2] == true)  
 System.out.print(" (" + i + ", " +  
 (i + 2) + ")");  
 }  
 }  
 public static void main(String args[]) {  
 printTwinPrime(100);  
 }  
}

Output:



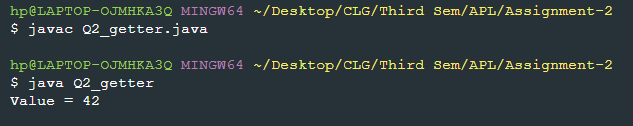
Q2) Write a program where you only define setter method in the class to make write-only. Similarly, write a program where you only define getter method in the class to make read-only.

1. Using getter

Code: Q2\_getter.java

class getterOnlyClass{  
 int val;  
 getterOnlyClass(int val){  
 this.val = val;  
 }  
 public int getValue(){  
 return this.val;  
 }  
}  
public class Q2\_getter {  
 public static void main(String args[]){  
 getterOnlyClass g = new getterOnlyClass(42);  
 System.out.println("Value = " + g.getValue());  
 }  
}

Output:

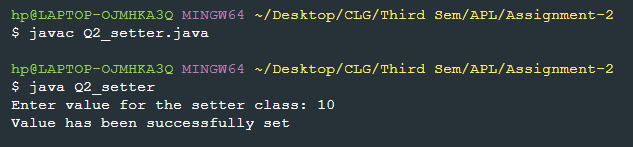


1. Using setter

Code: Q2\_setter.java

import java.util.Scanner;  
class setterOnlyClass{  
 int val;  
 public void setValue(int val){  
 this.val = val;  
 System.out.println("Value has been successfully set ");  
 }  
}  
public class Q2\_setter{  
 public static void main(String args[]){  
 Scanner sc = new Scanner(System.in);  
 System.out.print("Enter value for the setter class: ");  
 int input = sc.nextInt();  
 setterOnlyClass s = new setterOnlyClass();  
 s.setValue(input);  
 }  
}

Output:

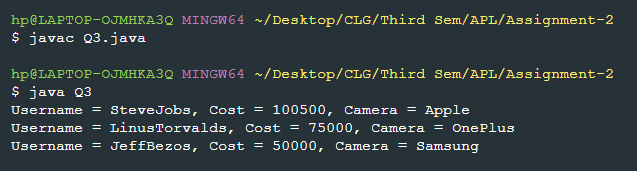


Q3) Write a simple program with a class “Mobile” and declare the variable cost and camera as private and username as public. Next print the values of three different users Mobile type based on three member variables.

Code: Q3.java

class Mobile{  
 public String userName;  
 private int cost;  
 private String camera;  
 Mobile(String userName , int cost , String camera){  
 this.userName = userName;  
 this.cost = cost;  
 this.camera = camera;  
 }  
 void printData(){  
 System.out.println("Username = " + userName + ", Cost = " + cost + ", Camera = " + camera);  
 }  
}  
public class Q3{  
 public static void main(String args[]){  
 Mobile m1 = new Mobile("SteveJobs" , 100500 , "Apple");  
 Mobile m2 = new Mobile("LinusTorvalds" , 75000 , "OnePlus");  
 Mobile m3 = new Mobile("JeffBezos" , 50000 , "Samsung");  
  
 m1.printData();  
 m2.printData();  
 m3.printData();  
 }  
}

Output:



