



School of Applied Sciences And Technology

Department of IST

Program: CNT

CMPE 1666-Lab02- Creating and Sorting lists of structs

In this lab, you are going to adapt the sorting algorithms (discussed in class) 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 similar but longer list from data read from a 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 100 lines, each line containing the employee id, first names and last names of an employee. The data items on each line (i.e for each employee) are separated from one another by a comma (",").

Start by creating a form-based application with the controls shown below. The form has 2 DataGridView components, a number of buttons, a pair of radio buttons, a read-only text box plus a few labels accompanying the controls. The default radio button must be the **"Provided List"** one. To each DataGridView, add 3 columns as shown and set the SortMode for each column to **Not Sortable**. The Form must also contain an **OpenFileDialog**. The "Open File Through OFD" button must initially be disabled.

ICA08-Sorting Structs

Unsorted Data

Employee Id	First Name	Last Name
-------------	------------	-----------

Sorted Data

Employee ID	First Name	Last Name
-------------	------------	-----------

Display Unsorted List

Clear Unsorted Data Grid

Data Source

☒ Provided List

☐ File Data

Sort By EmployeeID

Sort By Last Names

Clear Sorted Data Grid

Open File Through OFD

Time Taken (Elapsed Ticks):

In the Form class, declare an **Employee** struct type having, as members, an **employee id**, **first name** and **last name**. Do not include any method within the struct.

To the form class add a method **CreateEmployee()** that has, as parameters, an employee id, a first name and a last name. It creates and returns an Employee struct.

The application must declare and initialize 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 be member variables of the Form1 class.

To the application add the following sorting methods:

- A method that sorts a List of **Employee** on ascending order of employee ids, using the **Selection Sort** technique.
- A method that sorts a List of **Employee** in ascending alphabetical order of last names, using the **Insertion Sort** technique. Don't worry about the order between employees having the same last name.

For easy testing during the development of your code, construct a first List of Employee structs from the values given in the table below. The easiest way to create the List will be to create 3 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	First Name	Last Name
28	Emily	Johnson
53	Michael	Smith
12	Olivia	Williams
18	Daniel	Brown
8	Sophia	Jones
2	Ethan	Davis
19	Ava	Miller
57	Benjamin	Wilson
62	Isabella	Moore
34	Jacob	Taylor
23	Mia	Anderson
14	William	Thomas
48	Emma	Jackson
35	Alexander	Moore
55	Charlotte	Harris
22	James	Martin
26	Amelia	Thompson
15	Logan	Garcia
7	Harper	Martinez
9	Elijah	Robinson
32	Grace	Smith
43	Noah	Rodriguez
41	Lily	Lewis
51	Lucas	Clark

The form load event must cause the creation of a List of Employees from the data provided in the above table. The List must also be displayed in the “Unsorted Data” DataGridView.

If the “**File Data**” radio button is checked, the “Open File through OFD” button must be enabled. The user must be able to open a file by dragging and dropping it onto the “**Unsorted Data**” DataGridView or by clicking on the “**Open File Through OFD**” button which opens a file through the Open File Dialog (You need to implement both).

In each case, you will open a file in which each line of data will be used to create an **Employee** struct. Your program must read the file, create the **Employee** structs and add them to the second List variable of your Form class. Note that each line from the file contains an employee id, a first name and a last name. So you will want to split the line, convert to the required type (using TryParse()) where applicable, then create the struct. The “**Unsorted Data**” DataGridView must be cleared and loaded with the data from this new List.

At any time, when the user clicks on the “**Display Unsorted List**” button, if the “**Provided List**” radio button is checked, the left DataGridView will display the unsorted data from the small list (created from the table data). If the “**File Data**” radio button is checked, the data in the longer list, created from the file, will be displayed.

When the user clicks on the “**Sort By Employee Id**” or the “**Sort By Employee Names**” button, the program will use the corresponding sorting method to sort the List accordingly. 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 DataGridView. The sorting time must be calculated (using a Stopwatch object) and displayed in the read-only textbox.

Also Implement the 2 buttons for clearing the DataGridViews.

You may find the following info about DataGridViews helpful:

- To Add/Edit columns use the property Columns (under Misc. in the property pane).
- To Add a row of Data in the DataGridView, you’ll need to create a variable of type **DataGridViewRow**. You will then clone **Row[0]** of your **DataGridView** control and assign it to your row variable.

Eg. DataGridViewRow row = (DataGridViewRow)UI_UnsortedData_DGV.Rows[0].Clone();

- A row created this way will have an array Cells representing the columns of your DataGridView. You can then assign values to the value property of these cells. **Eg. row.Cells[0].Value=**
- You finally add the row to the property Rows of your **DataGridView** control.

