DS221: Take-home Midterm Exam *(Data Structures and Algos.)*

150 Points

- You can use any textbook, class slides/notes/recordings or Internet resources to learn topics related to these questions. But you must solve and type the answers by yourself.
- You must not copy-paste or otherwise write verbatim solutions from external sources. Provide citation to any external sources or concepts used.
- Typed answers (e.g., using Word or Latex) are strongly encouraged. You can also submit hand-written answers; there will be a risk of losing points if we face difficulty while understanding hand-writing.
- Specify your name and IISc email address on top. Number the pages
- You may ask questions using the course Teams channel for this module.
- **Keep your answers concise, and to the point.** Once you are able to figure out correct answer to a question, think how to communicate the answer concisely before typing.
- You should upload your submission as a single pdf file before the due date. Deadline extension will not be granted.

1. A *Set abstract data structure* is used to store a unique collection of items, i.e., there are no duplicate items in the data structure. Provide the pseudo code for implementing the following functions for a Set data structure using any suitable backing data structure, such as an array, linked representation, etc. Justify your design choices. [20 points]

```
class Set {
      // Initializes the Set with an optional initial capacity
      public Set(int capacity);

      // Adds an item to the set if it did not already exist. Returns true
      if the item was added, false otherwise.
      public boolean add(int item);

      // Returns true if the given item exists in the set, and false
      otherwise.
      public boolean exists(int item);

      // Deletes the item if it was present in the set. Returns true if
      the item was deleted, false otherwise.
      public boolean delete(int item);
}
```

2. You are given a stack containing *n* distinct integers. Write a pseudocode to sort the stack in an ascending order such that the smallest items are on the top. You may use an additional temporary stack, but you may not copy the elements into any other data structure (such as an array). The stack supports the following operations: **push**, **pop**, **peek** and **isEmpty**. [15 points]

3. A *Graph* abstract data structure may be implemented either as an *Adjacency Matrix* or an *Adjacency list*. [5+15 points]

    a. What are the pros and cons of using Adjacency Matrix vs. Adjacency list.

b. Provide a complete pseudocode for checking presence of a cycle in a given graph using either one of these two data structures. The input graph is undirected and can have one or more connected components.

4. A *Hash Table* (or Hash Map) is an implementation of the *Dictionary* abstract data structure. [20 + 5 points]
   a. What are the best, worst, and expected time complexities for the **lookup**(key) operation of a hash table which uses linear probing, has *b* buckets, and uses (*key* mod *b*) as the hash function? Justify your answers. You can make the following assumptions:
      a. Keys in the table are uniformly randomly distributed in range [0, *R*], *R*>>*b*
      b. Count of keys saved in hash table is < *b*.
   b. What are the types of applications for which the Hash Table data structure is good for and bad for?

5. Arrange the following functions in ascending order of growth rate. That is, if function *g(n)* immediately follows function *f(n)* in your list, then it should be the case that *f(n)* is *O(g(n))*. [15 points]

   $f_1(n) = 10^n$
   $f_2(n) = n^{1/3}$
   $f_3(n) = n^n$
   $f_4(n) = log_2 n$
   $f_5(n) = 2^{\sqrt{log_2 n}}$

6. Contrast the pros and cons of performing *complexity analysis* and *empirical analysis* on an algorithm/application. [10 points]

7. Give the pseudo-code for the *in-order traversal* of a *binary tree* without using recursion. Hint: You may use a stack. [10 points]

8. What is the expected output of the following PySpark pseudocode? Explain with an example. [15 points]

```
marksRDD = sc.textFile("marks.csv");
courseRDD = marksRDD.filter(lambda x : x.split(",")[1] == "DS221");
pairRDD = courseRDD.map(lambda y :
    ( y.split(",")[0], float(y.split(",")[2]) ) );
gradesRDD = pairRDD.map(lambda k,v:
    (k, "A") if v > 80 else
        ((k, "B") if v > 70 else
            ((k, "C") if v > 60 else (k, "D"))
        )
    );

gradesRDD.collect();
```

   The input file marks.csv is of CSV file type with the following columns present: "studentID, courseID, marks". There is no header row.

9. Write an algorithm as pseudocode to print all the distinct ways of arranging eight queens on an 8x8 chessboard so that none of them share the same row, column, or diagonal. ("diagonal" means all diagonals, and not just the ones that bisect the board). Each output printed can be 8 entries, where each entry is (x,y) where x denotes the row and y denotes the column of the queen placed. [20 points]