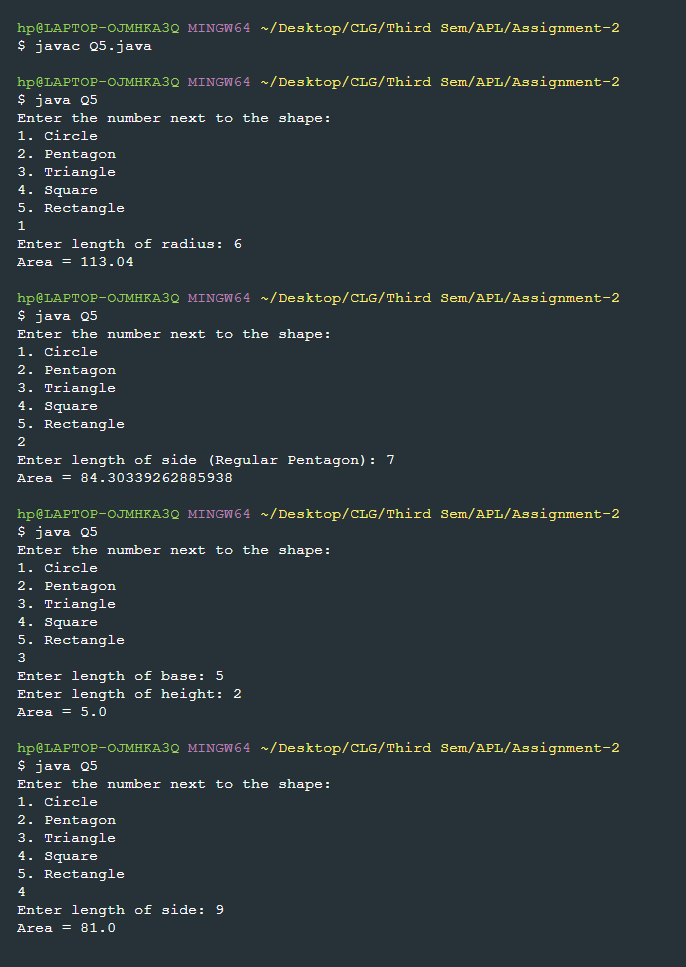
Q4) Assume that a program contains one method to set the random location of the unmanned vehicle and two getter methods to retrieve the accurate location value in relevant units. Please justify which java concept is used in the above sample program.

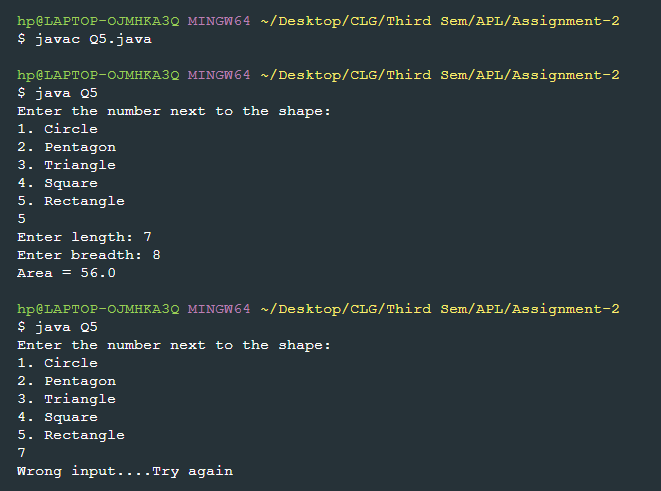
In the sample program, the concepts of encapsulation and abstraction in java are being used. As the variable, random location, is being set by the setter method and being retrieved by getter methods it would have been declared as private. So only the functions within the class are allowed to access this variable, encapsulating it from the rest of the world and providing a layer of abstraction for the user and other programmers.

Q5) Write a Java program to calculate the area of circle, pentagon, triangle, square and rectangle using switch statement. Remember at any time users/programmer can see the area of any one of the geometrical shapes.

Code: Q5.java

import java.util.Scanner;  
public class Q5{  
 public static void main(String arg[]){  
 Scanner sc = new Scanner(System.in);  
 System.out.println("Enter the number next to the shape: ");  
 System.out.println("1. Circle ");  
 System.out.println("2. Pentagon");  
 System.out.println("3. Triangle");  
 System.out.println("4. Square");  
 System.out.println("5. Rectangle");  
 int input = sc.nextInt();  
 switch(input){  
 case 1:  
 System.out.print("Enter length of radius: ");  
 double radius = sc.nextInt();  
 System.out.println("Area = " + (radius\*radius\*(3.14)));  
 break;  
 case 2:  
 System.out.print("Enter length of side (Regular Pentagon): ");  
 double p = sc.nextInt();  
 double area\_p = (0.25)\*(Math.sqrt(5\*(5+(2\*Math.sqrt(5)))))\*(p\*p);  
 System.out.println("Area = " + area\_p);  
 break;  
 case 3:  
 System.out.print("Enter length of base: ");  
 double base = sc.nextInt();  
 System.out.print("Enter length of height: ");  
 double height = sc.nextInt();  
 System.out.println("Area = " + (0.5 \* base \* height));  
 break;  
 case 4:  
 System.out.print("Enter length of side: ");  
 double side = sc.nextInt();  
 System.out.println("Area = " + (side \* side));  
 break;  
 case 5:  
 System.out.print("Enter length: ");  
 double length = sc.nextInt();  
 System.out.print("Enter breadth: ");  
 double breadth = sc.nextInt();  
 System.out.println("Area = " + (length \* breadth));  
 break;  
 default:  
 System.out.println("Wrong input....Try again ");  
 }  
 }  
}

