Semester one of academic year (2015—2016) of BJUT

Data Structures and Algorithms I

COMP2002J

Exam Instructions: Answer 2 of 3 parts

Honesty Pledge:

I have read and clearly understand the Examination Rules of Beijing University of Technology and University College Dublin and am aware of the Punishment for Violating the Rules of Beijing University of Technology and University College Dublin. I hereby promise to abide by the relevant rules and regulations by not giving or receiving any help during the exam. If caught violating the rules, I would accept the punishment thereof.

Pledger:______________ Class No:______________
BJUT Student ID:______________ UCD Student ID:______________

Notes:

The exam paper has 3 parts on 6 pages, with a full score of 100 points. You are required to use the given Examination Book only.

Instructions for Candidates:

Full marks will be awarded for complete answers to **All** questions.

Instructions for Invigilators:

Candidates are allowed to use non-programmable calculators during this examination.

Obtained Score

Part: 1 Queue and Complexity

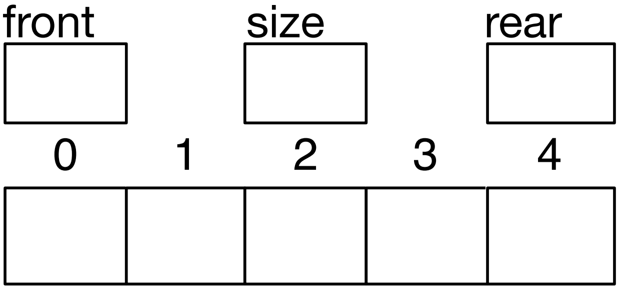
a. Give the definition of the queue abstract data type? List the operations provided, and the complexity of each operation in big O notation.

(6 Points)

b. Analyse the following operations performed on a newly created empty queue. Every time an item is dequeued, this item is printed. Give the output of these operations.
`enqueue(45), enqueue(23), enqueue(45), dequeue(), dequeue(), enqueue(27), enqueue(99), dequeue()`

(4 Points)

c. Copy the drawing below in your answer book and show the contents of the variables `front`, `rear`, `size` and the array `values` in the circular array based implementation of the Queue abstract data type after every third operation has been completed, there should be 3 diagrams in total.
`enqueue(15), enqueue(23), dequeue(), enqueue(99), enqueue(45), dequeue(), enqueue(67), enqueue(66), dequeue()`



(5 Points)

d. Based on the code below, write the code for the enqueue method in the circular array implementation of the queue abstract data type.

```
public class CircularQueue implements Queue {
  private int front;
        private int rear;
        private int size;
        private int[] values;
  ...
}
```

(10 Points)

e. Determine the complexity of the following piece of code in big O notation? Explain your answer.

```
int sum = 0;
for(int i = n; i > 0; i /= 2){
  sum += i;
}
```

(5 Points)

f. Determine the complexity of the following piece of code in big O notation? Explain your answer.

```
public int answer(int k) {
  for (int i = 0; i < k; ++i) {
    return k;
  }
}
```

(5 Points)

g. Order the following big O values from most efficient to least efficient: $O(n \log n), O(1), O(n^2), O(n)O(\log n), O(n^3), O(n!)$

(5 Points)

h. The Stack ADT has two methods push (to put something in) and pop (to take something out):
Which data of the following data structures would be the most efficient for implementing the Stack, a singly linked list or a doubly linked list? Explain your answer by talking about the complexity of the methods that would be used in push and pop

(10 Points)

(Total 50 Points)

Obtained Score

Part: 2 **Lists and Sorting**

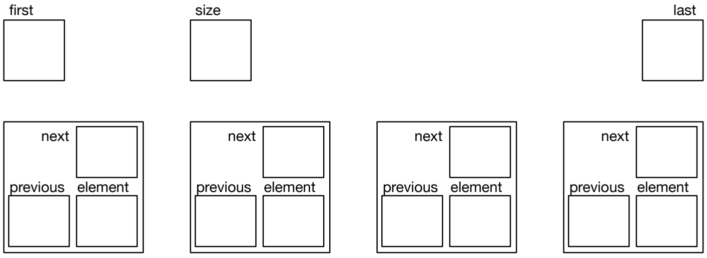
a. We have studied 3 implementations of the List abstract data type, which of the three is the most efficient? Explain your answer by comparing the complexity of some operations in the three implementations.

