Lab Assignment 7

Kevin Burton

100228943

CPSC1150 - Section 1

Instructor: Hossein Darbandi

Exercise Title: Arrays Lab

Date submitted: June 30, 2014

Department: CPSC

**Program Arrays Lab**

File Name: Lab07.java

Purpose: To solve some problems using java arrays:

* + - Fill an array with random numbers.
    - Print out an array.
    - Reverse an array.
    - Print the sum of an array in an interesting way.
    - Fill an array with random numbers from a chosen min to a chosen max value.
    - Count the number of occurrences of a number in an array.
    - Create an array of characters from keyboard input.
    - Print out an array of characters, but replace spaces with newline characters.

Input: Will be done as the program runs, but currently the only input is a string to turn into an array of characters

Output: Various outputs:

* 10 digit array with random numbers from 1 – 10.
* The previous array reversed.
* The an array with numbers equal to the sum of the reversed array up to the counter/iterator
* 50 digit array with random numbers from 10 – 20
* The number of occurrences of each number in the previous array
* User inputted character array printed out so that spaces are replaces by newlines

Technical Information:

Compiler: Java SDK 1.7

Computer: Intel Celeron 2955U 1.4GHz, 2.00 GB of RAM

Language: Java

Source Code:

Attached

Program Logic (Pseudocode)

**reverseArray**

(definitions)

0.1 Input ← array passed to function

0.2 Output ← new array with same length as Input

0.3 Length(arr) ← length of an array

START

1. Count1 ← 0
2. Count2 ← length(input) – 1
3. While count2 >= 0
   1. Output[Count2] ← Input[Count1]
   2. Count1 ++
   3. Count2 –
4. Return Output

END

**sumArray**

(definitions)

* 1. Input ← array passed to function
  2. Input2 ← array passed to function (should be same length as Input, but full of 0’s)

START

1. For each number in Input (Count)
   1. Count2 ← 0
   2. For each number from 0 to Count (Count2)
      1. Sum += Input[Count2]
   3. Print Sum

END

**countArray**

(definitions)

0.1: Input ← Array passed to function

0.2: Number ← A single number in the input array

START

1. Min ← max int value (2^32 – 1)
2. Max ← min int value (-2^32)
3. Count ← 0
4. For each number in input
   1. If number < min
      1. Min ← Number
   2. If number > max
      1. Max ← Input
5. For number in range (Min, max)
   1. Count ← 0
   2. For item in Input
      1. If item equals number
         1. Count ++
   3. Print number and count

END

**printOneInLine**

(definitions)

0.1: Input ← Array passed to function

0.2: Item ← A single character in the input array

START

1. For each item in input
   1. If item equals ‘ ‘
      1. Print *newline*
   2. Else
      1. Print item

END

**Test Cases:**

Encode/Decode

FORMAT: Input – coding – method

Test Case 1:

Input: ABC – 4 - (encoding)

Returns: 010001010100011001000111

Test Case 2:

Input: ZXC – 9 - (encoding)

Returns: 010010110100100101001100

Test Case 3a:

Input: AXCV – 8 – (encoding)

Returns: 01001001010010000100101101000110

Test Case 3b:

Input: 01001001010010000100101101000110 – 8 – (decoding)

Returns: AXCV

Test Case 4a:

Input: ABC1297SDERSA#$% - 8 – (encoding)

Returns: 01001001010010100100101100111001001100100011100100110111010000110100110001001101010110100100001101001001001000110010010000100101

Test Case 4b:

Input: 01001001010010100100101100111001001100100011100100110111010000110100110001001101010110100100001101001001001000110010010000100101 – 8 – (decoding)

Returns: ABC12***1***7SDERSA#$%

*Not quite but pretty close… :/*