3305 Assignment 3 – Arrays & Linked Lists 100 pts Due 8/28 @ 11:59 pm

GENERAL SUBMISSION REQUIREMENTS

Note: Never hard-code test data in the test program, unless explicitly stated otherwise in the assignment. Always allow the user to enter the test data using a menu option

Upload all files individually as specified, not as zip files, to Assignments in D2L. Do not email files.

Make sure your program compiles, runs, and produces the correct output.

Ensure you have the correct file name(s), and author header, as specified in the Assignment.

Always use meaningful labels for prompts, inputs, and outputs.

Well documented programs use comments, indentation and whitespace as shown in text.

Objective: To reinforce the concept of arrays and linked lists. There are two parts to this assignment.

Part 1 – Linked List Iterator (Note, there are 3 sub parts to Part 1, 1.A, 1.B and 1.C) (50 pts)

Write a program that creates a linked list of integers, assigns integers to the linked list, prints a range of values in the list and eliminates duplicate numbers in the list. You may use the Java Library Linked List Class or you may write your own. Your program must have both of the following methods:

Part 1.A - First method is called printRange which has two parameters, x and y. printRange should write out all integers in the list that are between the first occurrence of x (inclusive) and the first occurrence of y (but not including y). You may assume integers can be compared for equality using ==

Use the following header for the method:

```
void printRange(int x, int y);
```

Use the following logic to develop printRange This makes the implementation easier.

- Do not sort or reorganize the data
- Print integers from x to y including x but not including y
- If there isn't an entry with a value equal to x, then print nothing
- If there isn't an entry after x, that has a value equal to y, then print the integers from x (inclusive) to the end of the list

- Print the values on one line separated by space.
- Put an end of line after the values are ALL printed.

Hint, your program will need to make multiple calls to printRange

For this program, you are [**hard coding**] creating your data in the program, and not entering it from the console. In your program create a linked list with the following members:

{1, 1, 2, 3, 3, 4, 4, 5, 6, 7}.

Review the sample output below. **You MUST USE** the values for x and y, shown, in bold below, as parameters in multiple calls to printRange to test the logic of your program.

Below is sample output. . // Comments

List [1,2,3,4,5,6,7] // **x = 2 y = 5**

prints 2 3 4 // prints list including x but not y

List [1,2,3,4,5,6,7] // **x = 2 y = 78**

prints 2 3 4 5 6 7 // prints list including x, no value =y so print to end

List [1,2,3,4,5,6,7] // **x = 2 y = 1**

prints 2 3 4 5 6 7 // prints list including x, no value =y so print to end

List [1,2,3,4,5,6,7] // **x = 8 y = 5**

// prints nothing

Part 1.B - The second method is called removeRepetitions. removeRepetitions steps through the linked list comparing numbers and when duplicate numbers are found, removes one of the duplicate numbers. Again, you may assume integers can be compared for equality using ==

Use the following header for the method removeRepetitions:

```
void removeRepetitions()
```

Here is a brief outline of an algorithm for removeRepetitions:

```
variable p steps through the list
for each number in the list, define a new variable q equal to p
While q is not the last number in the list
If the next number has data equal to the data in p, remove the next number
Otherwise move q to the next number
```

Part 1.C - Finally, write a test program to test both of these functions.

Do not forget to include author header in your file as shown, no header, no points!

```
// Name: <your name>
// Class: CS 3305/ put your section number after the /
// Term: Fall YYYY
// Instructor: Sharon Perry
// Assignment: 2 – Part 1 Iterator
```

DELIVERABLE INSTRUCTIONS – Part 1

Capture a **READABLE** screenshot(s) of your program output and paste into a word/pdf document. Readable means readable! Screenshots ***should not be an entire desktop*** – use some type of snipping tool. After your output screenshots, copy and paste the source code for your program into the word/pdf doc. Save doc as a file named LastName-A3-Part-1-Iterator.docx or .pdf. word. Last step is to upload everything to D2L, to clarify upload

1. word/pdf doc that contains output screenshot and copy/pasted source code; AND
2. your .java file that contains your source code.

