Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

Faculty of Engineering, Mathematics & Science

School of Computer Science and Statistics

Integrated Computer Science Programme **Semester 1, 2020-21**
BA (Mod) CSLL
BA (Mod) Business & Computing
BA MSISS
Year 2 Annual Examinations

CS2011: Algorithms and Data Structures I

online exam **2hr duration**

Dr. Vasileios Koutavas

Instructions to Candidates:

- Answer ALL questions.

Materials permitted for this examination:

- Open books exam.

Question 1 [0 marks]

In some of the following questions you will be asked to provide an answer which depends on the following numbers. Please calculate and **write down these numbers in the answer box on blackboard and on a piece of paper in front of you**.

(a) **SID**: this is your TCD student number. It should be an 8-digit number.

(b) $\mathbf{SID^2}$: this is **SID** squared; it should be a 15-digit number.

(c) $\mathbf{d_1}, \mathbf{d_2}, \mathbf{d_3}, \mathbf{d_4}, \mathbf{d_5}, \mathbf{d_6}, \mathbf{d_7}, \mathbf{d_8}, \mathbf{d_9}, \mathbf{d_{10}}, \mathbf{d_{11}}, \mathbf{d_{12}}, \mathbf{d_{13}}, \mathbf{d_{14}}, \mathbf{d_{15}}$: these are the fifteen digits of $\mathbf{SID^2}$, from left to right.

(d) $\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3}, \mathbf{u_4}, \mathbf{u_5}, \mathbf{u_6}, \mathbf{u_7}, \mathbf{u_8}, \mathbf{u_9}, \mathbf{u_{10}}$: these are the first 10 **unique** single-digit numbers of the d_i sequence, from left to right. If there are less than 10 unique numbers in your d_i sequence, you should complete **the remaining** u_i sequence with single-digit numbers in ascending order. In the end, the u_i sequence should not contain duplicate numbers.

Question 2 [20 marks]

Consider the Abstract Data Type (ADT) `StackQueue` with the following API:

`public class StackQueue`

`StackQueue()`	Creates a `StackQueue` object.
`void add(int n)`	Adds `n` to the ADT.
`int pop()`	Removes and returns an element from the ADT **following the LIFO principle**.
`int dequeue()`	Removes and returns an element from the ADT **following the FIFO principle**.
`int size()`	returns the number of elements currently in the ADT.

The following program is a client of the `StackQueue` ADT:

```
public class Client {
  public static void main(String[] args)
  {
    if (args.length < 10) return;
    Integer[] input = new Integer[10];
    for (int i = 0; i < 10; i++)
      input[i] = Integer.parseInt(args[i]);

    StackQueue buffer = new StackQueue();
    for (int i = 0; i < input.length; i++)
    {
      buffer.add(input[i]);
      if (input[i] % 3 == 0)
        System.out.println(buffer.pop());
      else if (input[i] % 2 == 0)
        System.out.println(buffer.dequeue());
    }
    while (buffer.size() > 0) {
      System.out.println(buffer.dequeue());
    }
  }
}
```

Give the sequence of numbers printed by the command

`java Client u1 u2 u3 u4 u5 u6 u7 u8 u9 u10`

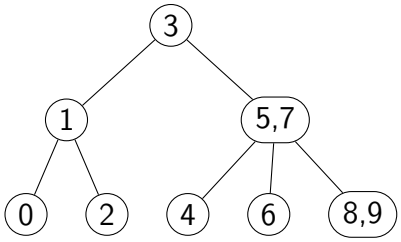
where `u`i is the corresponding single-digit number of the u_i sequence from Question 1 (d). *[20 marks]*

Question 3 [20 marks]

Give the logical representation of the **2-3 tree** obtained by inserting the sequence $u_1, ..., u_{10}$ from Question 1 (d) to an empty 2-3 tree.

Show the logical representation of the tree after every node split.

For example inserting the sequence $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$ produces the tree:



[20 marks]

Question 4 [10 marks]

Give the logical representation of the **left-leaning red-black tree** that corresponds to the 2-3 tree you provided in your answer of Question 3. This is the red-black tree obtained by inserting the sequence $u_1, ..., u_{10}$ to the empty tree.

[10 marks]

Question 5 [20 marks]

Consider points in a 2-dimensional space written as (x, y), where x and y are the x- and y-coordinates of the point, respectively. Recall that a **2d-tree** is a binary search tree that divides the space using the x- and y- coordinates, alternatively, by level. In the root node the space is divided by the x-coordinate of the point in the node.

Show the logical representation of the 2d-tree after we insert the following points in order:

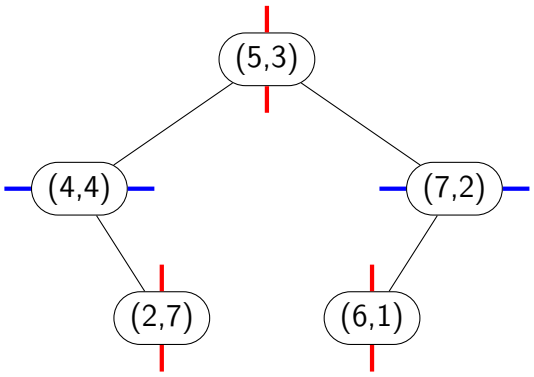
$$(u_1, u_{10}),\ (u_2, u_9),\ (u_3, u_8),\ (u_4, u_7),\ (u_5, u_6),\ (u_6, u_5),\ (u_7, u_4),\ (u_8, u_3)$$

where ui is the corresponding single-digit number of the u_i sequence from Question 1 (d).

For example, if we insert the points

$$(5, 3),\ (4, 4),\ (2, 7),\ (7, 2),\ (6, 1)$$

we obtain the 2d-tree:



[20 marks]

Question 6 [10 marks]

When studying the performance of algorithms we can use three types of asymptotic notation: O, Θ, Ω. Explain the intuitive difference between Ω and O. Use examples in your explanation. *[10 marks]*

Question 7 [20 marks]

Consider the min priority queue ADT which has the following API:

`public class MinPQ<Key extends Comparable<Key>>`	
`MinPQ(int k)`	create an empty priority queue of size `k`.
`void insert(Key v)`	insert a key into the priority queue.
`Key delMin()`	return and remove the smallest key.
`int size()`	number of keys in the priority queue.

The following code finds the k'th largest element in the input array, and returns **`null`** if no such number exists.

```
public Integer findKthLargest(int[] nums, int k)
{
  if (nums.legth < k) return null;
  MinPQ<Integer> heap = new MinPQ<Integer>(k+1);
  for (int i = 0; i < nums.length; i++)
  {
    heap.add(nums[i]);
    if (heap.size() > k)
      heap.delMin();
  }
  return heap.delMin();
}
```

Analyse the performance of this code with respect to time and space. You are free to use any one of the analyses that have been covered in the module. You should clearly identify the analysis you use and sufficiently explain your answer.

[20 marks]