Homework 1

carlos guisao

COP5339

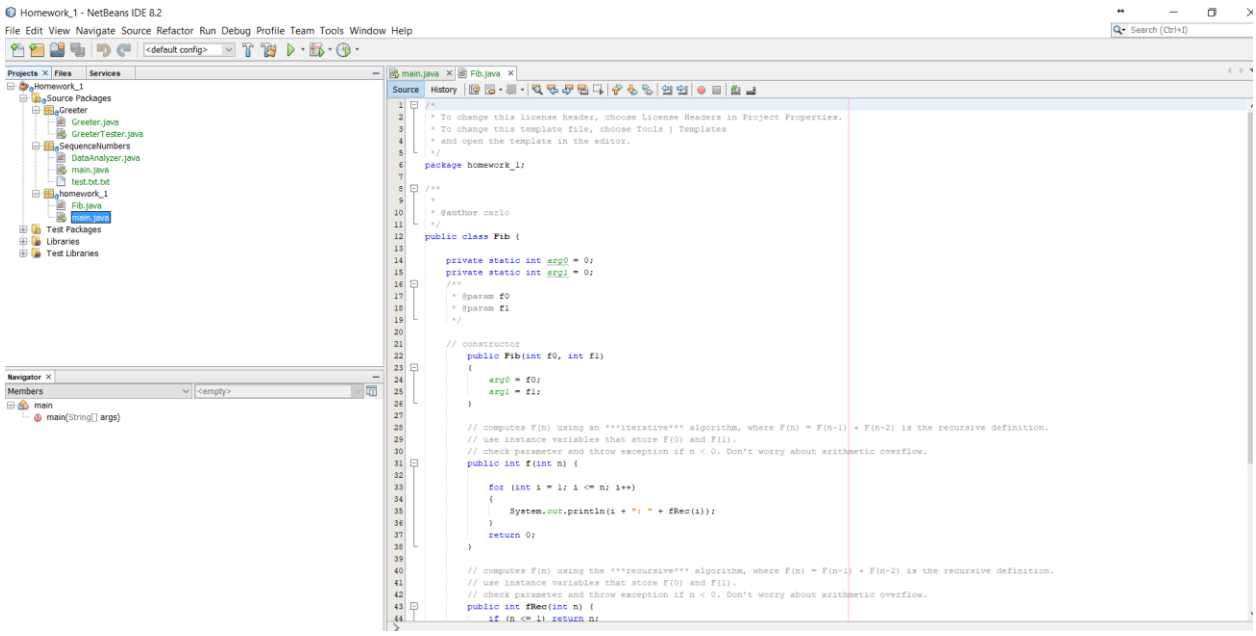
January 21, 2018

1

Write a Java program that calculates and displays the Fibonacci

series, defined by the recursive formula F(n) = F(n-1) + F(n-2).

F(0) and F(1) are given on the command line.



Fib Class

main.java | Fib.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package homework_1;

/**
 *
 * @author carlo
 */
public class Fib {

    private static int arg0 = 0;
    private static int arg1 = 0;
    /**
     * @param f0
     * @param f1
     */

    // constructor
        public Fib(int f0, int f1)
        {
            arg0 = f0;
            arg1 = f1;
        }

        // computes F(n) using an ***iterative*** algorithm, where F(n) = F(n-1) + F(n-2) is the recursive definition.
        // use instance variables that store F(0) and F(1).
        // check parameter and throw exception if n < 0. Don't worry about arithmetic overflow.
        public int f(int n) {

            for (int i = 1; i <= n; i++)
            {
                System.out.println(i + ": " + fRec(i));
            }
            return 0;
        }
```

