

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. a) Write the function **findFirst** which takes a sentence and a string to be searched as input parameters, finds and returns the index of the **first occurrence of the given string** in the sentence. If the sentence does NOT contain the searched string the function should return -1.

Write a C program that will input a sentence, find and display the position of the given **string** in the sentence. If the searched string is NOT found, display an appropriate message.

Project Name: LabGuide4_1a

File Name: Question_1a.cpp

Example Run#1:

Enter a sentence: this is a good idea

Enter a string: is

The first occurrence of the string <is> is 2

Example Run#2:

Enter a sentence: Why your smartphone will be your next pc

Enter a string: are

The sentence does NOT contain the string <are>

- b) Modify the **Question_1a.cpp**, so the program will **delete** the **first occurrence** of the searched string.

Write the function **deleteFirst** that takes a sentence, a string and the starting index of the given string in the sentence as parameters. The function will delete the given string in the sentence.

Project Name: LabGuide4_1b

File Name: Question_1b.cpp

Example Run#1:

Enter a sentence: home sweet home alabama home

Enter a string: home

The new form of the sentence after deletion: sweet home alabama home

- c) Modify the **Question_1b.cpp**, so the program will **delete** the **first occurrence** of the searched **WORD**.

Write the function **deleteFirst** that takes a sentence, a word and the starting index of the given word in the sentence as parameters. The function will delete the given WORD in the sentence.

Project Name: LabGuide4_1c

File Name: Question_1c.cpp

Example Run#1:

Enter a sentence: brush your teeth before you go to bed.

Enter a word: you

The new form of the sentence after deletion: brush your teeth before go to bed.

2. Write a simple parser, which separates data from the string and store the fields in a structure. The fields are separated by a semicolon. User enters the salary increase, and displays the employee structure.

Example Run:

```
Enter an employee: Evren:Aykac:3462:31:6700:Sales Department
Enter increase in the salary: 250
```

```
Employee Information
-----
Name : Evren
Surname : Aykac
Employee ID : 3462
Age : 31
Salary : 6950
Department : Sales Department
```

Project Name: LabGuide4_2
File Name: Question_2.cpp

Hint: Use atoi() function for converting a string to an integer. Please examine the following code and its output.

```
int main()
{
    int number;
    char str[4];
    printf("Enter a number: ");
    scanf("%s",str);
    number = atoi(str);

    printf("Value of the number is %s\n", str);
    printf("Value of the number * 2 is %d\n", number * 2);

    return 0;
}
```

Example Run:

```
Enter a number: 9
Value of the number is 9
Value of the number * 2 is 18
```

3. **PhoneCorp** and **PhoneTech** are the two biggest phone companies in the US, **PhoneCorp** bought the company **PhoneTech**. They are now faced with the daunting task of merging their client data files into a single file. In particular each company has an unsorted text file with the **social security numbers, name and surname** of their clients.

Your task is to create a program that takes client information from the files into two structure arrays, sorts and merges the lists and writes the new list to the file **clients.txt**.

Write necessary functions;

read client info from a file into a structure array,

sort client list according to the social security numbers in ascending order.

merge client lists of both companies into the list which keeps the information of the company **phoneCorp**.

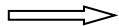
Project_name: LabGuide4_3

File_name: Question_3.cpp

<PhoneCorp.txt>

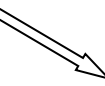
342654658 James Abbot
154564564 David Addison
993219953 Robert Court
422354943 Patricia Stark
493215812 Linda Baker
276546846 Steven Parker
896546543 Karen Davies
399321545 Ronald Fox
113624982 Sarah Glover
283216569 Deborah Walker
393546562 Nancy Forbes
316243213 George Taylor

Sort



Sorted PhoneCorp Array

113624982 Sarah Glover
154564564 David Addison
276546846 Steven Parker
283216569 Deborah Walker
316243213 George Taylor
342654658 James Abbot
393546562 Nancy Forbes
399321545 Ronald Fox
422354943 Patricia Stark
493215812 Linda Baker
896546543 Karen Davies
993219953 Robert Court



Merge

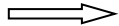
<clients.txt>

113624982 Sarah Glover
135648623 William Windsor
154564564 David Addison
168943568 Margaret Holmes
236584654 Richard Walker
276546846 Steven Parker
279461322 Susan Miller
283216569 Deborah Walker
295464654 Carol Murphy
316243213 George Taylor
333218954 Nancy Adams
342654658 James Abbot
345654858 Steven Edwards
393546562 Nancy Forbes
395221354 Edward Stone
399321545 Ronald Fox
412135468 Thomas Evans
422354943 Patricia Stark
462213589 Jennifer Rogers
493215812 Linda Baker
539565482 Kevin Hunter
783213542 Betty Johnson
896546543 Karen Davies
973213215 John Smith
993219953 Robert Court

<PhoneTech.txt>

395221354 Edward Stone
973213215 John Smith
539565482 Kevin Hunter
412135468 Thomas Evans
135648623 William Windsor
168943568 Margaret Holmes
333218954 Nancy Adams
462213589 Jennifer Rogers
295464654 Carol Murphy
279461322 Susan Miller
345654858 Steven Edwards
236584654 Richard Walker
783213542 Betty Johnson

Sort



Sorted PhoneTech Array

135648623 William Windsor
168943568 Margaret Holmes
236584654 Richard Walker
279461322 Susan Miller
295464654 Carol Murphy
333218954 Nancy Adams
345654858 Steven Edwards
395221354 Edward Stone
412135468 Thomas Evans
462213589 Jennifer Rogers
539565482 Kevin Hunter
783213542 Betty Johnson
973213215 John Smith