Output:

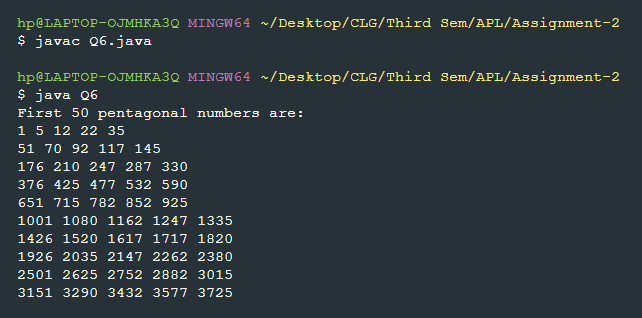


Q6) Write a Java method to display the first 50 pentagonal numbers such as (1, 5, 12, 22, 35, 51, ………).

Code: Q6.java

class pentagonalNumbers{  
 public void calculate(){  
 for(int i=1;i<=50;i++){  
 int num=(i\*(3\*i-1))/2;  
 System.out.print(num + " ");  
 if(i%5==0)  
 System.out.println();  
 }  
 }  
}  
public class Q6{  
 public static void main(String arg[]){  
 System.out.println("First 50 pentagonal numbers are: ");  
 pentagonalNumbers obj = new pentagonalNumbers();  
 obj.calculate();  
 }  
}

Output:



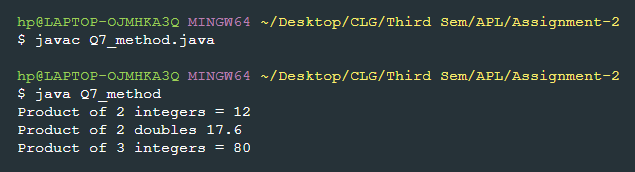
Q7) Write two Java programming on method overloading and constructor overloading in Java by varying the number of arguments and changing the data types of the arguments of multiplication.

1. Method Overloading

Code: Q7\_method.java

class methodOverloading{  
 public void multiply(int a , int b){  
 int ans = a\*b;  
 System.out.println("Product of 2 integers = " + ans);  
 }  
 public void multiply(double a , double b){  
 double ans = a\*b;  
 System.out.println("Product of 2 doubles " + ans);  
 }  
 public void multiply(int a , int b , int c){  
 int ans = a\*b\*c;  
 System.out.println("Product of 3 integers = " + ans);  
 }  
}  
  
public class Q7\_method{  
 public static void main(String arg[]){  
 methodOverloading m = new methodOverloading();  
 m.multiply(3,4);  
 m.multiply(3.2 , 5.5);  
 m.multiply(8,5,2);  
 }  
}

Output:

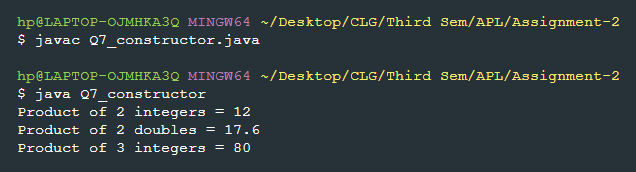


1. Constructor Overloading

Code: Q7\_constructor.java

class Multiply{  
 Multiply(int a , int b){  
 int ans = a\*b;  
 System.out.println("Product of 2 integers = " + ans );  
 }  
 Multiply(double a , double b){  
 double ans = a\*b;  
 System.out.println("Product of 2 doubles = " + ans );  
 }  
 Multiply(int a , int b , int c){  
 int ans = a\*b\*c;  
 System.out.println("Product of 3 integers = " + ans);  
 }  
}  
public class Q7\_constructor{  
 public static void main(String args[]){  
 Multiply m1 = new Multiply(3,4);  
 Multiply m2 = new Multiply(3.2 , 5.5);  
 Multiply m3 = new Multiply(8,5,2);  
 }  
}

Output:



Q8) Write a Java program to print the Fibonacci series of the number up to 100.

Code: Q8.java

public class Q8 {  
 static void Fibonacci(){  
 int a=0,b=1;  
 System.out.println(a);  
 while(b<=100){  
 System.out.println(b);  
 int temp = b;  
 b += a;  
 a = temp;  
 }  
  
 }  
 public static void main(String arg[]){  
 System.out.println("Fibonacci series upto 100 is as follows ");  
 Fibonacci();  
 }  
}

Output:

