NATIONAL UNIVERSITY OF SINGAPORE

Special Semester, 2018/2019

TIC2001 Data Structure and Algorithm

Time Allowed: 2 Hours

INSTRUCTION TO CANDIDATES

1. This is NOT an open book assessment. You are allowed to bring one piece of A4 size cheat sheet only.
2. This assessment paper contains **EIGHT (8)** questions and comprises SEVEN **(7)** printed pages.
3. Answer *ALL* questions within the spaces provided in this booklet.
4. You are allowed to use the back of the paper but please remember to state "P.T.O."
5. *Cross out any draft* or otherwise we will mark the poorer answers.
6. Please write your student number below, but NOT your name.
7. We assume all array starts with index 0.

TIDINESS COUNTS!

We will deduct marks if your writing is too messy.

STUDENT NUMBER:_____

(This portion is for examiner's use only)

Question	Max. Marks	Score	Check
Q1	10		
Q2	6		
Q3	4		
Q4	6		
Q5	8		
Q6	8		
Q7	4		
Q8	4		
Total	50		

Question 1 [10 marks]

You are given 10 statements below. State if each statement is True (T) or False (F) by writing your answer in the box provided. A correct answer will give you 1 mark. An empty answer will give you zero, but ***a wrong answer will result in -1 mark***. (Lowest mark for this question is 0)

T	It is correct to say that the running time for Quicksort is $O(n^3)$
T	We can perform Quicksort on a linked list of intergers and the time complexity is the same as performing on an array of intergers.
T	We can use a linked list of linked lists of nodes to model a graph.
T	In C++, a parent class can have more than one children classes.
F	If the ADT of a set only have three operations: adding an item to a set, union of two sets, and checking if an item exists in a set, the best data structure to implement the set ADT is by trees.
F	The worst running time to find the successor of an item in a hash table is O(n) if there are n items inserted in the hash table.
T	The shortest edge in a graph is always in the minimal spanning tree of that graph if no two edges have the same weight.
F	If a graph has negative weighted edges, Dijkstra algorithm will always fail to find the shortest paths of the graph from a single source.
F	If a maxheap is stored in an array, we can always convert a maxheap to a minheap by reversing the array, and vice versa. For example, if an array A store a maxheap of [10,9,8,7] then reversing A as [7,8,9,10] will be a minheap.
F	For any single deletion of AVL tree, we need either zero, one or two rotations to balance the tree after deletion.

Question 2 (6 marks)

What is the time complexity of calling the functions on the right column in terms of n for n > 0? Give your answer in the Big O notation. The function **`doSomething(m)`** will have a time complexity of O(m) depending on the input m.

`int f(int n) {`
`  for(int i=1;i<(n/2);i++)`
`  for(double j=0;j<100;j += (100.0/n))`
`  doSomething(n/2);`
`  return 0`
`}`
	Time complexity of `f(n)` = $O(n^3)$

`int f(int n) {`
`  if (n<100000)`
`  return 0;`
`  for (int i=0;i<(n*n);i++)`
`  return f(n-1);`
`}`
	Time complexity of `f(n)` = O(n)

`void f(double n) {`
`  if (n<1) return 0;`
`  doSomething(n);`
`  for (int i=0;i<10;i++)`
`  f(n/10);`
`}`
	Time complexity of `f(n)` = O(n log n)

Question 3 (4 marks)

Here is the array after one round of pivoting by Quicksort

10	9	50	6	52	61	55	65	53	77

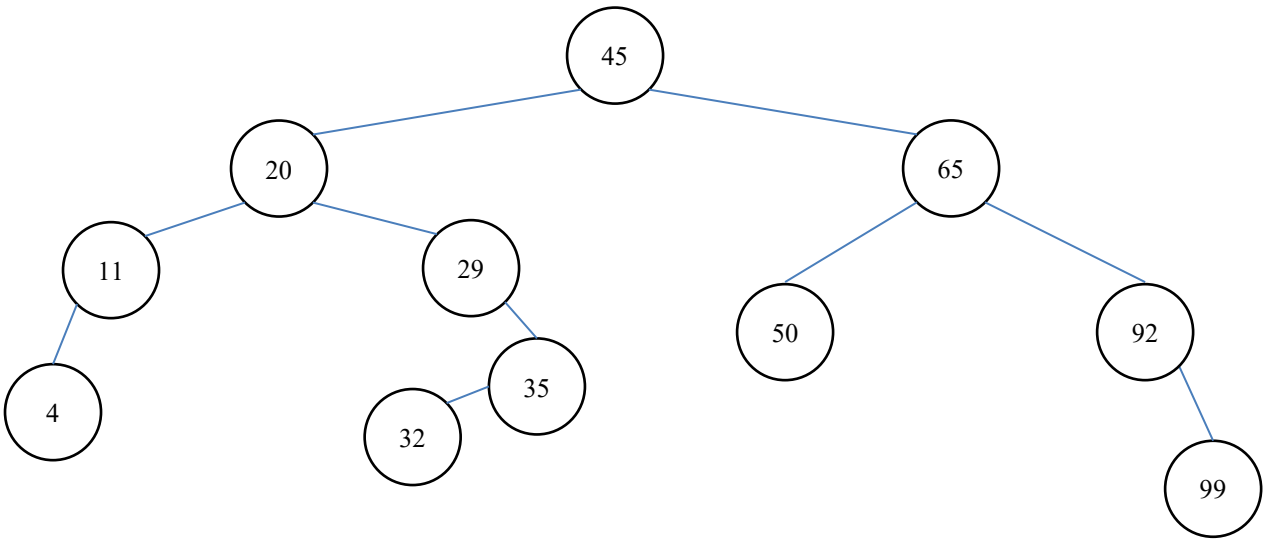
Which number was the pivot for the last round? Answer: ____52_____

