Objective:

- It will also help you to feel easy with keyword 'struct'.
- You will get to know the basics of struct variable passing by value/reference.

Task 1: Focus on defining struct members → *must* do it before lecture no. 5

***Also remember to use character array to store strings. You are not allowed to use string data type in this regard.**

A. Look at the following structure declaration.

```
struct Point
{
    int x;
    int y;
};
```

Write statements that

1. Define a Point structure variable named center
2. Assign 12 to the x member of center
3. Assign 7 to the y member of center
4. Display the contents of the x and y members of center

B. Look at the following structure declaration.

```
struct FullName
{
    char lastName[50];
    char middleName[50];
    char firstName[50];
};
```

Write statements that

1. Define a FullName structure variable named info
2. Assign your last, middle, and first name to the members of the info variable
3. Display the contents of the members of the info variable

C. Weather Statistics

Write a program that uses a structure to store the following weather data for a particular month:

City
Country
Total Rainfall
High Temperature
Low Temperature
Average Temperature

The program should create a structure to store the above information by taking input from user and display it on console.

D. Corporate Sales Data

Write a program that uses a structure to store the following data on a company division:

Division Name (such as East, West, North, or South)

First-Quarter Sales

Second-Quarter Sales

Third-Quarter Sales

Fourth-Quarter Sales

Total Annual Sales

Average Quarterly Sales

The program should use four variables of this structure. Each variable should represent one of the following corporate divisions: East, West, North, and South. The user should be asked for the four quarters sales figures for each division. Each division's total and average sales should be calculated and stored in the appropriate member of each structure variable. These figures should then be displayed on the screen.

Task 2: Storing Rational Numbers → *must* do it before lecture no. 5
As discussed in lecture: Define all the following functions for the following Rational struct

```
struct Rational
{
        int numerator;
        int denominator;
};
```

Functions to implement:

2.1. void inputRational(Rational &)
This function takes input from user for numerator and denominator and stores it in received Rational variable.

2.2. void printRational(Rational)
This function prints on screen the received Rational variable.

2.3. Rational addRational(Rational a, Rational b)
This function returns a Rational variable, which is the addition of two received rational variables.

2.4. Rational diffRational(Rational a, Rational b)
This function returns a Rational variable, which is the difference of two received rational variables.

2.5. Rational divRational(Rational a, Rational b)
This function returns a Rational variable, which is the division of two received rational variables.

2.6. void reduce(Rational * a)
This function modifies the received rational variables by reducing the numerator and denominator. For Example, if the received variable has 12/30 then this function change it to 2/5.

Note: All function of Rational must make sure to store the final result of rational in reduced form.