Dept. of Computer Science
Data Structures (CS210)
1st Semester, 2019-20
Major Exam 1

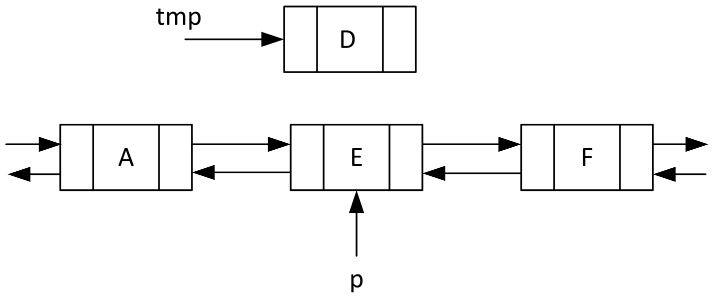
Time: 60 minutes **Marks: 30 marks**

Question 1. (4 + 3 + 3 = 10 marks)

(a) Write the method removeLast for the singly-linked list discussed in the class. The method removes the last node in the list. The singly-linked list maintains the size, the references to the head and tail of the list. [4 marks]

```
void removeLast(){
```

(b) Write statements to insert the node `tmp` after the node p in the doubly-linked list. Assume that each node has a `next` and `prev` data members to hold references to next and previous nodes in the list. [3 marks]



(c) Write a method for the circular singly-linked class that displays all the data elements in the list. Assume that the size of the list is not being maintained. [3 marks]

Question 2. (2 + 3 + 1 + 1 + 6 = 13 marks)

(a) How many times is the **`count++`** executed in the following code segment? [2 marks]

```
count = 0;
for(int i = 1; i < 10; i++) {
    count++;
}
```

(b) What is the runtime for the following code snippet? Give the runtime as an equation $T(n)$ based on estimation and the Big-Oh notation. [3 marks]

```
1. s = 0;
2. for (int i = 0; i < n; i++)
3.   for (int j = i; j < n; j++)
4.       s = s + 1;
```

(c) Which of the following two algorithms has a better time complexity:
(i) algorithm A with a step count function $2^{100} + \log n^{100}$
(ii) algorithm B with step count function $n + 2\log n$. [1 mark]

(d) Which of the following two algorithms has a better time complexity:
(i) algorithm A with a growth rate $O(n^2)$
(ii) algorithm B with a growth rate $O(nlogn)$. [1 mark]

(e) For what values of c and n_0 , the function $f(n)$ is $O(n^3)$, $g(n)$ is $O(n^2 \log(n))$.([6 marks]

(i) $f(n) = 4n^3 + 6n^2 + 2n + 1$

(ii) $g(n) = n^2 \log(10n^4 + 7) - 3n$

Question 3 (7 marks)

(a) Write a recursive method to sum the odd-integer-values in an array. What is the best-case and worst-case time complexity of the method? For example for an array [1, 2, 3, 4, 5, 6, 7], the sum of all odd-elements is 16. [3.5 marks]

(b) Show the trace for the following recursive method; estimate the run-time complexity as given an appropriate Big-Oh notation. For tracing use the top-level call: **`recur(32)`** [3.5 marks]

```
int recur (int n) {
   if (n == 1)
       return 0;
   return 1 + recur(n / 2);
}
```