```java
        }

        // computes F(n) using the ***recursive*** algorithm, where F(n) = F(n-1) + F(n-2) is the recursive definition.
        // use instance variables that store F(0) and F(1).
        // check parameter and throw exception if n < 0. Don't worry about arithmetic overflow.
        public int fRec(int n) {
            if (n <= 1) return n;
            else return fRec(n-1) + fRec(n-2);
        }
}
```

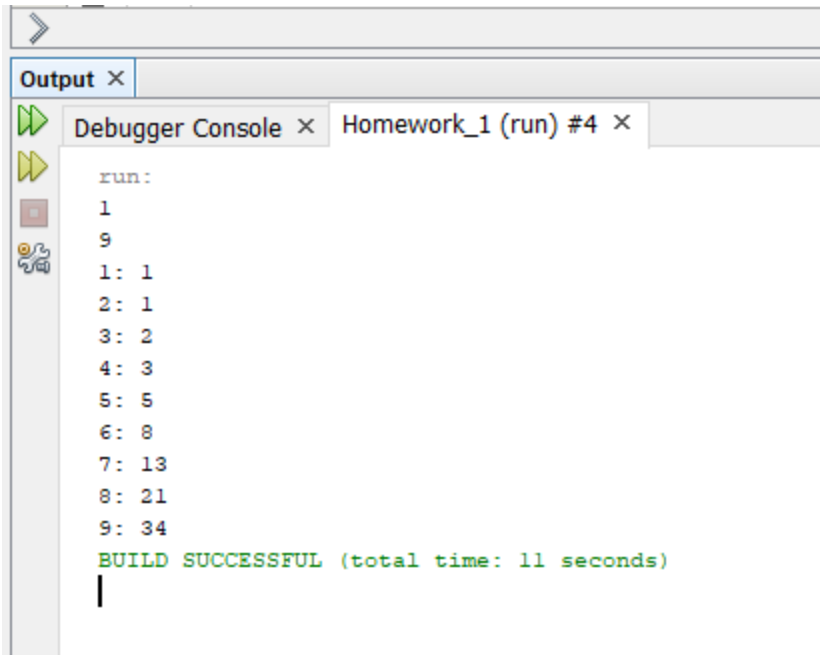
Main Class

main.java

Source History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package homework_1;

import java.util.InputMismatchException;
import java.util.Scanner;

/**
 *
 * @author carlo
 */
public class main {
        public static void main(String[] args) throws Exception {

        /*1
            Write a Java program that calculates and displays the Fibonacci
            series, defined by the recursive formula F(n) = F(n-1) + F(n-2).
            F(0) and F(1) are given on the command line.
            Define and use a class Fib with the following structure:
        */

        int arg0 = 0;
        int arg1 = 0;

        Scanner cin = new Scanner(System.in);

        try{
            arg0 = cin.nextInt();
            arg1 = cin.nextInt();
        }catch(InputMismatchException ex)
        {
            throw new Exception("No a good, try again. This time an integer: "
                    + ex.getMessage());
        }

        Fib fibonacci = new Fib(arg0, arg1);

        fibonacci.f(arg1);

    }
}
```

Results:



Write javadoc comments for the Fib class

2

a. Write a new method for the Greeter class,

```
public void swapNames(Greeter other) {...}
```

that swaps the names of this greeter and another instance.

b. write a new method for the Greeter class:

```
public Greeter createQualifiedGreeter(String qualifier) { ..... }
```

that returns a new Greeter object with its name being the qualifier string followed by

" " and the executing greeter's name (i.e. this.name).

For example:

```
Greeter g = new Greeter("world");
```

```
Greeter g2 = g.createQualifiedGreeter("beautiful");
```

g2.name will be the string "beautiful world"

c. Write a GreeterTester class that shows how the swapNames() and the createQualifiedGreeter() methods are used.

