

**OBJECTIVE :** String Operations, Usage of Sorting and Binary Search Algorithms, Usage of Binary Search and Merge Algorithms

**Instructor :** Yusuf Evren AYKAÇ

**Assistants :** Elif GÜL, Yusuf Şevki GÜNAYDIN, Hatice ÇATALOLUK

1. Write the function **longest** that takes the string array and the number of words in the array as an input parameters, finds and returns the index of the **last longest word** in the array.

Write a C program that takes a several words from the user until the word “STOP” is entered, stores the words in the string array, finds the longest word in the array and displays the longest word and its length on the screen.

**Example Run#1:**

```
Enter a word (or STOP): biscuit
Enter a word (or STOP): coffee
Enter a word (or STOP): brownie
Enter a word (or STOP): beef
Enter a word (or STOP): bread
Enter a word (or STOP): STOP
```

```
Longest word:brownie
Length: 7
```

**Project Name:** LabGuide4\_1

**File Name:** Question\_1.cpp

2. Write a function named **findLastOcc()** that takes a sentence and a string to be searched as input parameters, finds and returns the index of the **last occurrence of the given string** in the sentence.

Write a C program that will read a sentence and a key string from the user, finds the LAST OCCURENCE of the given key string and displays the sentence back until the key string's last occurrence.

**Example Run #1:**

```
Enter a sentence: do not go gentle into that gentle good night
Enter a key string: gentle
Result: do not go gentle into that
```

**Example Run #2:**

```
Enter a sentence: do not go gentle into that good night
Enter a key string: do
Result: That's an empty string, sorry..
```

**Example Run #3:**

```
Enter a sentece: rage against the dying of the light rage
Enter a key string: rage
Result: rage against the dying of the light
```

**Project Name:** LabGuide4\_2

**Source Name:** Question\_2.cpp

**BUBBLE SORT ALGORITHM:**

1. Repeat
  2. Initialize sorted 1
  3. Repeat for each pair of adjacent array elements
  4. If the values in a pair are out of order
    - 4.1. Exchange the values
    - 4.2. Set sorted to 0
- as long as the array is not sorted

**SELECTION SORT ALGORITHM:**

1. For each value of fill from 0 to n - 1
2. Find index\_of\_min, the index of the smallest element in the unsorted sub array list[fill] through list[n-1].
3. If fill is not the position of the smallest element (index\_of\_min)
4. Exchange the smallest element with the one at position fill.

3. Write a C program that will get the points for several football teams from a file “**points.txt**” and:
- a) Sort the points with selection sort algorithm in **ascending order** and display them on the screen.

**Example Run:**

Points

-----

16  
17  
20  
25  
26  
26  
27  
29  
30  
30  
31  
32  
35  
35  
40  
41  
49  
53

**points.txt**

41  
40  
35  
26  
30  
17  
16  
29  
53  
49  
27  
26  
35  
32  
31  
30  
25  
20

Project\_name: Labguide4\_3a

File\_name: Question\_3a.cpp

- b) Sort the points with bubble sort algorithm in **descending order** and display them on the screen.

**Example Run:**

Points

-----

53  
49  
41  
40  
35  
35  
32  
31  
30  
30  
29  
27  
26  
26  
25  
20  
17  
16

Project\_name: Labguide4\_3b

File\_name: Question\_3b.cpp

4. a) Write a C program that forms a Disco Song list by getting song names from the user until "end" is entered. Each song name will be added to the list and the list should be sorted in ascending order by using bubble sort algorithm. After sorting, the new form of the list will be displayed on the screen. Assume that NO duplicate value is given.

Project\_name: Labguide4\_4a

File\_name: Question\_4a.cpp

**Example Run:**

```
Enter a song name (end to stop): Stanga
Enter a song name (end to stop): Bonbon
Enter a song name (end to stop): OMG
Enter a song name (end to stop): Rockabye
Enter a song name (end to stop): Habib Galbi
Enter a song name (end to stop): Just Say
Enter a song name (end to stop): But A Lie
Enter a song name (end to stop): Age of Emotions
Enter a song name (end to stop): end
```

Disco Songs

```
-----
1) Age of Emotions
2) Bonbon
3) But A Lie
4) Habib Galbi
5) Just Say
6) OMG
7) Rockabye
8) Stanga
```

- b) modify the program **Question\_4a.cpp**, so the list will be sorted in descending order by using bubble sort algorithm.

Project\_name: Labguide4\_4b

File\_name: Question\_4b.cpp

**Example Run:**

```
Enter a song name (end to stop): Stanga
Enter a song name (end to stop): Bonbon
Enter a song name (end to stop): OMG
Enter a song name (end to stop): Rockabye
Enter a song name (end to stop): Habib Galbi
Enter a song name (end to stop): Just Say
Enter a song name (end to stop): But A Lie
Enter a song name (end to stop): Age of Emotions
Enter a song name (end to stop): end
```

Disco Songs

```
-----
1) Stanga
2) Rockabye
3) OMG
4) Just Say
5) Habib Galbi
6) But A Lie
7) Bonbon
8) Age of Emotions
```

**BINARY SEARCH ALGORITHM:**

1. Let top be the subscript of the initial array element.
2. Let bottom be the subscript of the last array element.
3. Repeat until top exceeds bottom, thus there are no more elements to check
4. Let middle be the subscript of the element halfway from top to bottom.
5. If the element at middle is the target, than return middle.
6. else if the element at middle is larger than the target, let bottom be middle-1, thus continue the search in the first half.
7. else let top be middle+1, thus continue the search in the second half.
8. Return -1 since the loop terminated, but the number is not found.

