University of Wisconsin PARKSIDE

Department of Computer Science

CSCI 340/CIS 540 Data Structures/Algorithm Design

Homework #1: 50 points

Outcomes:

- Use `Iterable` interface defined in `java.lang` package as a generic type `Iterable<T>` T type parameter (type of elements returned by the iterator)
- Use `for-each` loop for iterating over collections.
- Show iterating over a List collection with `String`, `Integer`, and `Double` objects as elements
- Write programs that use `Iterable` interface.
- Reinforce concepts learned in CSCI 242.
- Generate random numbers for experimental analysis of algorithms.

Scoring:

- If you do not submit a zip file containing your source code, your score will be zero.
- If you submit the source code that does not compile, your score will be zero.
- If you submit source code without the correct class names (`MyIterableTester`, `StdRandomTester`) or your program partially works, you will receive at most half credit for this assignment.
- Deductions will be made for not meeting the usual requirements:
    - The source code is not formatted according to the usual style guidelines, including commenting on each method to explain its purpose.
- Suggested Readings:
    - https://docs.oracle.com/javase/tutorial/java/generics/index.html
    - https://en.wikipedia.org/wiki/Generics_in_Java
    - https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html
    - https://www.oracle.com/java/technologies/javase/codeconventions-contents.html
    - https://google.github.io/styleguide/javaguide.html

Create an IntelliJ Java Project called Program1. Write the comment lines to include the author's name(s) and complete the project following the instructions specified in Part 1 and Part 2.

Part 1: 25 points

Assignment:

1. You are given the class `MyIterable` that implements the `Iterable` interface and a text file "input.txt". Read and understand the purpose of the `MyIterable` Class. The `Iterable` interface has a method to override: `Iterator <T> iterator()`.

2. Write a class `MyIterableTester` that reads input arrays of *any type* (Read Step 3, Note – for more details) and iterates it in a **`for-each`** loop and displays the input in **reverse sorted order**.
3. `MyIterableTester` class reads the input data for a text file (input.txt) into appropriate `ArrayLists` of different types (String, Integer, and Double) and stores them into corresponding arrays.
   Note: (i) You need to create the input.txt file with the data provided. (ii) Use `hasNext()`, `hasNextInt()` and `hasNextDouble()` to read appropriate data into different ArrayLists using while loops. (iii) convert the ArrayLists into arrays that serve as inputs in `MyIterableTester` class.
   MyIterable Class:

```
import java.util.List;
import java.util.Arrays;
import java.util.Iterator;
import java.util.Collections;

public class MyIterable<T> implements Iterable<T> {

    private List<T> list;

    public MyIterable(T [] t) {
        list = Arrays.asList(t);
        Collections.sort(list,Collections.reverseOrder());
    }

    @Override
    public Iterator<T> iterator() {

        return list.iterator();
    }
}
```

input.txt

```
Andrew Sasha Mike John Amy Prince Patty
12 2 32 14 5 10 7 6
10.4 30.6 4.95 4.9 6.9
```

Note: This is just a sample data provided for you to test the program. Your program should work for any data file of varying numbers of data inputs.

Sample run:

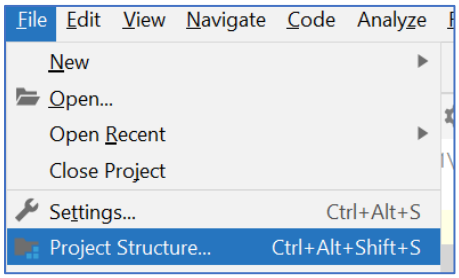
```
My friends:
Sasha Prince Patty Mike John Andrew Amy
----------+---------
My numbers:
32 14 12 10 7 6 5 2
----------+---------
My scores:
30.6 10.4 6.9 4.95 4.9
----------+---------
```