Eg. UI_UnsortedData_DGV.Rows.Add(row);

Sample runs:

1. Unsorted List (from table data) and List sorted on employee Id.

The screenshot shows the ICA08-Sorting Structs application window. It features two data grids: 'Unsorted Data' on the left and 'Sorted Data' on the right. The 'Unsorted Data' grid lists employees with IDs 28, 53, 12, 18, 8, 2, 19, 57, 62, 34, 23, 14, 48, 35, 55, and 22. The 'Sorted Data' grid shows the same employees sorted by their Employee ID, with IDs 2, 7, 8, 9, 12, 14, 15, 18, 19, 22, 23, 26, 28, 32, 34, and 35. The 'Data Source' is set to 'Provided List'. The 'Sort By' is set to 'EmployeeID'. The 'Time Taken (Elapsed Ticks)' is 11611.

Employee Id	First Name	Last Name
28	Emily	Johnson
53	Michael	Smith
12	Olivia	William
18	Daniel	Brown
8	Sophia	Jones
2	Ethan	Davies
19	Ava	Miller
57	Benjamin	Wilson
62	Isabella	Moore
34	Jacob	Taylor
23	Mia	Anderson
14	William	Thomas
48	Emma	Jackson
35	Alexander	Moore
55	Charlotte	Haris
22	James	Martin

Employee ID	FirstName	Last Name
2	Ethan	Davies
7	Harper	Martinez
8	Sophia	Jones
9	Elijah	Robinson
12	Olivia	William
14	William	Thomas
15	Logan	Garcia
18	Daniel	Brown
19	Ava	Miller
22	James	Martin
23	Mia	Anderson
26	Amelia	Thompson
28	Emily	Johnson
32	Grace	Smith
34	Jacob	Taylor
35	Alexander	Moore

Time Taken (Elapsed Ticks): 11611

2. File data, unsorted List and List sorted on last name

The screenshot shows the ICA08-Sorting Structs application window. It features two data grids: 'Unsorted Data' on the left and 'Sorted Data' on the right. The 'Unsorted Data' grid lists employees with IDs 123, 245, 367, 489, 512, 634, 756, 878, 901, 134, 256, 378, and 491. The 'Sorted Data' grid shows the same employees sorted by their Last Name, with IDs 904, 644, 903, 256, 640, 764, 496, 520, 141, 512, 513, 493, 522, 143, 138, and 494. The 'Data Source' is set to 'File Data'. The 'Sort By' is set to 'Last Names'. The 'Time Taken (Elapsed Ticks)' is 6492.

Employee Id	First Name	Last Name
123	Emma	Smith
245	Liam	Johnson
367	Olivia	Williams
489	Noah	Jones
512	Ava	Brown
634	Elijah	Davis
756	Sophia	Miller
878	Oliver	Wilson
901	Isabella	Moore
134	Lucas	Taylor
256	Mia	Anderson
378	Mason	Thomas
491	Hamer	Jackson

Employee ID	FirstName	Last Name
904	Scarlett	Adams
644	Aaron	Alexander
903	Alexander	Allen
256	Mia	Anderson
640	Aria	Bailey
764	Claire	Barnes
496	Elena	Bell
520	Mila	Bennett
141	Jack	Brooks
512	Ava	Brown
513	Ethan	Brown
493	Henry	Brown
522	Camila	Bryant
143	Aaliyah	Butler
138	Penelope	Campbell
494	Chloe	Carter

Time Taken (Elapsed Ticks): 6492

Rubric: Max Marks: 100

Item	Max Marks	Penalty
UI	15	Unprofessional layout: -4 No proper tab order: -2 Using default control names: -3
Struct Properly defined	5	
Provided unsorted data properly displayed in left DataGridView.	10	Wrong Format: -2
Provided table Data properly sorted through Selection Sort as required	15	
Provided table Data properly sorted through Insertion Sort as required	15	
Drag and Drop working as required. Data loaded in left DataGridView	10	
File data sorted and displayed- Both sorting methods working as required	10	
File Loaded through Open File Dialog	10	
Execution Time calculated and Displayed	7	
DataGridViews cleared by corresponding buttons	3	
		Data Loaded in List boxes not according to radio buttons: -5
Documentation: Programmer Block Appropriate Variable Names Program Properly commented		Missing components of documentation: -5 to -20

If struct and lists of structs are not used, the lab will be marked as 0. Also your DataGridView columns must be set to be Unsortable.