5. a) Write a C Program which search a record of a car plate from the sorted data in **plates.txt** produced in. Use sequential search, and prints how many comparisons made to find the record.

**Example Run:**

```
Enter plate of a car (END for exit): 06HTC452
EFE KOROGU 06HTC452
14 comparisons.
```

```
Enter plate of a car (END for exit): 01KL5641
BARIS COLAK 01KL5641
1 comparisons.
```

```
Enter plate of a car (END for exit): 53EF4587
DEMIRCAN COSKUN 53EF4587
37 comparisons.
```

```
Enter plate of a car (END for exit): 06GHK567
NOT FOUND
37 comparisons.
```

```
Enter plate of a car (END for exit): END
```

- b) Modify the program **Question\_1a.cpp** using Binary Search algorithm.

**Example Run:**

```
Enter plate of a car (END for exit): 06HTC452
EFE KOROGU 06HTC452
3 comparisons.
```

```
Enter plate of a car (END for exit): 01KL5641
BARIS COLAK 01KL5641
5 comparisons.
```

```
Enter plate of a car (END for exit): 53EF4587
DEMIRCAN COSKUN 53EF4587
6 comparisons.
```

```
Enter plate of a car (END for exit): 06GHK567
NOT FOUND
6 comparisons.
```

```
Enter plate of a car (END for exit): END
```

**Project\_name:** LabGuide4\_5a

**File\_name:** Question\_5a.cpp

**plates.txt**

BARIS	COLAK	01KL5641
DENIZ	CORBACI	01RR5678
DENIZ	AGAH	06ABC453
OZAN	CERGEL	06BA1246
BARIS	ARSLAN	06BB9987
ERGIN	ERANT	06BLN027
ABDULLAH	ARSLAN	06CAN063
EMRE	DINCEL	06CRN027
MUSTAFA	KAPLAN	06EMN652
FATMA	KAYA	06ERD063
BERAT	KINA	06ERK751
OZGUR	AKBABA	06ES4587
HUSEYIN	KINIKLI	06GNS697
EFE	KOROGU	06HTC452
CELAL	ERBAY	06JAL254
ABDULLAH	AKTAS	06KM3657
ONUR	BAGDADIOGLU	06KRM323
AYLIN	GURTUNA	06KRM442
BATURAY	HASER	06MNV660
MERTHAN	ILVAN	06OIL633
BARIS	INALOZ	06PRL471
ESIN	INANC	06REN694
DENIZ	INTEPE	06RMZ458
CAN	KANBAY	06VYS452
MUSTAFA	BOLAT	07ABC487
BILGEN	BILGIN	07TRK658
EDIZ	CITAK	19HJ1245
ALI	CICEK	19HT6547
DERYA	DEDEOGLU	27CNR006
AYSE	BAYHAN	34KLM365
ZEYNEP	AKANDIR	34ZZ1785
MERT	AKDEMIR	35ED5678
TAMER	CAN	35TT2369
CUNEYT	EKINCI	35ZRFO06
GAMZE	BASIBUYUK	42HKN451
ABDULLAH	BATTAL	42MN4178
DEMIRCAN	COSKUN	53EF4587

**Project\_name:** LabGuide4\_5b

**File\_name:** Question\_5b.cpp

6. Write a C program that gets the names of bicycles from two different sorted files named **bike1.txt** and **bike2.txt** into two arrays, and merges them and writes in a new file named **bike\_list.txt**.

Write the following functions;

- **readBikeList** that takes the file pointer and the array which will keep the bike list as parameters, reads the bike names from the file into the specified array. The function should also return the number of bikes in the array.
- **shiftDown** that takes the bike list array, number of bikes and the position of the bike which will be shifted down as parameters. The function moves down all of the elements starting from specified position in the array.
- **merge** that takes two bike lists and their sizes as parameters, merges two array putting the bike names in the second array into the first array in a sorted form.

In main call the necessary functions.

**Project\_name:** LabGuide4\_6

**File\_name:** Question\_6.cpp

Example Run:

**bike1.txt**

bianchi  
bisani  
colnago  
gazelle  
look  
raleigh  
trek

**bike2.txt**

beldesani  
beldeyama  
bisani  
bmx  
giant  
kona  
marin  
pinokyo  
polo  
salcano  
scott

**bike\_list.txt**

beldesani  
beldeyama  
bianchi  
bisani  
bmx  
colnago  
gazelle  
giant  
kona  
look  
marin  
pinokyo  
polo  
raleigh  
salcano  
scott  
trek

**Note that:** If we want to eliminate the **duplicate** bike names, then remember the lecture notes..