Greeter:

Greeter.java | GreeterTester.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Greeter;

/**
 *
 * @author carlo
 */
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**

/**
   A class for producing simple greetings. (Revised to include sayGoodbye)
*/
public class Greeter
{
   /**
      Constructs a Greeter object that can greet a person or
      entity.
      @param aName the name of the person or entity who should
      be addressed in the greetings.
   */
   public Greeter(String aName)
   {
      name = aName;
   }
```

```java
    /**
       Greet with a "Goodbye" message.
       @return a message containing "Goodbye" and the name of
       the greeted person or entity.
    */
    public String sayGoodbye()
    {
       return "Goodbye, " + name + "!";
    }

    /**
       Greet with a "Hello" message.
       @return a message containing "Hello" and the name of
       the greeted person or entity.
    */
    public String sayHello()
    {
       return "Hello, " + name + "!";
    }

    public String giveName()
    {
        return name;
    }

    private String name;

    public void swapNames(Greeter other)
    {
        String temp = "";
        temp = this.name;
        this.name = other.name;
        other.name = temp;
    }


    public Greeter createQualifiedGreeter(String qualifier)
    {
        this.name += " " + qualifier;
        return new Greeter(qualifier + this.name);
    }
}
```

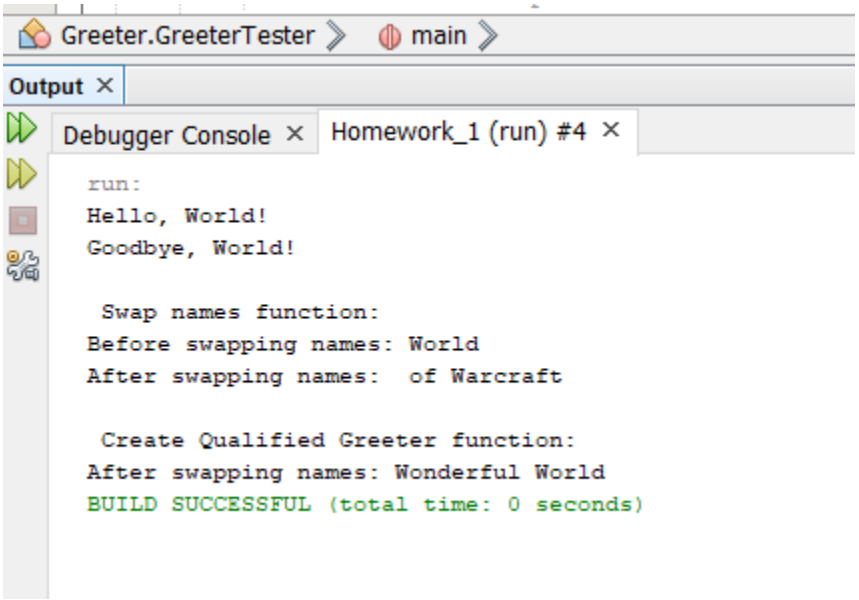
Greeter Tester

Greeter.java | GreeterTester.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Greeter;

/**
 *
 * @author carlo
 */
public class GreeterTester {
    public static void main(String[] args)
    {
        /*
            Part 2 of the homework 1
            Greetertester part
        */

        Greeter worldGreeter = new Greeter("World");
        String greeting = worldGreeter.sayHello();
        System.out.println(greeting);

        greeting = worldGreeter.sayGoodbye();
        System.out.println(greeting);

        Greeter world = new Greeter("World");
        Greeter warcraft = new Greeter(" of Warcraft");

        System.out.println("\n Swap names function:");
        System.out.println("Before swapping names: " + world.giveName());
        world.swapNames(warcraft);
        System.out.println("After swapping names: " + world.giveName());

        System.out.println("\n Create Qualified Greeter function:");
        Greeter BetterGreeter = new Greeter("Wonderful");
        BetterGreeter.createQualifiedGreeter("World");
        System.out.println("After swapping names: " + BetterGreeter.giveName());
    }
}
```

Results:



Write a program that:

a. reads from the terminal a sequence of numbers (integers)

b. saves them to a file with the name given from the command line

c. calculates,then displays on the terminal, and also saves to that file

the maximum, minimum, and average.

Additional requirements:

Store the numbers in a LinkedList<Integer>.