Part 2: 25 points

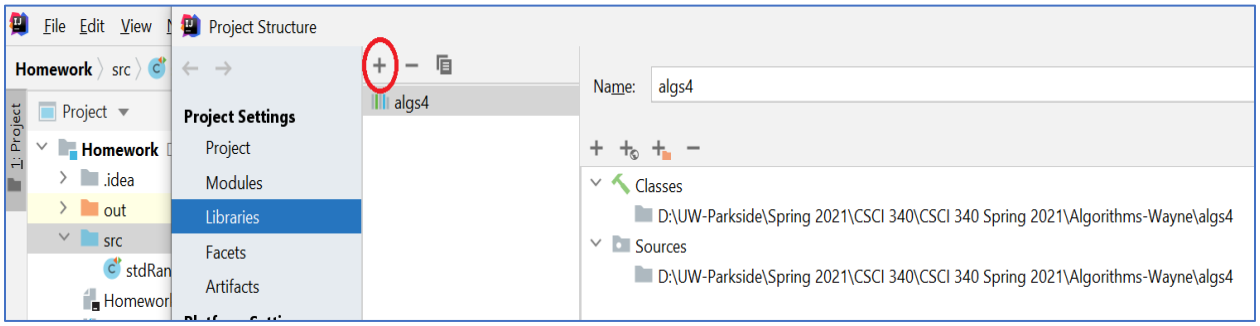
1. Install the algs4.jar library of code from the reference book's website. https://algs4.cs.princeton.edu/code/ into the course folder of your computer.
2. Read this article about probability distributions:
   https://www.analyticsvidhya.com/blog/2017/09/6-probability-distributions-data-science
3. You can also locate a collection of java files from the library of code that you downloaded in step 1. Then, you should open and get the methods in the class **StdRandom.java** running. Alternatively, you can also download the class **StdRandom.java** from the following link.
   https://github.com/kevin-wayne/algs4/blob/master/src/main/java/edu/princeton/cs/algs4/StdRandom.java
   Understand how the methods work to generate different kinds of random numbers (See Step 2). You also notice the best practices that the authors have used in documenting the code.

Assignment:

4. Create a java source file StdRandomTester.java in your project **Program1** and copy the contents of StdRandom.java into it. Don't forget to modify the class name in the header to StdRandomTester.
5. You have to link the algs4 folder to the path.
   For this, Locate and click the **Project Structure** under the **File** menu of the IntelliJ IDEA editor as shown.



   Under **Project Settings**, click Libraries. Then, click the **'+'** symbol shown in the following figure. The **Select Library Files** window appears. Add the 'algs4' library path and click **OK**.



6. Now, modify the main() method to create and display two arrays of one thousand and 10 thousand random numbers, respectively, using any two distributions of your choice.

Grading Rubric:

Requirement	Full credit	Partial credit
Implement the `MyIterableTester` Class using the data from the text file "input.txt" (20 points)	Your program implements the `MyIterableTester` Class as described.	Your program implements the class, but with some errors.
Implement the `StdRandomTester` Class using the StdRandom Class from the reference book code: https://algs4.cs.princeton.edu/code/ (20 points)	Your program implements the `StdRandomTester` Class as described.	Your program implements the class, but with some errors.
Format code (follow Style Guidelines) and output as specified (10 points)	Your output is formatted as specified, including proper spacing, spelling, and so on.	You did not follow some or all of the requirements for output.

Submission:

If you are working in pairs, only one of you should take the lead (which you should agree upon earlier) and submit the assignment and leave a comment on the other member's name that teamed up with you for the project. This way, multiple instances of the same file will not be submitted, making the grading process difficult.

Follow these steps to submit your work:

a. Zip up your entire IntelliJ IDEA project (the project contains the java source code files `MyIterable`, `MyIterableTester.java` and `StdRandomTester.java`) into a single zip file named program1.
b. The instructor should be able to unzip the entire project and load it in IntelliJ IDEA without incident.
c. Submit only the zip file to the Canvas website. Please ensure that only one of the team member submits the project.

Note: If you submit your work and decide to modify the program, you need to resubmit a new zip file containing the source code file. Do not rename your source code file. Do not rename the zip file. Canvas may add a number to the name of your zip file, which is fine. But you keep the filenames the same.