Homework 7

**Name: Peter Dobbs**

**MU ID: 005803169**

**Date: 5 March, 2015**

**Homework Problem 1**

**Requirements**

Approximate the square root of an input number.

**Design**

*main*

Prompt the user to enter a double to root. Create the scanner and variables and assign the variable to the input double. Call the method “mySqrt” and display the approximation and the actual value, using Math.sqrt();. After the method has been used, close the scanner.

*mySqrt*

Use the equation “root = (root + (x/root))/2”, where root begins at one and x is the input double, in a loop to approximate the value of the root. Then return the root.

**Iterative Development Steps**

1. Import the scanner utility into the program and create a new class.
2. Create two methods, one main and one named “mySqrt.” Use the main method for assigning the scanner to a variable and creating and assigning variables to the input integers.
3. In the method “mySqrt” use the equation “root = (root + (x/root))/2”, where root begins at one and x is the input double, in a loop to approximate the value of the root. Then return the root.

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | 25 | 5 | 5 | Good code |

**Homework Problem 2**

**Requirements**

Write a program that displays a bar graph based on random input numbers between 1 and 4

**Design**

*main*

Enable the random number generator for numbers from 0 to 4. Create a loop that calls the countIt(n) method, where n is a random number, a million times.

*countIt()*

Use a switch statement to compile all values of 0, 1, 2, 3, and 4 in individual counters.

*graphIt()*

Use for loops to print out the appropriate bar graph. Use an array to adjust values scaled to less than 50.

**Iterative Development Steps**

1. Import the Random class and create a new class. In the main method enable the random number generator for numbers between 0.0 and 4.0.
2. Create a switch statement that compiles the appropriate numbers into counters
3. Use these values to create a bar graph of asterisks

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | N/A | c0: \*\*\*\*\*\*\*\*\*\*  c1: \*\*\*\*\*\*\*\*\*\*  c2: \*\*\*\*\*\*\*\*\*\*  c3: \*\*\*\*\*\*\*\*\*\*  c4: \*\*\*\*\*\*\*\*\*\* | c0: \*\*\*\*\*\*\*\*\*\*  c1: \*\*\*\*\*\*\*\*\*\*  c2: \*\*\*\*\*\*\*\*\*\*  c3: \*\*\*\*\*\*\*\*\*\*  c4: \*\*\*\*\*\*\*\*\*\* | Good Code |

**Homework Problem 3**

**Requirements**

Create a program that plays the “pick-up-sticks” game, aka. Nim.

**Design**

*main*

Use a random number generator to create an initial number of sticks as well as a value of sticks picked-up by the computer and a Scanner to allow the user to enter a value of sticks picked-up. Use these values as the basis for a game where the last player with a stick loses. A do-while loop where the computer and user’s number of pick-up sticks should continue to subtract from the initial number of straws.

**Iterative Development Steps**

1. Import the Scanner and Random classes and create a new class. In main, initialize the scanner and a random number generator and assign the input values to variables.
2. The random number for the computer’s turn subtracts from the initial random number and the user is allowed to input a value. This continues until one of the players is left with the last stick.
3. At that point one player is said to have won and the user is asked if he or she wants to play again. If yes, then the game starts over. If no, then the while loops ends.

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | 3  3  2  1 | unsure | You Win! | Good Code – random generator causes some confusion |

**References**

An Introduction to Programming Using Java, Anthony J Dos Reis.

Lauren Ernst, classmate