|  |  |
| --- | --- |
| File:COMSATS new logo.jpg - Wikimedia Commons  OBJECT ORIENTED PROGRAMMING  *Lab Assignment*  *Generics* | **submitted by:**  **Shahzaneer Ahmed**  **registration number:**  **sp21-bcs-087**  **submitted to:**  **mA’M sANEEHA aMIR**  **date of submission:**  **may 27, 2022** |

Question 1

# Pair

public class Pair<T> {

    private T first;

    private T second;

    public Pair() {

    }

    public Pair(T first, T second) {

*this*.first = first;

*this*.second = second;

    }

    public T getFirst() {

        return *this*.first;

    }

    public void setFirst(T first) {

*this*.first = first;

    }

    public T getSecond() {

        return *this*.second;

    }

    public void setSecond(T second) {

*this*.second = second;

    }

    @Override

    public String toString() {

        return "{" +

            " first='" + getFirst().toString() + "'" +

            ", second='" + getSecond().toString() + "'" +

            "}";

    }

}

# Runner

public class Runner {

    public static void main(String[] args) {

        Pair<String> pair = new Pair<>("Hello", "World");

        Pair<Integer> pair2 = new Pair<>(1, 2);

        Pair<Double> pair3 = new Pair<>(1.0, 2.0);

        Pair<Boolean> pair4 = new Pair<>(true, false);

        Pair<Character> pair5 = new Pair<>('a', 'b');

        System.out.println(pair);

        System.out.println(pair2);

        System.out.println(pair3);

        System.out.println(pair4);

        System.out.println(pair5);

    }

}

Question 2

# Pair

public class Pair<T1,T2>{

    private T1 first;

    private T2 second;

    public Pair() {

    }

    public Pair(T1 first, T2 second) {

*this*.first = first;

*this*.second = second;

    }

    public T1 getFirst() {

        return first;

    }

    public T2 getSecond() {

        return second;

    }

    public void setFirst(T1 first) {

*this*.first = first;

    }

    public void setSecond(T2 second) {

*this*.second = second;

    }

    @Override

    public String toString() {

        return "(" + first.toString() + ", " + second.toString() + ")";

    }

}

# Runner

public class Runner {

    public static void main(String[] args) {

        Pair<Integer, String> pair = new Pair<Integer, String>(1, "one");

        Pair<Double, Boolean> pair2 = new Pair<>(2.4,true);

        Pair<String, Double> pair3 = new Pair<>("Shahzaneer Ahmed - Data Scientist",10.0);

        Pair<Character, Student> pair4 = new Pair<>('A',new Student("Abdur Rehman", 2));

*// These types should not be the primitives but be the the wrapper classes*

        System.out.println(pair);

        System.out.println(pair2);

        System.out.println(pair3);

        System.out.println(pair4);

    }

}

# Student

public class Student {

    private String name;

    private int rollNumber;

    public Student() {

    }

    public Student(String name, int rollNumber) {

*this*.name = name;

*this*.rollNumber = rollNumber;

    }

    public String getName() {

        return *this*.name;

    }

    public void setName(String name) {

*this*.name = name;

    }

    public int getRollNumber() {

        return *this*.rollNumber;

    }

    public void setRollNumber(int rollNumber) {

*this*.rollNumber = rollNumber;

    }

    @Override

    public String toString() {

        return "{" +

            " name='" + getName() + "'" +

            ", rollNumber='" + getRollNumber() + "'" +

            "}";

    }

}

Question 3

# MYMathClass

import java.util.ArrayList;

public class MyMathClass<T extends  Number>{

*// Now the Type Parameter for the class should be the descendents of Numbers class!*

    private T number;

    public MyMathClass() {

    }

    public MyMathClass(T number) {

*this*.number = number;

    }

    public double getStandardDeviation(ArrayList<T> values) {

        double sum = 0.0, standardDeviation = 0.0;

        int length = values.size();

        for(T num : values) {

            sum += num.doubleValue();

        }

        double mean = sum/length;

        for(T num: values) {

            standardDeviation += Math.pow(num.doubleValue() - mean, 2);

