Name: Malav Doshi , Herik Patel

Systems programming

Professor Francisco

Assignment 0

Description

For assignment 0 we are supposed to sort given set of data which could either be given in form of strings or numbers. For numbers we sort them by comparing in a regular way. But for string we compare ascii value of first digit of each string and sort them accordingly. If the value of first character is same we compare value of same string for second character and so on. Algorithm that we will be using to compare data is quicksort which has time complexity of nlogn and insertion sort with complexity n^2.To store data we are using node and data structure that we have used is linked list. We are storing the data that we have in file in it that we have from file.

-**store_to_Node:** Function takes char pointer and int as argument and runs a loop until the null character is found , and stores the string it into a node and after storing it into the node we check if the string is complete or incomplete if it is complete we and head is null we are storing it in the head node. If the String is not complete then we return and the function is called again and we find remaining part of string and then we store the complete string in the node. To find the incomplete string we take the string stored in and try to find if the character that separates string like comma or character similar to it. This function does not return anything.

-**check_digits:** In this function we take head node as parameter and check if the values stored in the nodes are numbers or not if the value in node is number we change them to integers as they will be stored as characters. We change them so that we can sort them . This function does not return anything.

- **print_list:** This function takes head node as argument and checks if the stored value is a digit or character if and then prints it in a sorted way. This function does not return anything

- **delete_prev:** deletes all the contents in the given string and returns an empty string

-**make_str_arr:** Makes a structure to store strings .It does not return any value

-**make_arr:** Makes an array that stores numbers in the array. It does not return any value.

-**lastNode:** runs a loop to till we reach last node of the list. This function returns node pointer.

-**int_partition:** Function takes two int argument high and low so that it can implement quick sort by using pivot and it increments and changes value of high and low value and pivots used to sort integers . Value returned by this function is an integer value.

-**int_quick_sort:** Implements quick sorting algorithm and parameter needed are two integer value it sorts numbers . BigO of quicksort algorithm is nlogn .It does not return any value .

-**string_partition:** Function takes two int argument high and low so that it can implement quick sort by using pivot and it increments and changes value of high and low value and pivots used to sort Strings. Value returned by this function is an integer value.

-**string_quick_sort:** quick sorting algorithm and parameter needed are two integer value it sorts strings. BigO of quicksort algorithm is nlogn. It does not return any value.

-**string_arrange:** Takes three pointer as argument and sorts string as we perform insertion sort on strings. It arranges value of the string according to pivot.

- **quicksort:** This function checks if the data given to us is strings or integer and calls function to quicksort after identifying data type. It returns an integer value .

-**insertion_sort:** Performs insertion sort algorithm on given data and sorts it with time complexity of n^2.Value returned by this function is an integer value.

-**string_comparator:** Takes two void pointer as argument and compares two string with respect to their ascii values and returns integer value. Used to get data which value is bigger so that we can sort given sort of data. Value returned by this function is an integer value.

-**int_comparator:** Also takes two void pointer as argument and compares two numbers and returns integer value. Used to get data which value is bigger so that we can sort given sort of data. Value returned by this function is an integer value.

-**print_array:** print the given integer array.

-**print_str_array:** print the given string array.

- **free_list:** free the memory allocated to any linked list.