Data Structures and Algorithms Lab

02. Strengthening the Basics

Lab Code: 17ECSP201 **Lab No:** 02 **Semester:** III

Date: 17 Aug, 2017 **Batch:** C1

Theme: Code. Be a Code. Do a Code.

Objective: Goofing around the learnt basics

Below listed are tasks which you will carry out in a team of two. I know the codes are handful but that's why you have got two hands!

Task 01:

Below listed are few statements about Pointers. You need to prove them by demonstrating through a C program. Pick any 05 out of given 07. The blue colored ones are compulsory.

- Pointer is a variable which holds the address of another variable
- **A globally declared pointer is automatically initialized to NULL by compiler where as locally declared is not**
- A pointer variable cannot be divided by a constant or a variable
- Two pointers cannot be multiplied or divided
- **When we increment a pointer it gets incremented by pointer data-type number of bytes**
- Modifying the address of constant pointer is not allowed
- Typecast a void pointer to integer pointer **[Marks: 50]**

Task 02:

Write a program to generate cyclic words for a given word.

In main function: accept a string from the user of length 04 and generate all cyclic words. For example if user enters "care", then cyclic words are:

arec
reca
ecar
care **[Marks: 50]**

Task 03:

Collect an integer array from the user. Expand the same array elements into an array where every third number is sum of the first two numbers. Don't use the generated number in any calculation.

Example:
If user enters first array as {2, 5, 23, 45, 10, 7, 22, 3}
Then generate the second array as
{2, 5, 7, 23, 45, 68, 10, 7, 17, 22, 3, 25}

If the user enters odd numbers of elements then display an error message. While collecting input from the user, collect them sequentially without skipping any index. The number of array elements can be collected from the user initially. **[Marks: 70]**

Task 04:

Accept two strings and 'n' value from user. Then using pointer operations implement the function:
strncmp(str1, str2, n) - compare the first 'n' characters of str1 and str1.

- Return 0 if both are equal
- Return positive value if str1 > str2
- Return negative value if str2> str1 **[Marks: 70]**

Task 05:

Pointer 01: Hey, hello… Have I seen you somewhere? You look very familiar to me.
Pointer 02: eww! aaan??
Pointer 01: Why are you holding all the waste in your hand? Why don't you throw them?
Pointer 02: Who is this talking?? Why don't you have any shape?
Pointer 01: Me and my fate! And that's how my existence is! I can take any shape.
(Both are disturbed and confused. Just like the way you are now.)

Pointer 01: You look funny!
Pointer 02: You too. Very funny!

Pointer 01: Who cares? Let's party. Hop in.
Pointer 02: Dude!

Which of the following statement would be true with respect to above conversation??

A. Pointer 01 is NULL Pointer and Pointer 02 is not actually a pointer
B. Pointer 01 is a void pointer and Pointer 02 is a NULL pointer
C. Pointer 01 and Pointer 02 are both NULL Pointers
D. Pointer 01 is void pointer and Pointer 02 is dangling pointer
E. Pointer 01 is dangling pointer and Pointer 02 is a NULL pointer
F. Pointer 01 is void pointer and Pointer 02 is NULL pointer
G. Pointer 01 is NULL pointer and Pointer 02 is dangling pointer
H. Really, Pointers can talk??
I. Both Pointer 01 and Pointer 02 are dangling pointers
J. Pointer 01 is normal pointer and Pointer 02 is NULL pointer **[Marks: 20]**

Task 06:

The local train station provides the daily train departure details. It has a registry maintained having the details of train name, number, source, destination, departure time and reach time. Provide software solution for the local train station system. Use the below mentioned structures:

```
struct time
{
  int hh;
  int mm;
  int ss;
};

struct train_details
{
  char train_name[20];
  int train_number;
  char source[20];
  char destination[20];
  struct time departure_time;
  struct time reach_time;
};
```

Support your program with following options:

- Add a new train details
- Provide all train details to passenger
- Search based on source and destination. This search should also give the total journey time, if there is a match. **[Marks: 100]**

Task 07:

Your seniors have challenged you to be ready for the first match of the season. Their team is already ready and they are waiting for you to form a team and fix the fixtures. One of your batch mates also plans to automate everything to keep the match smooth. Here is what you are supposed to do to help out:

First finalize the match details. Following structures can be used. The input for below structures has to be taken from the user clearly indicating the purpose.

```
struct date
{
   int dd;
   int mm;
   int yy;
};

struct time
{
   int hh;
   int mm;
   int ss;
};

struct stadium
{
   char name[20];
   char city[20];
   int total_seats;
};

struct match_schedule
{
   struct date d;
   struct time t;
   struct stadium st;
};
```

Next are the details of umpire, team and the players. Statically initialize the data for all the 03 following structures. There has to be 03 umpire data, 01 team data and 40 player's data.

```
struct umpire
{
   char name[20];
   int age;
   char years_of_experience;
};

struct team
{
   char team_name[30];
   char coach_name[20];
   int team_id;
   int total_players;
};

struct player
{
  char name[20];
  int age;
  char city[20];
  int matches_played;
  int total_score;
};
```

Now there is a final structure which has to be output to the challenged senior team.

```
struct match
{
   struct match_schedule ms;
   struct umpire u;
   struct team t;
   struct player list[13];
};
```

While building the structure 'match', consider the following:

- The first structure member 'ms' is given by the user
- Selecting one best umpire out of 3 based on highest years of experience

- Select top 13 players based on average score (Average score = total_score / matches_played)
- Don't select any player aged below 17 or above 27
- Finally output all the details neatly and clearly to the user. **[Marks: 150]**

**** May The Force Be With You ****