        }

        return Math.sqrt(standardDeviation / length);

    }

}

# Runner

import java.util.ArrayList;

public class Runner {

    public static void main(String[] args) {

*// Making three arrays of Number Class Descendants*

            ArrayList <Integer> a1 = new ArrayList<>();

            ArrayList <Double> a2 = new ArrayList<>();

            ArrayList<Float> a3 = new ArrayList<>();

            ArrayList<String> a4 = new ArrayList<>();

            a1.add(10);

            a1.add(11);

            a1.add(12);

            a1.add(13);

            a1.add(15);

            a2.add(1.5);

            a2.add(2.743225);

            a2.add(4.6532);

            a2.add(54.8643);

            a2.add(99.9999);

            a3.add(1.1f);

            a3.add(15.0f);

            a3.add(11.65f);

            a3.add(10.32f);

            a3.add(9.31f);

            a4.add("Shah");

            a4.add("Laiba");

            a4.add("Raheem");

            a4.add("Anas");

            a4.add("Rabbiya");

*// Calculating Standard Deviation for each Arraylist*

            MyMathClass <Integer> m1 = new MyMathClass<>();

            MyMathClass <Double> m2 = new MyMathClass<>();

            MyMathClass<Float> m3 = new MyMathClass<>();

*// MyMathClass<String> m4 = new MyMathClass<>(); // compiler will complain String is not the descendent of Numbers*

            MyMathClass<Number> m4 = new MyMathClass<>();

*// m4.getStandardDeviation(a4); complaining compiler will throw error!*

            System.out.println(m1.getStandardDeviation(a1));

            System.out.println(m2.getStandardDeviation(a2));

            System.out.println(m3.getStandardDeviation(a3));

        }

    }

Question 4

# Generic Drawing

import java.util.ArrayList;

import java.util.Random;

public class GenericDrawing<T> {

    ArrayList<T> box = new ArrayList<>();

    public void add(T item) {

        box.add(item);

    }

    public boolean isEmpty() {

        int size = box.size();

        return size == 0;

    }

    public T drawItem() {

        if (isEmpty())

            return null;

        T drawingItem = null;

*//      random choice*

*// int randomChoice = (int) Math.random() \* box.size();*

        Random r = new Random();

        int randomChoice = r.nextInt(box.size());

        drawingItem = box.get(randomChoice);

*// box.remove(randomChoice);*

            return drawingItem;

    }

}

# Runner

public class Runner {

    public static void main(String[] args) {

        GenericDrawing <String> box1 = new GenericDrawing<>();

        GenericDrawing<Integer> box2 = new GenericDrawing<>();

        box1.add("Shahzaneer Ahmed");

        box1.add("Data Scientist");

        box1.add("Mobile App Developer");

        box1.add("Entrepreneur");

        box1.add("Community Builder");

        box2.add(1);

        box2.add(2);

        box2.add(3);

        box2.add(4);

        box2.add(5);

        box2.add(6);

        System.out.println(box1.drawItem());

        System.out.println(box2.drawItem());

    }

}

Question 5

# GenericStack

import java.util.ArrayList;

public class GenericStack<E>{

    private ArrayList<E> list = new ArrayList<>();

    public int getSize() {

        return list.size();

    }

    public void push(E o) {

        list.add(o);

    }

    public E pop() {

        E o = list.get(getSize() - 1); *//last element will be returned*

        list.remove(getSize() - 1);

        return o;

    }

    public boolean isEmpty() {

        return list.isEmpty();

    }

    @Override

    public String toString() {

        return "Stack : " + list.toString();

*// As Arraylist is object type so it has toString method.*

    }

}

# Runner

public class Runner {

    public static void main(String[] args) {

        GenericStack<String> stack1 = new GenericStack<String>();

        stack1.push("Islamabad");

        stack1.push("Lahore");

        stack1.push("Karachi");

        GenericStack<Integer> stack2 = new GenericStack<Integer>();

        stack2.push(1);

        stack2.push(2);

        stack2.push(3);

        System.out.println(stack1);

        System.out.println(stack2);

    }

}