Only your latest submission is kept so if you update a file and want to resubmit you must upload all files. Submit everything to the assignment submission folder in D2L by the due date posted in D2L.

No zip file or email submissions are accepted. Late penalties are in effect for this assignment.

Important Note: Code must be correctly running and produce correct results before being uploaded.

Late penalties of 10 % per day are in effect for this assignment.

Part 2 – Arrays - Guess The Capital of A State (50 pts)

Write a program that repeatedly prompts the user to guess the capital of a randomly displayed state.

Upon receiving the user input, the program reports whether the answer is correct. Your program must run a minimum of 5 times and display the total number of correct answers. The user's answer is not case-sensitive.

I have provided the list of states and capitals for you below and in a text file attached to the assignment.

```
{"Alabama", "Montgomery"},
{"Alaska", "Juneau"},
{"Arizona", "Phoenix"},
{"Arkansas", "Little Rock"},
{"California", "Sacramento"},
{"Colorado", "Denver"},
{"Connecticut", "Hartford"},
{"Delaware", "Dover"},
{"Florida", "Tallahassee"},
{"Georgia", "Atlanta"},
{"Hawaii", "Honolulu"},
{"Idaho", "Boise"},
{"Illinois", "Springfield"},
{"Maryland", "Annapolis"},
{"Minnesota", "Saint Paul"},
{"Iowa", "Des Moines"},
{"Maine", "Augusta"},
{"Kentucky", "Frankfort"},
{"Indiana", "Indianapolis"},
{"Kansas", "Topeka"},
{"Louisiana", "Baton Rouge"},
{"Oregon", "Salem"},
{"Oklahoma", "Oklahoma City"},
{"Ohio", "Columbus"},

{"North Carolina", "Raleigh"},
{"North Dakota", "Bismark"},
{"New York", "Albany"},
{"New Mexico", "Santa Fe"},
{"New Jersey", "Trenton"},
{"New Hampshire", "Concord"},
{"Nevada", "Carson City"},
{"Nebraska", "Lincoln"},
{"Montana", "Helena"},
{"Missouri", "Jefferson City"},
{"Mississippi", "Jackson"},
{"Massachusettes", "Boston"},
{"Michigan", "Lansing"},
{"Pennslyvania", "Harrisburg"},
{"Rhode Island", "Providence"},
{"South Carolina", "Columbia"},
{"South Dakota", "Pierre"},
{"Tennessee", "Nashville"},
{"Texas", "Austin"},
{"Utah", "Salt Lake City"},
{"Vermont", "Montpelier"},
{"Virginia", "Richmond"},
{"Washington", "Olympia"},
{"West Virginia", "Charleston"},
{"Wisconsin", "Madison"},
{"Wyoming", "Cheyenne"}
```

Do not forget to include author header in your file as shown, no header, no points!

```
// Name: <your name>
// Class: CS 3305/ put your section number after the /
// Term: Fall YYYY
// Instructor: Sharon Perry
// Assignment: 2 – Part 2 Capitals
```

DELIVERABLE INSTRUCTIONS – Part 2

Capture a **READABLE** screenshot(s) of your program output and paste into a word/pdf document. Readable means readable! Screenshots ***should not be an entire desktop*** – use some type of snipping tool. After your output screenshots, copy and paste the source code for your program into the word/pdf doc. Save doc as a file named LastName-A3-Part-2-Capitals.docx or .pdf. word. Last step is to upload everything to D2L, to clarify upload

1. word/pdf doc that contains output screenshot and copy/pasted source code; AND
2. your .java file that contains your source code.

Only your latest submission is kept so if you update a file and want to resubmit you must upload all files. Submit everything to the assignment submission folder in D2L by the due date posted in D2L.

No zip file or email submissions are accepted. Late penalties are in effect for this assignment.

Important Note: Code must be correctly running and produce correct results before being uploaded.

Late penalties of 10 % per day are in effect for this assignment.