The University of Saskatchewan

Saskatoon, Canada

Department of Computer Science

CMPT 280– Intermediate Data Structures and Algoirthms

Assignment 2

Date Due: February 2, 2024, 6:00pm

Total Marks: 41

General Instructions

- Assignments must be submitted using Canvas.
- Responses to written (non-programming) questions must be submitted in a PDF file, plain text file (`.txt`), Rich Text file (`.rtf`), or MS Word's `.doc` or `.docx` files. Digital images of handwritten pages are also acceptable, provided that they are **clearly** legible, and they are in either the JPEG or PNG image format.
- Programs must be written in Java.
- VERY IMPORTANT: Canvas is very fragile when it comes to submitting multiple files. We insist that you package all of the files for all questions for the entire assignment into a ZIP archive file. This can be done with a feature built into the Windows explorer (Windows), or with the zip terminal command (LINUX and Mac). We cannot accept any other archive formats. This means no tar, no gzip, no 7zip. Non-zip archives will not be graded. We will not grade assignments if these submission instructions are not followed.

Your Tasks

The questions on this assignment are about timing analysis and ADT specification. There is no programming on this assignment.

Question 1 ():

For each of the following functions, give the **tightest upper bound** chosen from among the usual simple functions listed in Section 3.5 of the course readings. Answers should be expressed in big-O notation.

(a) (1 point) $f_2(n) = 42n + 9\sqrt{n} + \log_2 n$

(b) (1 point) $f_3(n) = 4n^{0.7} + 6n \log_2 n + 280$

(c) (1 point) $f_1(n) = n \log_2 n + \log_2 n^2 + 280n$

Question 2 ():

Suppose the **exact** time required for an algorithm A in both the best and worst cases is given by the function

$$T_A(n) = \frac{1}{280}n^2 + 42 \log n + 12n^3 + 280\sqrt{n}$$

(a) (2 points) For each of the following statements, indicate whether the statement is true or false.

1. Algorithm A is $O(\log n)$
2. Algoirthm A is $O(n^2)$
3. Algoirthm A is $O(n^3)$
4. Algoirthm A is $O(2^n)$

(b) (1 point) Can the time complexity of this algorithm be expressed using big-Θ notation? If so, what is it?

Question 3 ():

If possible, simplify the following expressions. *Hint: See slide 11 of topic 3 of the lecture slides!*

(a) (1 point) $O(n^2) + O(\log n) + O(n \log n)$

(b) (1 point) $O(2^n) \cdot O(n^2)$

(c) (1 point) $42O(n \log n) + 18O(n)$

(d) (1 point) $O(n) + O(m)$ *(yes, that's an 'm', not a typo; note that m is independent of n)*

Question 4 ():

Consider the following Java code fragment:

```
1 // Print out all ordered pairs of numbers between 1 and n
2 for(i = 0; i <= n; i++) {
3    for(j = 0; j <= n; j++) {
4       System.out.println( i + ", " + j) ;
5    }
6 }
```

(a) (3 points) Use the *statement counting approach* to determine the exact number of statements that are executed when we run this code fragment as a function of `n`. Show all of your calculations.

(b) (1 point) Express the answer you obtained in part (a) in big-Θ notation (since the best and worst cases are the same – there is only one path of execution through this loop).

Question 5 ():

Consider the following pseudocode:

```
1  Algorithm roundRobinTournament(a)
2  This algorithm generates the list of matches that must be
3  played in a round-robin pirate-dueling tournament (a tournament where
4  each pirate duels each other pirate exactly once).
5
6  a is an array of strings containing names of pirates in the tournament
7
8  n = a.length
9  for i = 0 to n-1
10    for j = i+1 to n-1
11       print a[i] + " duels " + a[j] + ", Yarrr!"
```

Note: the pseudocode `for i = a to b` *means that the loop runs for all values of* `i` *between* `a` *and* `b`*, inclusive, that is, including the values* `a` *and* `b`*.*

(a) (6 points) Use the *statement counting approach* to determine the exact number of statements that are executed by this pseudocode as a function of `n`. Show all of your calculations.

(b) (1 point) Express the answer you obtained in part a) in big-Θ notation (since, again, the best and worst cases are the same).

Question 6 (3 points):

Using the active operation approach, determine the time complexity of the pseudocode in question 5. Show all your work and express your final answer in big-Θ notation.

Question 7 (6 points):

Consider the following pseudocode.

```
1  Algorithm multiSearch( data, target ):
2  data:  a list of arrays of integers; in each array the
3         integers are sorted in ascending order; the list
4         'data' has a cursor.
5  target: an integer
6
7  // Iterate over the arrays in the list 'data' using
8  // its cursor:
9  data.goFirst()
10 found = false
11 while( !data.after() and !found ) {
12    // search for integer 'target' in A
13    found = binarySearch(data.currentItem(), target)
14    data.goForth()
```

Using the active operation approach to timing analysis determine the time complexity of this pseudocode in the **worst case**. Assume that the list of arrays contains n arrays and that each array has exactly m items in it. Be sure to clearly identify the line that is the active operation. Show all your work and express your final answer in Big-O notation (because we are doing a worst-case analysis).

Question 8 (11 points):

A priority queue is a queue where a numeric priority is associated with each element. Access to elements that have been inserted into the queue is limited to inspection and removal of the elements with smallest and largest priority only. A priority queue may have multiple items that are of equal priority.

Give the ADT specification for a bounded priority queue using the specification method described in Topic 7 of the lecture notes. By "bounded", it is meant that the priority queue has a maximum capacity specified when it is created, and it can never contain more than that number of items.

Your specification must specify the following operations:

newPriorityQueue: make a new queue
insert: inserts an element with a certain priority
isEmpty: test if the queue is empty
isFull: test if the queue is full
maxItem: obtain the item in the queue with the highest priority
minItem: obtain the item in the queue with the lowest priority
deleteMax: remove from the queue the item with the highest priority
deleteAllMax: remove from the queue all items that are tied for the highest priority
deleteMin: remove from the queue the item with the lowest priority
frequency: obtain the number of times a certain item occurs in the queue (with **any** priority)

Files Provided

None.

What to Hand In

You must submit the following files:

assignment2.doc/docx/rtf/pdf/txt - your answers to questions 1 to 8. Acceptable file formats are Word (`.doc` or `.docx`), PDF (`.pdf`), rich text (`.rtf`), or plain text (`.txt`). Digital images of handwritten pages are also acceptable, provided that they are **clearly** legible and that they are in JPEG (`.jpg` or `.jpeg`) or PNG (`.png`) format, or they are embedded in a Word or PDF file. Other image formats are not accepted and will receive a grade of zero.

If you are submitting a single file, for example, a PDF or DOC containing answers to all questions, you can submit just the document without zipping it.

If you are submitting multiple files (e.g. multiple PNG or JPEG images), submit them in a ZIP file archive.