# Sheet 7 (*Structures*)

## Class Work

1- Define struct named "**Point**" that represents a point in the 2D space (representing the **x** and the **y** position in the X-Y plane).
   a) Write a function that takes a point as a parameter, and prints the x-value and y-value of this point on the screen.
   b) Write a function that takes a point as a parameter, and returns its norm.
      norm of a point (p) : ‖p‖ = (x^2 + y^2) ^(-1/2)
   c) Write a function that takes two points as parameters, and returns their sum.
   d) Write a program that reads two points from the user and use the above functions to:
      i) Print the two points.
      ii) Print the norm of each point.
      iii) Print their sum.

2- Define struct named "**Student**" that represents the ID (4 digits only) and marks (out of 100) of a student.
   a) Write a function that reads the data of **5** students and store them in an array of the struct "**student**".
   b) Write a function that prints the students' data on the screen.
   c) Write a function that prints the ID of the student with maximum marks.

## Home Work

1- Declare a structure "**Employee**" that contains information about employees in a company. These data are:
   - Name (30 characters).
   - Date of Birth: (hint: struct of: day, month and year).
   - Salary
   a) Write a function that reads the data of **10** employees and store them in an array of the struct "**Employee**".
   b) Write a function that prints the data of the employee with maximum salary.
   c) Write a function that prints the data of the youngest employee.

2- Define struct named "**Student**" that represents the data of a student:
   - **Name** (30 character),
   - **ID** (10 digits)
   - **GPA**

   Write the following functions:
   a) Read the data of **10** students and store them in an array of the struct "**student**".
   b) Read an **ID** from the user and print the data of that student.
   c) Print the average **GPA** of the students.
   d) Print the data of the students with **GPA** above the average **GPA**.
   e) Print the students in ascending order of names.
   f) Print the students in descending order of **GPA**.

3- Define a type "**Rect**" for rectangles that are parallel to the axes in a Cartesian coordinate system. Represent a rectangle by its lower left and upper right endpoints using the struct **Point** you defined in **class work**.

   a) Implement the function **getlength** that receives a struct of type "**Rect**" and computes its length.
   b) Implement the function **rectPeri** that receives a struct of type "**Rect**" and computes its perimeter.
   c) Implement the function **rectArea** that receives a struct of type "**Rect**" and computes its area.
   d) Implement the function **rectcomp** that receives two structs of type "**Rect**" and returns:
      - "**1**" if the first rectangle has larger area than the second one.
      - "**0**" if the two rectangles have the same area.
      - "**-1**" if the first rectangle has smaller area than the second one.
   e) Implement the function **rectcont** that receives two structs of type "**Rect**" and returns "**1**" if the second rectangle is totally contained inside the first one, returns "**0**" otherwise.
   f) Implement the function **rectperf** that receives a struct of type "**Rect**" and returns "**1**" if the rectangle is perfect square (the length equals the width), returns "**0**" otherwise.
   g) Implement the function **rectorg** that receives a struct of type "**Rect**" and returns "**1**" if one of the rectangle vertices is on the origin **( 0 , 0 )**, returns "**0**" otherwise.