Midterm. 2020-10-22,

100 minutes (9:00 – 10:40)

Question 1.

There is a 2-dimensional array `arr` of size n by n; its elements are integer numbers (each number contains no more than n digits in its decimal representation).

You are given this C++ code with function `main()` calling function `fun()`.

```
1  int fun(int row) {
2    int count = 0;
3    for (int col=0; col<n; col++) {
4      while (arr[row][col]>0) {
5        arr[row][col] /= 2;
6        count++;
7      }
8    }
9    return count;
10 }
11
12
13 int main() {
14   for (int row=0; row<n; row++) {
15     cout << fun(row);
16   }
17 }
```

Analyze the time complexity of this algorithm "inside out":

(A) Estimate with some $O(g(n))$ the time spent on a single iteration of the loop in Lines 5-6. (How fast one can run these lines just once.)

(B) Estimate with some $O(g(n))$ the time spent on the loop Lines 4-7 (for one specific pair of row,col).

(C) Estimate with some $O(g(n))$ the time spent in the outer loop on Lines 3-8. (And also in a single call of `fun(...)`.)

(D) Estimate the time spent in the loop on Lines 14-16.

Note. We use integer division in this algorithm; for example `7/2` equals `3` and `1/2` equals `0`. So it will eventually stop.

Question 2.

Definition of Big-O:
Real-valued function with natural arguments $f(n)$ is in $O(g(n))$ iff there exist $C, n_0 > 0$ such that

$$|f(n)| \le C \cdot |g(n)|,$$

whenever $n > n_0$.

(A) Use the definition of Big-O Notation to prove or to disprove that $f(n) = 100n^2 + \frac{1}{4}n^3$ is in $O(n^3)$.

(B) Use the definition of Big-O Notation to prove or to disprove that $f(n) = (\log_2 n)^2$ is in $O(\log_2 n)$.

The reasoning should either provide the examples of C, n_0 that satisfy the definition for any n; or a method that for any given C, n_0 finds find n such that the definition is not satisfied.

Question 3.

Consider the Queue implementation from `https://bit.ly/35n7bKd`.

Denote a, b, c to be the last 3 digits of your Student ID, and compute the following numbers:

- $F = ((a + b + c) \mod 3) + 2$
- `x1` $= (a + b + c) \mod 10$
- `x2` $= ((a + b) \cdot 2) \mod 10$
- `x3` $= ((b + c) \cdot 3) \mod 10$
- `x4` $= ((c + a) \cdot 7) \mod 10$

The queue Q is implemented as an array of size $N = 6$; its elements have indices from $\{0, 1, 2, 3, 4, 5\}$.

Initially the queue parameters are these:
`Q.front` $= F$,
`Q.length` $= 4$,
`Q.size` $= 6$.
And the content of the array is the following:

i	0	1	2	3	4	5
array[i]	1	3	5	7	9	11

Somebody runs the following code on this queue:

```
Q.enqueue(x1)
Q.enqueue(x2)
Q.dequeue()
Q.dequeue()
// show the state of Q
Q.enqueue(x3)
Q.enqueue(x4)
Q.dequeue()
// show the state of Q
```

After Line 4 (and at the very end) show the current state of the queue `Q`. The state should display the content of the array and also the values of `Q.front` and `Q.length`.
You can use shading, if it helps to visualize the array cells that are not currently used by your queue.
Note. Painting something gray is not required (since front/length indicate the state of your queue anyway). But painting cells gray may be helpful, if you want to visualize where your queue has the useful values (and what is some old garbage – you can shade it over).

Question 4.

Introduction. Binary trees are often represented as arrays (where the array starts with the root node; followed by all the other nodes, displayed layer by layer. If any child of a node in this tree is missing, it is replaced by Λ (capital Lambda denoting an empty tree) in the array. Once we reach the last non-empty node in the tree, this is the last element of the array. For example, the binary tree shown in this picture:

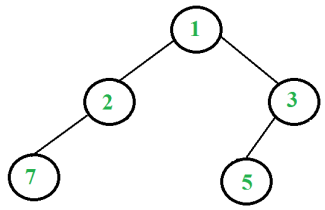

Figure 1: Example binary tree.

is represented by the following array:

$$\texttt{int a[]} = \{1, 2, 3, 7, \Lambda, 5\};$$

Problem. Assume that you have a binary tree that is represented by the following array:

$$\texttt{int a[]} = \{1, 2, 4, \texttt{a}, \Lambda, \Lambda, 6, \texttt{b}, \Lambda, \Lambda, \Lambda, \Lambda, \Lambda, \Lambda, \texttt{c}\}; \quad (1)$$

Values a, b, c are the last three digits taken from your Student ID.
Note. In the original the array contained a mistake (it had four Λ instead of six; but this was wrong, it does not correspond to any tree).

(A) Draw the binary tree represented by the array 1 in your answer. The tree should look nice: Draw left children to the left (and right children to the right) of their parents. Nodes on the same levels should be aligned.

(B) What is the number of internal nodes in this tree? The number of leaves in this tree?

(C) List the vertices of this tree in the post-order traversal order.
Note. You only show real nodes in the post-order sequence (all Λ are just technical symbols indicating absence of nodes; they are not part of the tree).

(D) Write pseudo-code for an algorithm GETPARENT(i) that receives the index i of some node in this array, returns the index of the parent of this node (or -1, if the node has no parent). All indices i are zero-based (in an array of length 10, $i \in \{0, \ldots, 9\}$).

(E) Assume that there is a different array (representing another binary tree) which does not contain any Λ values; all values there represent some nodes. Describe the property such trees must satisfy.