



School of Media And Information Technology  
Department of IST  
Program: CNT

**CMPE 1666-ICA 12- Sorting lists of structs using QuickSort**

In this assignment, you are going to adapt the quick-sort algorithm to sort a list of structs. You are going to create a simple list of structs from a provided data table to test your algorithm then construct a longer list of structs from data read from the provided text file. Once you are confident that the sorting algorithm is working properly, you'll apply it to the longer list created from the file.

The provided text file contains 1000 lines, each line containing the **employee id** and **salary** of an employee.

Start by creating a form-based application with the controls shown below. The form has 2 list boxes, 4 buttons, a pair of radio buttons, a read-only text box, a label for Drag & Drop, plus a few other labels accompanying the controls.

The screenshot shows a Windows application window titled "ICA08-Sorting Structs". The window has a light gray background and a blue border. It is divided into two main sections: "Unsorted Data" on the left and "Sorted Data" on the right, each with a large white list box. In the center, there are several controls: a "Display Unsorted List" button, a "Clear Unsorted Listbox" button, two radio buttons labeled "Provided List" (selected) and "File Data", a "Sort" button, a "Clear Sorted Listbox" button, and a blue "Drag And Drop File Here" button. At the bottom right, there is a label "Time Taken (Elapsed Ticks):" followed by a small white text box.

In the Form class, define an **Employee** struct type having, as members, an **employee id** and **salary**. s.

The application must declare and create 2 lists of Employee structs. One will contain structs built from data given below. The second one will contain structs built from data provided in the file. The 2 lists must also be member variables of the Form1 class.

Adapt the **Quick-Sort** it to sort employee structs in order of employee id.

For easy testing during the development of your code, construct the list of employee structs from the values given in the table below. The easiest way to create the list will be to create 2 arrays, initializing them respectively with the values of the employee ids and salaries given, then iterate through the arrays to create each struct and add it to the list.

Employee ID	Salary
28	4500
53	2800
12	1900
18	3100
8	7000
2	3500
19	2200
57	2800
62	2850
34	3150
23	4000
14	4500
48	6000
35	7200
55	3700
22	2100
26	2450
15	2500
7	3250
9	3700
32	3800
43	4200
41	4100
51	3900

The form load event must cause the creation of a list of Employees from the data provided in the above table.

When the user drags and drops the provided file onto the “Drag And Drop” label, each line of the data from the file must be used to create an **Employee** struct, which must then be added to the second list. Note that each line from the file contains an employee id and a salary. So you will want to split the line, convert to the required type (using TryParse()), then create the struct.

When the user clicks on the “Display Unsorted List” button, if the “Provided List” radio button is checked, the left list box will display the unsorted data from the small list (created from the data provided in the

table). If the “File Data” radio button is checked, the data in the longer list, created from the file will be displayed. The displayed data in both cases will be of the form **employeeid: Salary**

When the user clicks on the “Sort” button, the program will use the adapted sorting method to sort the list in ascending order of employee id. Again the list that will be sorted will be determined by the radio button checked. The sorted data will then be displayed in the right list box. The sorting time must be calculated (using a Stopwatch object) and displayed in the read-only textbox.

### Sample runs:

1. Unsorted and sorted versions of the small list (created from the table data)

ICA08-Sorting Structs

**Unsorted Data**

28:	4500
53:	2800
12:	1900
18:	3100
8:	7000
2:	3500
19:	2200
57:	2800
62:	2850
34:	3150
23:	4000
14:	4500
48:	6000
35:	7200
55:	3700
22:	2100
26:	2450
15:	2500
7:	3250
9:	3700
32:	3800
43:	4200
41:	4100
51:	3900

**Sorted Data**

2:	3500
7:	3250
8:	7000
9:	3700
12:	1900
14:	4500
15:	2500
18:	3100
19:	2200
22:	2100
23:	4000
26:	2450
28:	4500
32:	3800
34:	3150
35:	7200
41:	4100
43:	4200
48:	6000
51:	3900
53:	2800
55:	3700
57:	2800
62:	2850

Time Taken (Elapsed Ticks): 3860

2. File data, unsorted and sorted list

ICA08-Sorting Structs

**Unsorted Data**

3793:	4840
2488:	3860
1382:	3030
3633:	4720
1554:	3160
3052:	4280
2635:	3970
2897:	4170
3853:	4890
3698:	4770
25:	2010
2577:	3930
94:	2070
1078:	2800
2176:	3630
2406:	3800
1928:	3440
2020:	3510
2468:	3850
1823:	3360
3635:	4720
1639:	3220
2754:	4060
60:	2040
3735:	4800
1269:	2950
3990:	4990
2405:	3800
3087:	4310
480:	2360
248:	2180

**Sorted Data**

2:	3500
7:	3250
8:	7000
9:	3700
12:	1900
14:	4500
15:	2500
18:	3100
19:	2200
22:	2100
23:	4000
26:	2450
28:	4500
32:	3800
34:	3150
35:	7200
41:	4100
43:	4200
48:	6000
51:	3900
53:	2800
55:	3700
57:	2800
62:	2850
2:	4580
24:	2640
25:	2010
31:	2520
33:	4710
36:	4380
40:	2020

Time Taken (Elapsed Ticks): 123281

Rubric: Max Marks: 30

Item	Max Marks	Penalty
UI	4	Unprofessional layout: -2 No proper tab order: -2 Using default control names: -3
Struct Properly created	4	
Unsorted data properly displayed in left List box	4	Wrong Format: -2
Provided table Data properly sorted and displayed in right list box	6	
Drag and Drop working as required. Data loaded in left listbox	4	
File data sorted and displayed	6	
Execution Time Displayed	2	
Documentation: Programmer Block Appropriate Variable Names Program Properly commented		Missing components of documentation: -1 to -6

**This ICA has to use lists of structs and Quick-Sort. Otherwise it will be marked as 0..**