***FAST NUCES, Karachi***



***Project Report Computer Organization and Assembly Language***

***Submitted to: Sir Nadeem Ghouri***

***Project: Sorting Program***

***Group Members:***

* ***Bilal Shakeel (21K-4874)***
* ***Muhammad Tahir (21K-4503)***
* ***Muhammad Samamah (21K-3205)***

***Application, techniques and library used:***

Our project is made on Visual Studio 2019 and the library we used was Irvine. Our code includes nested and normal loops with various functions, which are both, custom made and pre made. Most of the code relies on the conditions used for the loops.

***Abstract***

The sorting program is a multipurpose program that sorts strings, characters and integers. The user has to enter the number of inputs after which the sorting algorithm works and produces the output. The string algorithm works by separating words by comparing with spaces between the strings. Characters are sorted according to their ASCII values whereas integers are sorted according to their values. The user can continue to sort through a user-friendly menu until they want to exit.

***Problem Statement:***

The main reason to pursue this was to make an efficiently working program that sorts characters, integers and strings separately with separate functions for them. Using the same sorting function technique for different options can cause errors. Creating different functions for different options leads to accuracy in results.

***Work Distribution:***

**Bilal Shakeel (21K-4874)**

* **Project Report/Project Proposal**
* **User interface and integer sort (Bubble Sort)**

**Muhammad Tahir (21K-4503)**

* **Word/Sentences Sort**
* **String Sort**

**Muhammad Samamah (21K-3205)**

* **Character Sort (Bubble Sort)**
* **Working on visuals (crlf, clear screen etc)**

***Methodology:***

* We first present a home page that asks user to select what type of sorting they need.
* After that, we call the function made specifically for that sorting option. It asks the user for the number of inputs it wants to give.
* The user is then prompted to enter the elements.
* For character and integer sort, we are using normal bubble sort (i – 1), with I being the iterator used.
* For string and sentences sort, we use spaces to separate words from the strings through conditions.
* We put the separate words in a 2D array for sorting. For example, a 2-line sentence is entered. Row would show line and column would show word in that line.
* We then use the offset of the 2D array and the string entered.
* The compare loop stores the addresses of the two strings and compares them in registers.
* We compare words according to their ASCII values. If the starting character is the same, we check the second and onwards character.
* We keep on checking till we get to the end of the string array.
* The user interface keeps on popping up until the user exits.

***Future Work/Recommendations:***

* To make this program better, we can give the user an option to select a sorting algorithm of their own choice to sort the options.
* We can provide different scenarios and items for one to sort.

***Sample Output:***











