Programming Systems and Environments – Lab 3

Jakub Grzana, 241530

Task 1 – Numbers

I've created basic dynamic array that gets resized every time new element is added. To achieve complexity O(1) for getting min and max value in the list I've made private attributes storing indexes of those values, updated each time new element is added. Similarly, to achieve O(1) for calculation of average value, I've made private attribute "sum" that stores sum of all elements in the list. Random number generator used in this task is safe for cryptographic purposes. To get insight into objects of this class, I've overloaded toString method.

Task 2 – Serialization

I've used Gson library to serialize/deserialize Java objects to JSON that was then saved into files. It's great, generic way to save variables (objects) to be used by different program. As far as I know, it will work just fine regardless of programming language or library version (opposite to Pythonic pickle.dumps, which is language and version dependent)

Task 3 – Multinumbers

To store list of Numbers I've used ArrayList. Added overloaded toString method that itself uses Numbers.toString to print content of Multinumbers object. Serialization works exactly the same as in Task 2.

Conclusions

It's weird that despite being high-level Java and that It uses garbage collector, Java doesn't automatically close files. Gson library turned out to be great tool. Adding dependencies to project is easy, but finding proper library and how to import it is properly is still challenging – at least for novices.

I didn't manage exceptions in code properly cause I wasn't asked to in task description and doing so would be bothersome.

CODE: Mavenproject1.java

```
public class Mavenproject1 {

    public static void main(String[] args) throws Exception {

        Numbers z = new Numbers(10, 1, 10);
        z.Save("temp.json");

        Numbers n = new Numbers("temp.json");
        System.out.println(n);

        MultiNumbers v = new MultiNumbers("araara.json");
        System.out.println(v);
        v.Save("araara.json");
    }
}
```

CODE: Numbers.java

```
import com.google.gson.*;
import java.io.*;

import java.util.Arrays;
import java.security.SecureRandom;

import java.io.File;
import java.io.IOException;

public class Numbers {
    private double array[];
    final private static SecureRandom randomGenerator = new SecureRandom();
    private double sum;
    private int size;
    private int min_index;
    private int max_index;

    private static int GetRandomNum(int min, int max)
    {
        final int diff = Math.abs(max - min);
        final int rand_val = randomGenerator.nextInt(diff);
        return min + rand_val;
    }
```

```
public Numbers(int n, int min, int max)
{
   this.array = new double[n];
   this.sum = 0;
   this.size = n;
   this.min_index = 0;
   this.max_index = 0;
   for(int i = 0; i < n; ++i)
   {
      this.array[i] = this.GetRandomNum(min, max);
      this.sum = this.sum + this.array[i];
      if(this.array[i] > this.array[this.max_index]) this.max_index = i;
      if(this.array[i] < this.array[this.min_index]) this.min_index = i;
   }
}

public Numbers(String filename) throws Exception
{
   this.Load(filename);
}

@Override public String toString()
{
   String out = "[ ";
   for(int i = 0; i < this.size; ++i)
   {
      out = out + this.array[i] + ", ";
   }
   out = out + "]";
   return out;
}

public void Add(double num)
{
   double new_array[] = new double[this.size+1];
   System.arraycopy(this.array, 0, new_array, 0, this.size);
   this.array = new_array;
   this.size = this.size + 1;
   if(num > this.Max()) this.max_index = this.size - 1;
   if(num < this.Min()) this.min_index = this.size - 1;
}
```

```
    public void Sort()
    {
        Arrays.sort(this.array);
        this.min_index = 0;
        this.max_index = this.size - 1;
    }
    public double Average() { return this.sum / this.size; }
    public double Min() { return this.array[this.min_index]; }
    public double Max() { return this.array[this.max_index]; }
    public double Median()
    {
        double arr[] = new double[this.size];
        System.arraycopy(this.array, 0, arr, 0, this.size);
        Arrays.sort(arr);

        if((this.size % 2) == 1) return arr[this.size / 2];
        else {
            double l = arr[this.size / 2 - 1];
            double r = arr[this.size / 2];
            return (l+r)/2;
        }
    }

    public void Save(String filename) throws Exception
    {
        Gson gson = new Gson();
        try (FileWriter writer = new FileWriter(new File(filename))) {
            gson.toJson(this, writer);
            writer.flush();
        }
    }

    public void Load(String filename) throws Exception
    {
        Gson gson = new Gson();
        try (FileReader reader = new FileReader(new File(filename))) {
            Numbers loaded = gson.fromJson(reader, Numbers.class);
            this.array = loaded.array;
            this.size = loaded.size;
            this.sum = loaded.sum;
            this.min_index = loaded.min_index;
            this.max_index = loaded.max_index;
        }
    }
}
```

CODE: MultiNumbers.java

```
import com.google.gson.Gson;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.util.ArrayList;
/**
 *
 * @author meron
 */
public class MultiNumbers {
    ArrayList<Numbers> numbers;

    public MultiNumbers(int s, int n)
    {
        numbers = new ArrayList<Numbers>();
        for(int i = 0; i < s; ++i)
        {
            numbers.add(new Numbers(n, 0, 10));
        }
    }

    public MultiNumbers(String filename) throws Exception
    {
        this.Load(filename);
    }

    public void Save(String filename) throws Exception
    {
        Gson gson = new Gson();
        try (FileWriter writer = new FileWriter(new File(filename))) {
            gson.toJson(this, writer);
            writer.flush();
        }
    }

    public void Load(String filename) throws Exception
    {
        Gson gson = new Gson();
        try (FileReader reader = new FileReader(new File(filename))) {
            MultiNumbers loaded = gson.fromJson(reader, MultiNumbers.class);
            this.numbers = loaded.numbers;
        }
    }
```

```
    @Override public String toString()
    {
        String out = "";
        for(int i = 0; i < this.numbers.size(); ++i)
        {
            out = out + this.numbers.get(i) + "\n";
        }
        return out;
    }
}
```