Department of Electrical Engineering and Computer Science
Lassonde School of Engineering
York University, Winter 2017

CSE 2011: Assignment 1

Due Date - Monday, Feb 16, in class!!!

Question 1 **Big-O Analysis** [5 points]

Sort the following functions by increasing order of growth:

$$n \lg n \quad n^{\lg n} \quad (\lg n)^n \quad 2^{\frac{\lg n}{2}} \quad n^2$$

Question 2 **Running Time Analysis** [10 points]

Express the running time of the following program segment using big-θ notation. Show/justify your work.

```
int k=1, sum=0;
for (int i=0; i<n; i++) {
   k=2*k; }
for (int j=k; j>1; j=j/3) {
   sum++; }
```

Question 3 **Algorithm Design** [10 points]

Array A of size n contains all integers form 1 to (n+1) but one!

1) Assuming the array is sorted, propose a (best-running time) algorithm that finds/identifies the missing number, and analyze its worst-case running time in terms of big-O notation.

2) Assuming the array is NOT sorted, propose a (best-running time) algorithm that finds/identifies the missing number and analyze its worst-case running time in terms of big-O notation.

Question 4 **ADTs / Stacks** [10 points]

Using the regular Stack ADT, design an advanced Stack ADT for which getMinimum() method runs in O(1) time. Pushing and popping n elements to this Stack should each run in O(n) (i.e., you are not allowed to perform any sorting (e.g.) as a part of a different method). You are only allowed to use one additional data structures (e.g., another Stack) in your design.

Describe your solution in words and provide a brief pseudocode for the new push(), pop(), and getMinimum() methods.

Question 5 **Java Programming Project: On-line Message Assembler** [65 points]

When Bob wants to send Alice a message (M) on the Internet, he breaks M into n data packets, numbers the packets consecutively, and injects them into the network. When the packets arrive at Alice's computer, they are out of order, so Alice must sort the sequence of n packets in order before she is able to read the entire message.

You are asked to create a Java program, which would help Alice assemble Bob's messages. You can assume that for each message (M), the contents of n received packets, together with their respective sequence numbers, are temporarily stored in a file, as illustrated in Figure 1. An example of such a file, which can be used for test purposes, is available at:
http://www.cs.yorku.ca/course/2011/Mystery.txt

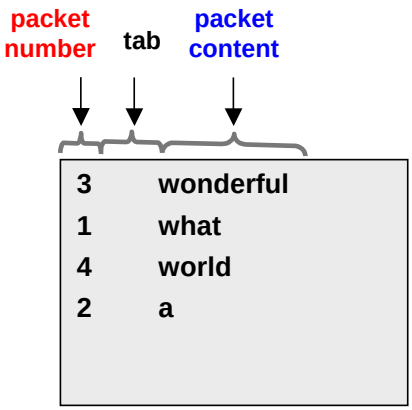

Figure 1 File content

Your program design should be based on the following guidelines:

(1) Create DLLNode[1] class, which will be used to store the content of each individual packet. The interface of this class is given below.

```
class DLLNode{
   DLLNode(int packetID, String packetContent, DLLNode prev, DLLNode next){}
   int getPacketID() { }
   String getPacketContent() { }
   DLLNode getNextNode() { }
```

[1] Here DLL stands for doubly linked list.

```
   DLLNode getPrevNode() { }
   void setNextNode(DLLNode n) { }
   void setPrevNode(DLLNode p) { }
}
```

(2) Create DLL class, which will be used to store the content of an entire file (i.e. message) – one line/packet per node. The outline of this class is given below. You are allowed to add new methods to this class, as needed.

```
class DLL {

   /* readAndAssemble performs the following operations:
      (1) Reads from file fileName (e.g. Mystery.txt) line by line
      (2) stores the content of each line in a DLLNode
      (3) places each DLLNode in DLL according to DLLNode's packetID number */
   void readAndAssemble(String fileName) { }

   /* printContent traverses DLL and prints out the content of its nodes so as to
      recreate the original message */
   void printContent() { }

}
```

(3) Create MessageAssembler class, which will contain main() method and act as a tester class. The outline of this class is given below. You are allowed to add new methods to this class, as needed.

```
public class MessageAssembler {

   public static void main(String[] args) {

            DLL myDLL = new DLL();
            myDLL.readAndAssemble("Mystery.txt");
            myDLL.printContent();

      }
}
```