Data Structures Lab

Session 4

Course: Data Structures (CS2001) **Semester:** Fall 2021

Instructor: **T.A:** N/A

Note:

- Lab manual cover following below elementary sorting algorithms **{Bubble, insertion, selection, Comb sort,Quick Sort}**
- Maintain discipline during the lab.
- Just raise your hand if you have any problem.
- Completing all tasks of each lab is compulsory.
- Get your lab checked at the end of the session.

Task-1:

Given an array of strings arr[]. Sort given strings using Bubble Sort and display the sorted array according to the increasing ASCII values of each character in the string .

String : SOFTWARE

Key Points:

1. Bubble Sort, the two successive strings arr[i] and arr[i+1] are exchanged whenever arr[i]> arr[i+1]. The larger values sink to the bottom and hence called sinking sort. At the end of each pass, smaller values gradually "bubble" their way upward to the top and hence called bubble sort.

Key Points:

boolean selectionSort(int *array, int size) {
Find the smallest element in the array and exchange it with the element in the first position.

Find the second smallest element in the array and exchange it with the element in the second position.

Continue this process until done.

}

Key Points:

boolean insertionSort (int *array, int size) {
Choose the second element in the array and place it in order with respect to the first element.

Choose the third element in the array and place it in order with respect to the first two elements.

Continue this process until done.

Insertion of an element among those previously considered consists of moving larger elements one position to the right and then inserting the element into the vacated position

}

Quick Sort:

Key Points:

- Step:1 Consider the last element of the list as **pivot** (i.e., Element at last position in the list).
- **Step 2 -** Define two variables low and high. Set low and high to first and last elements of the list respectively. and set pivot to arr[high]//last element.
- **Step 3 -** Initialize the smaller element i to low-1// i = low-1 and next element j. Set j= low.
- **Step 4 -** Traverse the elements from j to high -1(second last element)
- **Step 4.1-** If arr[j] < pivot then increment i and swap arr[i] and arr[j].
- **Step 6 -** Repeat steps 3,4 & 4.1 until j recahes high -1.
- **Step 7 -** Exchange the pivot element with arr[i + 1] element.
- **Step 8** - Return i when added 1.

arr[] = {10, 80, 30, 90, 40, 50, 70}

Indexes: 0 1 2 3 4 5 6

low = 0, high = 6, pivot = arr[h] = 70

Initialize index of smaller element, **i = -1**

Traverse elements from j = low to high-1

j = 0 : Since arr[j] <= pivot, do i++ and swap(arr[i], arr[j])

i = 0

arr[] = {10, 80, 30, 90, 40, 50, 70} // No change as i and j are same

j = 1 : Since arr[j] > pivot, do nothing

// No change in i and arr[]

j = 2 : Since arr[j] <= pivot, do i++ and swap(arr[i], arr[j])

i = 1

arr[] = {10, 30, 80, 90, 40, 50, 70} // We swap 80 and 30

j = 3 : Since arr[j] > pivot, do nothing

// No change in i and arr[]

j = 4 : Since arr[j] <= pivot, do i++ and swap(arr[i], arr[j])

i = 2

arr[] = {10, 30, 40, 90, 80, 50, 70} // 80 and 40 Swapped

j = 5 : Since arr[j] <= pivot,

do i++ and swap arr[i] with arr[j] **i = 3**

arr[] = {10, 30, 40, 50, 80, 90, 70} // 90 and 50 Swapped

We come out of the loop because j is now equal to high-1.

Finally we place pivot at correct position by swapping arr[i+1] and arr[high] (or pivot)

arr[] = {10, 30, 40, 50, 70, 90, 80} // 80 and 70 Swapped

Now 70 is at its correct place. All elements smaller than 70 are before it and all elements greater than 70 are after it.

Comb Sort:

1. Create and initialize variables gap and swapped and constant SHRINK_FACTOR
   a) gap = size of the array
   b) swapped = false
   c) SHRINK_FACTOR = 1.3
2. Set swapped = false
3. Set gap = gap/SHRINK_FACTORki
4. Iterate over the array from i = 0 to i < n - gap:
   if array[i] > array[i + gap]
   a) swap the elements array[i] and array[i + gap]
   b) set swapped = true
5. Repeat steps 2-4 while gap != 1 and swapped = true

Task02: Write a menu driven program that asks the user to enter any number, the number denotes a specific algorithm function to be called. Sort the array based on the algorithm function called by the user.

List=[5, 3, 8, 1, 4, 6, 2, 7, 10, 9].