Define a class DataAnalyzer that

* has a constructor that stores the list of numbers:

```
public DataAnalyzer(LinkedList<Integer> numList) {...}
```

* has a method each for computing min(), max() and average():

```
public int min() {...}, etc.
```

Define a class DataAnalyzerTester that reads the numbers from System.in, builds the number list, creates the DataAnalyzer object, and displays the min, max, and average using the DataAnalyzer instance.

The DataAnalyzerTester class implements the main() method.

Your code needs to handle invalid input and I/O exceptions.

Write javadoc comments.

Include both java files in your solution document.

Data Analyzer:

main.java | DataAnalyzer.java

Source | History

```java
import java.util.LinkedList;

/**
 *
 * @author carlos Guisao
 */

public class DataAnalyzer {
    private final LinkedList<Integer> list;

    public DataAnalyzer(LinkedList<Integer> IncomingList)
    {
        list = IncomingList;
    }

    public int FindMax()
    {
        Collections.sort(list);
        return list.getLast();
    }

    public int FindMin()
    {
        Collections.sort(list);
        return list.getFirst();
    }

    public short FindAverage()
    {
        short divide = 0;
        short sum = 0;
        for(short i= 0; i < list.size(); i++)
        {
            sum += list.get(i);
            divide++;
        }
        return (short) (sum/divide);
    }
}
```

Main:

main.java | DataAnalyzer.java

Source | History

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package SequenceNumbers;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.util.LinkedList;
import java.util.Scanner;

/**
 *
 * @author carlos Guisao
 */


public class main {
    public static void main(String[] args) throws Exception
    {
        System.out.println("Enter integers please: ");
        Scanner cin = new Scanner(System.in);
        String in = cin.nextLine();

        System.out.println("Enter the File Name:");
        String fileName = cin.nextLine() + ".txt";
        String[] nums = in.split(" ");
        // get the numbers from the screen
```

```java
LinkedList<Integer> list = new LinkedList();

for (String num : nums) {
    try {
        list.add(Integer.parseInt(num));
    }catch (NumberFormatException ex)
    {
        throw new Exception("Let's try again, just integers this time.");
    }
}

DataAnalyzer data = new DataAnalyzer(list);
int Max = (Integer)data.FindMax();
int Min = (Integer)data.FindMin();
short Ave = (Short)data.FindAverage();

System.out.println("Max Number: " + Max);
System.out.println("Min Number: " + Min);
System.out.println("Ave Number: " + Ave);
```

main.java | DataAnalyzer.java

Source | History

```java
            System.out.println("Min Number: " + Min);
            System.out.println("Ave Number: " + Ave);

            // Save it into a file now

            BufferedWriter writer = null;
            FileWriter fileWriter = null;

            try
            {
                fileWriter = new FileWriter(fileName);
                writer = new BufferedWriter(fileWriter);
                System.out.println("Finished!");
            }catch (IOException ex)
            {
            }
            finally
            {
                try
                {
                    if(writer != null)
                    {
                        writer.close();;
                    }
                    if(fileWriter != null)
                    {
                        fileWriter.close();
                    }
                }catch (IOException e)
                {
                    e.printStackTrace();
                }
            }
        }
    }
```

Answer:

main.java | DataAnalyzer.java | test.txt.txt

Source | History

```
Max Number: 800
Min Number: 1
Ave Number: 171
Finished!
```

Output

Debugger Console | Homework_1 (run) #4

```
run:
Enter integers please:
1 50 6 80 90 800
Enter the File Name:
test
Max Number: 800
Min Number: 1
Ave Number: 171
Finished!
BUILD SUCCESSFUL (total time: 14 seconds)
```

4.

Answer the question and explain what happens without running the code:

What is the value of x after the following code is executed ?

I believe that the answer is 3, once all of the objects are created the conditions of the if statements are true. Then the operator "==" checks when two objects are equal; but in this case, they are not, which only means that g2==null is excited. Once the print out for q2 calls the method SayHello(), which produces a null reference and then the catch block gets the value ++ and finally increases by another 1.