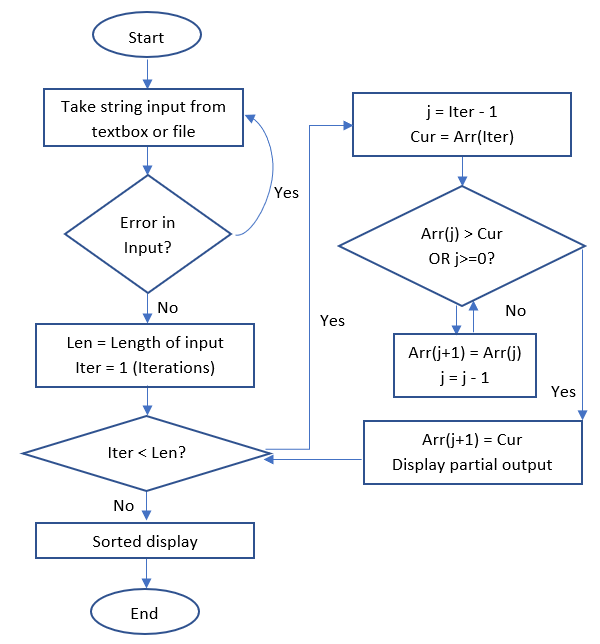
CS244 - Project 1

~ Technical Documentation

# Visual Learning

Group Number: 23

Group Members:

* Mayank Baranwal: 170101084
* Soumik Paul: 170101066
* N. K. V. Sai Kiran: 170101039

Stakeholders: School children, college youth, people new to coding

Design Goals: To implement insertion sort and visualize the intermediate steps (comparison, swapping) in a manner that both user friendly and easy to understand for audiences of all ages

Flowchart Breakdown:

1. Take a string input and check its validity (covered in corner cases below)
2. Run (Length-1) iterations of insertion sort on next button “next click”
3. In (k+1)th iteration, start at element in kth part of pre-sorted array and move leftwards till an element smaller than the (k+1)th element is found
4. Display the numbers being compared at each comparison
5. At end of each iteration, display the partial output with green highlighting sorted and red highlighting unsorted part
6. Continue till all entries are sorted

Insertion Sort Algorithm (Incremental Paradigm)

Input Format: Take data size n, followed by input of n elements to be sorted

Output Format: Step by step sorted output (visually) with option for user to skip steps

Inductive Algorithm:

* Base Case: For n=1, array of size 1 is already sorted
* Inductive Step: For n=(k+1), the first (k) elements are already assumed to be sorted. The (k+1)th element is compared with the sorted array from the right side until an element smaller than it is found. Then, it is inserted just after the first smaller element. This results in sorted array of (k+1) size.
* Termination: After the nth element is processed, the array is sorted.

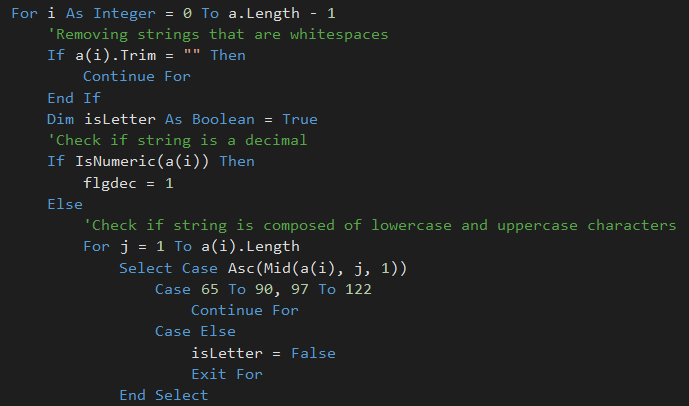
Time Complexity: Worst case – O(n2); Best case – O(n)

Space Complexity: O(1)

Corner/ Edge Cases:

1. Invalid input (Special characters not allowed) 🡪 e.g. “ap;ple”
2. Mixed type input (Strings and decimals) 🡪e.g. “apple 123”
3. Decimals special condition on negative numbers 🡪 e.g. “22-“
4. Overflow/Underflow of decimals 🡪 e.g. “1111….111”
5. No input (blank space) is given

Modules and Functions:

* Clr() 🡪 Clear function to reset flags, arrays and clear dynamically allocated textboxes
* InpFail() 🡪 Displays message box on invalid input being entered
* TxtIndex (strt index, fin index, int cnt, output string) 🡪 Finds out till where in the text box colour is green (sorted)
* outTextAdder(i index, tmp string,strt index, end index) 🡪 Dynamically allocates new text boxes to print partial output of insertion sort with green and red highlighting colour
* Delay (dblSecs) 🡪 System wait for x seconds
* SortStr() 🡪 Performs one iteration of insertion sort for strings, outputting the numbers being compared and then call TxtIndex and outTextAdder to print the partial output with sorted and unsorted text highlighted
* SortDec() 🡪 Same function as SortStr(), except that it does it for decimals
* Checker() 🡪 Takes in input from text field or input file. Processes the input string, removing extra whitespaces, checking and rejecting corner cases (invalid input, mixed input, overflow). Finally, it converts string to decimals and calls on the sorting functions to perform one iteration of insertion sort
* btnExit (sender, e as EventArgs) 🡪 Triggers the application closing
* btnSort (sender, e as EventArgs) 🡪 Triggers the Checker() function to take input and start sortin
* btnClr (sender, e as EventArgs) 🡪 Triggers the Clr() button to reset screen (except for input)
* btnNxtClick (sender, e as EventArgs) 🡪 Calls the next iteration of sorting if still unsorted or displays that sorting is finished
* Form1\_Load(sender As Object, e As EventArgs) 🡪 Loads the form1 and form2 with x and y coordinates of output box
* txtArr\_KeyDown(sender As Object, e As EventArgs) 🡪 Allows the user to start insertion sort by pressing the enter key
* btnResInp\_Click(sender As Object, e As EventArgs) 🡪 Clears the input textbox
* btnHelp\_Click(sender As Object, e As EventArgs) 🡪 Redirects user to user documentation

Buttons:

* Clear 🡪 Calls the Clr() function and brings back the Start Sorting function
* Start Insertion 🡪 Calls the Insertion sort function
* Next Step 🡪 Iterates the next round of sorting if array is still unsorted
* Exit 🡪 Close the application
* Read from .txt file ? 🡪 Opens Windows Explorer to pick a txt file as input for text file
* Help 🡪 Redirects user to user documentation

Text Fields:

* Input Field: Takes input string/ decimals from user
* Current Comparison: Shows greater than, less than comparison between element to be inserted and element being compared against
* Output Field: Dynamic set of boxes showing partial output at each step of the insertion sort

Error Prevention Techniques:

* Making the input text field read-only post sorting start, and keeping comparison and output read only throughout
* Making all fields multi-line to accommodate large inputs without text box overflow
* Clear and Next step buttons disappear during an iteration of insertion sort to prevent extra errors