(5 Points)

b. Describe in English the steps involved in searching a doubly linked list to check if it contains an element X. What is the complexity of the operation in big O notation

(5 Points)

c. Draw a diagram similar to the one below. This diagram should show the state of a doubly linked list after the following operations have been completed.
The references should be drawn as arrows and the values should be shown
`addFirst(A), addLast(B), p = first(), addAfter(p, C), l = last(), addAfter(l,D), p = after(p), addBefore(n, E)`



(5 Points)

d. Based on the code below, write the code for the `addFirst` method for the doubly linked list implementation. This method should take a single parameter, a single object to be added to the list. Remember that the method should return a `Position`.

```
public class DoublyLinkedList implements List{
  private Node first;
  private Node last;
  private int size;
  ...
}
```

(10 Points)

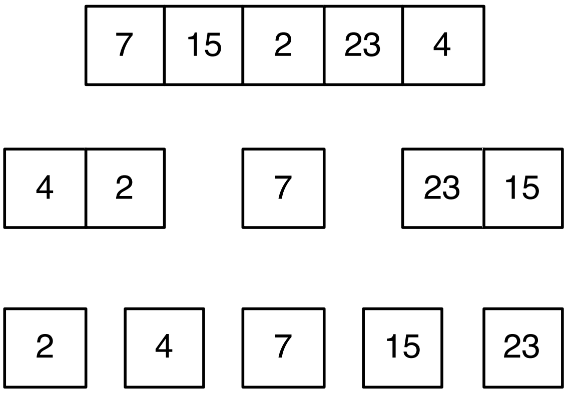
e. What is the complexity of the selection sort algorithm in big O notation?

(2 Points)

f. Explain in your own words how the mergesort algorithm functions. What is the complexity of this algorithm in big O notation?

(8 Points)

g. The following example shows the partitions created during each pass of the quicksort algorithm, where the first number in the partition is used as the pivot value. Apply the quicksort to the following array {2, 78, 7, 45, 9}. Draw a diagram showing the partitions and the values in the array. Explain the recursive calls of the algorithm and the partitions specified for each call.



(7 Points)

h. Describe the operation of radix sort. Illustrate this explanation with an example. Show the order of the array after each pass while sorting the following array 19, 29, 34, 78, 67, 58. The algorithm should use the radix 10.

(8 Points)
(Total 50 Points)

Obtained Score

Part: 3 **Maps**

a. What is a Map? List the and describe the operations of the Map ADT.

(10 Points)

b. Hash functions convert keys to integer values in the correct range. Describe the two basic mappings?

(4 Points)

c. The polynomial sum is a method to convert a sequence into a hash value. Describe how this process works. Calculate the hash code for the following string "abba" if 'a' = 97, 'b' = 98 and the polynomial used is 3.

(6 Points)

d. A hash table implementation for storing entries whose keys are integer values consists of an array of size 11. It uses the hash function $h(x) = (4x + 7) mod 11$. The collision strategy is linear probing. Show using a diagram, the state of the hash table after each of the following keys has been added. {45, 33, 23, 78, 29, 13, 54, 61}

(10 Points)

e. Based on the code below, write the `get` method for the array based implementation of the map using linear probing as a collision strategy. You should also fully explain the linear probing strategy for collision handling and comment on how it is implemented in the code.

```
public class HashMap implments Map{
  private Entry[] entries;
  private int size;
  ...
  public int hashCode(int k){
    return k % entries.length;
  }
}
```

(20 Points)

(Total 50 Points)