DATA STRUCTURES & ALGORITHMS 1

BATCH – B [MONDAY February 03, 2020: 3:00 PM – 6:00 PM]

LAB ASSIGNMENT – 3 CODE:assign03

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 assign03, then the file name should be 092-assign03.c
    c. Strictly follow the file name convention. When you are ready, submit the solution via google classroom.
5. Follow 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.

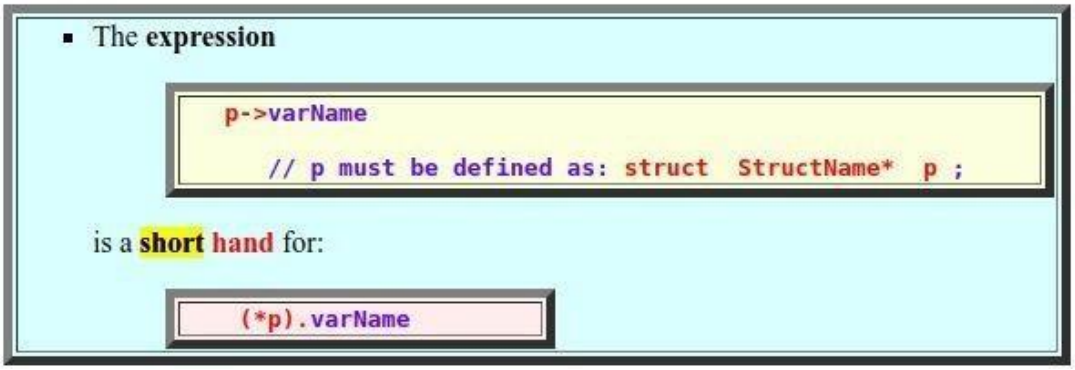
PROBLEM INSTRUCTIONS:

For the following problems write functions which satisfy the following:

1. The functions **should not have a return statement** (hence its return type should be void).
2. **All** the **arguments** to the functions should either be **pointers or void**
3. Do **not use global or static** variables(**except maybe for one employee pointer variable 'first-employee'**)
4. Please use the '->' shorthand dereference+access operator wherever necessary

The "->" operator: *short hand* for the expression "(*p)."

- The **-> operator**:



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

1. Employee ID (unique)
2. Employee Name
3. Address
4. Department
5. A pointer to next 'employee' (struct) instance (we will use this pointer to point to the next employee)
6. A pointer to previous 'employee' (struct) instance (we will use this pointer to point to the previous employee)

PROBLEMS [Total Marks: 20]:

1. [Marks: 4] Write a function which gets details of an employee from the user and creates an instance of the structure in **the Heap**. Its ok to initialize all 'next-employee' to NULL.

2. [Marks: 4] Write a function that takes as input two struct instances: employee1 and employee2. It populates the *next-employee of employee1 to point to employee2 (i.e the address of employee2 is stored in *next-employee of employee1) and *previous-employee of employee2 to point to employee1 (i.e the address of employee1 is stored in *previous-employee of employee2)

3. [Marks: 4] Write a function that takes one struct instance as input and prints the content clearly. It uses the *previous-employee to fetch and print the previous employee instance. The function continues until it reaches a 'first employee'. You must use **recursion** for implementing this.

4. [Marks: 8] Use all of the above functions to achieve the following.
   a. Ask user for no-of employees: 'n'
   b. Create 'n' employee instances, properly chain them using the next-employee and previous-employee pointers
   c. The 'first-employee' pointer should always point to the first employee
      i. You might need to modify the function you wrote for problem-1: IF the first-employee is null

the new instance becomes the first employee

ELSE

The new instance is created
First employee's next-employee now points to new-instance and new instance's previous-employee points to first employee

Please store the solutions. Future assignments might ask you to improve upon this solution

---