Perform one more round of pivoting on the left and the right arrays of the pivot above by using the first element as the pivot of each array. Circle the two pivots.

9	6	**10**	50	52	55	53	**61**	65	77

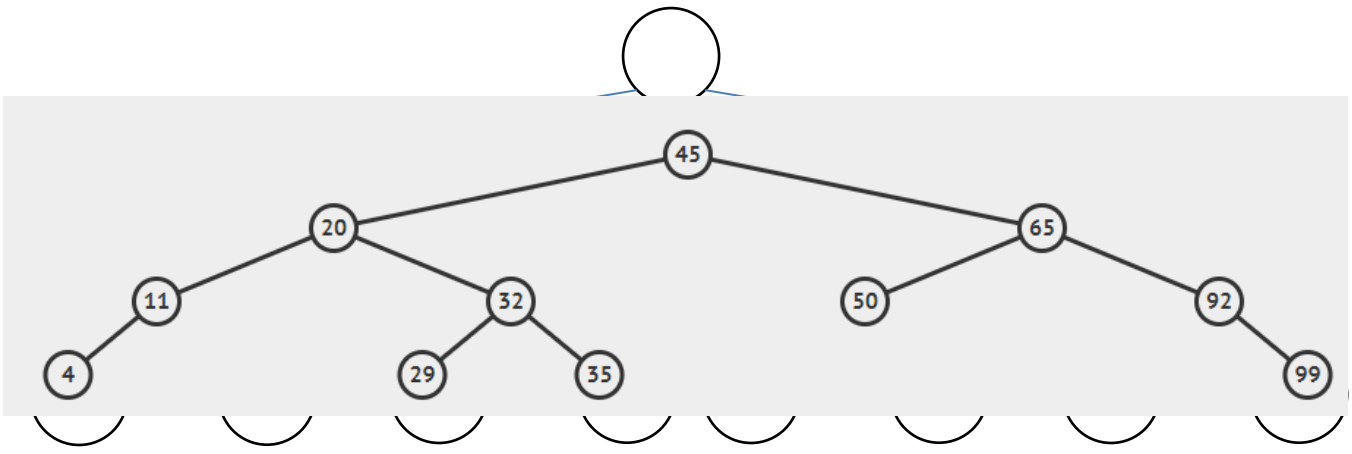
Question 4 (6 marks)

Here is an AVL tree after inserting one new node **before** balancing.



Which node is the newly inserted node? Answer: _____32_____

Perform the necessary rotations according to AVL trees and show the final balanced tree below. Add/remove any bubble(s) if necessary.



Question 5 (8 marks) MAXHEAP

You are given an array A of n = 10 unsorted integers as below (array index starts from 0):

6	5	1	2	3	7	8	9	4	10

Follow the algorithm in the lecture notes (On the right).

```
for (int i=(n-1); i>=0; i--)
    bubbleDown(i, A);
```

Your job is to heapify the unsorted array into a **Maxheap**. The first round, $i = 9$ is done for you as an example. Note:

- If the array doesn't change after one bubbleDown(), you can write "same" instead of copying the whole array.
- **Circle** the numbers that has been swapped/moved for that iteration that are different from the line above it.

After `bubbleDown(9,A)`:

6	5	1	2	3	7	8	9	4	10

After `bubbleDown(8,A)`:

same									

After `bubbleDown(7,A)`:

same									

```
Index of parent(x) = floor((x-1)/2)
Index of leftchild(x) = 2x+1
Index of rightchild(x) = 2x+2
```

After `bubbleDown(6,A)`:

same									

After `bubbleDown(5,A)`:

same									

After `bubbleDown(4,A)`:

6	5	1	2	10	7	8	9	4	3

After `bubbleDown(3,A)`:

6	5	1	9	10	7	8	2	4	3

After `bubbleDown(2,A)`:

