DATA STRUCTURES & ALGORITHMS 1

BATCH – B [FRIDAY FEBRUARY 22, 2019: 3:30 PM – 6:30 PM]

LAB ASSIGNMENT – 5 CODE:assign05

NOTES:

1. Please carefully read all assignments and there is **no choice**.
2. **Use the template for this assignment**
3. Each problem in this assignment has to be answered in the **same c file**.
4. Create a .c file following the **file name convention**:
    a. If your roll number is 'abc' and assignment code is 'assignXX'. Then use the following file name convention as follows: 'abc-assignXX.c'
    b. For example, if the roll number is 92 and assignment code is assign01, then the file name should be 092-assign01.c
    c. Strictly follow the file name convention. When you are ready, submit the solution via google classroom.
5. **Follow variable and function naming conventions**
    a. except for variables in for-loop, none of the other variables should be a single character.
    b. The variable names and function names should indicate what they are storing/computing. For this assignment, we have given you some of the variable names and function names to use. They are highlighted as function_name or variable_name
6. Indentation improves readability. Please pick an indentation style and **indent your code** appropriately.
7. When in doubt about naming or style conventions, consult the following link: https://users.ece.cmu.edu/~eno/coding/CCodingStandard.html

---

Create a Structure **'student'** with the following details

1. Student name
2. Rollno (int is fine)
3. CGPA
4. City-code (int)
5. A pointer to 'neighbor-student' (struct) instance (we will use this pointer to point to another student who is from the same city as the student)

PROBLEMS [Total Marks: 20]:

1. [Marks: 5] Do the following
    a. Use 'ifndef' to 'define' a constant TOTAL_CITIES with value 5. From this point onwards you should only use this constant in your program.

b. Use ‘Typedef’ to the ‘structure student’ as ‘Student’, to make it easier for you to use in other places. From this point onwards you should only use ‘Student’ in your program.
c. Declare a global array ‘g_students_city’ to store **an array of pointers** to ‘Student’. (Please keep in mind the index of the array corresponds to different cities. For example, the index ‘0’ is for ‘Hyderabad’, etc.). The size of g_students_city is TOTAL_CITIES.

2. [Marks: 5] Write a function ‘populate_student_details’ for populating the ‘g_students_city’ array with values provided by the user (via scanf).
   a. While entering student details, (say) the user enters the city-code as ‘0’ for a student. Then:
      i. if there are no elements in the 0th position of the array, then just populate a pointer to the new student instance in that slot.
      ii. if there is already an element in the 0th position of the array, then traverse to the end of the linked-list (via neighbor-pointer) and attach the new student instance to the end.

      (Hint: you can write an elegant solution to this problem using a recursive function)
   b. Make sure the memory for Student instances are allocated in **the Heap** (use malloc).

3. [Marks: 5] implement the following
   a. Write a **recursive** function ‘print_students_neighbors’ which takes as input a pointer (say: sptr) and prints the details of the student pointed by ‘sptr’ and then proceeds to print the details of the neighbor student (via recursion).
   b. Write a function 'print_students_from_city’ which takes as input an int(city code) and prints all students from that city. This function should use 'print_students_neighbors’

4. [Marks: 5] Write a function ‘are_neighbors’ which takes as input two student roll_no and then returns true, if the students are neighbors, false otherwise.