6	5	8	9	10	7	1	2	4	3

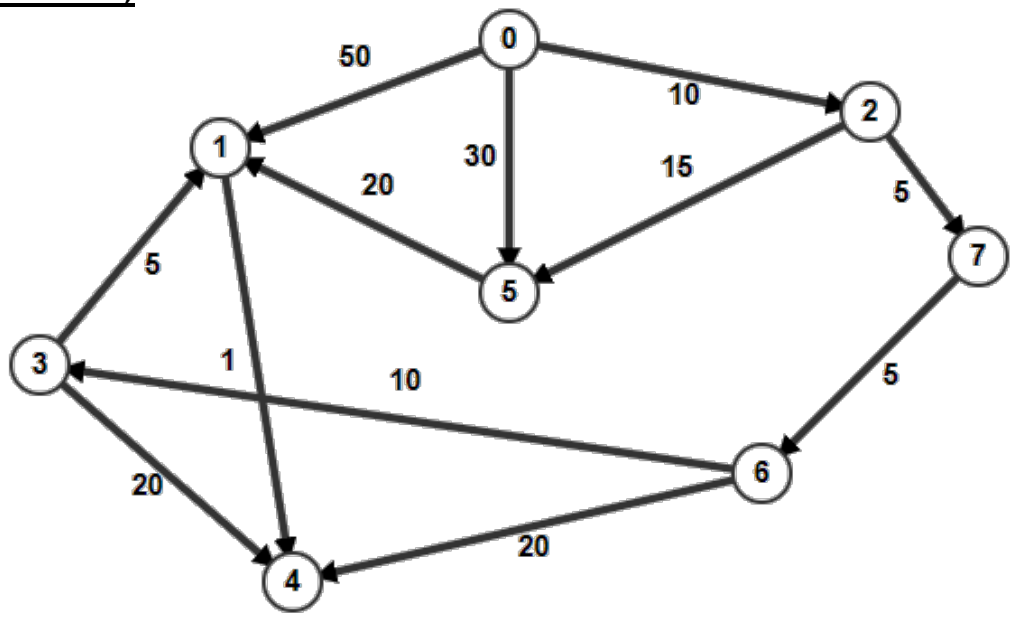
After `bubbleDown(1,A)`:

6	10	8	9	5	7	1	2	4	3

After `bubbleDown(0,A)`:

10	9	8	6	5	7	1	2	4	3

Question 6 (8 marks)



Perform Dijkstra Algorithm to this graph to find all the shortest distance from the node 0. Each of the following table is a priority queue sorted by the shortest estimated distance, δ(0,v), in ascending order. If two nodes have the same shortest estimated distance, extract the one with smaller index first.

Node v	δ(0,v)
0	0
1	∞
2	∞
3	∞
4	∞
5	∞
6	∞
7	∞

Extract 0 →

Node v	δ(0,v)
2	10
5	30
1	50
4	∞
3	∞
6	∞
7	∞

Extract ___ →

Node v	δ(0,v)
7	15
5	25
1	50
3	∞
6	∞
4	∞

Extract ___ →

Node v	δ(0,v)
6	20
5	25
1	50
3	∞
4	∞

Extract ___ →

Node v	δ(0,v)
5	25
3	30
4	40
1	50

Extract ___ →

Node v	δ(0,v)
3	30
4	40
1	45

Extract ___ →

Node v	δ(0,v)
1	35
4	40

Extract ___ →

Node v	δ(0,v)
4	36

The shortest distance of each node from 0 is:

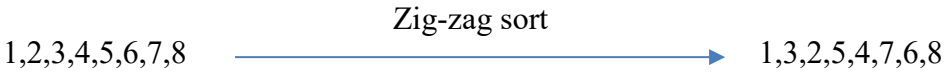
Node:	1	2	3	4	5	6	7
Distance:	35	10	30	36	25	20	15

Question 7 (4 marks)

You are given an array A of *n* unique integers. Design the most efficient algorithm to perform "zig-zag" sorting. Namely, for a number A[i] with index i:

- If i is even, the number(s) next to A[i] are larger than A[i]
- Otherwise, the number(s) next to A[i] are smaller than A[i]

For example:



Describe your algorithm here. You may write pseudo code also.

Starting from i = 0 to n-2, check if A[i] and A[i+1] satisfy the requirement. If not, swap them

What is the time complexity of your algorithm for an array with n elements?

O(n)

Question 8 (4 marks)

The SMART city project wants to make every building "smart" and put a high performance computer in each building to help managing a building. All the computers need to be connected with each other with a brand new network and the cost of the network cable is proportional to the direct physical distance between two computers. We need all the computers to be connected and minimize the cost in the same time.

However, for security reasons, there are a few number of pairs of computers needed to be connected in a specific way. Namely, some pairs must be connected directly. Making it worse, the cost of these secure connection cable is double the price of a normal cable. However, the good news is, we know that these special connections alone will not create any cycle and each computer has at most one special connection with another computer. Describe how to compute the layout of the network connection to minimize the overall cost of connecting every computer. And how do you calculate the final cost.

Construct a complete graph with every edge with Euclidean distance but 0 weight if the connection is special. Compute the MST. Finally add in the cost of the special connections.

(Not required. Worst cast is O(n^2) edges because of all nC2 connections that give O(n^2 log n). Could be improved by constraint Delaunay Triangulation by O(n log n))

- End of Paper -