**K L UNIVERSITY**

**FRESHMAN ENGINEERING DEPARTMENT**

**A Project Based Lab Report**

**On**

**ANAGRAM SEARCH**

**SUBMITTED BY**

2100030576 VENKATA VISHNU VARDHAN REDDY.VANGA

2100030573 VAMSI OMKAR.YARLAGADDA

2100030581 VAVILAPALLI LOKESH

2100030558 TUNUGUNTLA .AKHIL BHAGYESH

**UNDER THE ESTEEMED GUIDANCE OF**

**Dr. E.SREEDEVI**

**Associate Professor**

**KL UNIVERSITY**

Green fields, Vaddeswaram – 522 502

Guntur Dt., AP, India.



**DEPARTMENT OF BASIC ENGINEERING SCIENCES**



**CERTIFICATE**

This is to certify that the project based laboratory report entitled “ANAGRAM SEARCH” submitted by Mr./Ms**. Vamsi Omkar Yarlagadda** bearing Regd. No. 2100030573 to the **Department of Basic Engineering Sciences, KL University** in partial fulfillment of the requirements for the completion of a project based Laboratory in “Computational Thinking for Structure Designing” course in I B Tech I Semester, is a bonafide record of the work carried out by him/her under my supervision during the academic year 2021 – 2 022.

PROJECT SUPERVISOR HEAD OF THE DEPARTMENT

Dr.E.Sridevi Dr. D.HARITHA

**ACKNOWLEDGEMENTS**

It is great pleasure for me to express my gratitude to our honorable President **Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project-based laboratory report.

I express sincere gratitude to our Coordinator and HOD-BES **Dr. D.Haritha** for her leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor Dr.E.Sreedevi for his/her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

Name: Vamsi Omkar Yarlagadda

Regd. No: 2100030573

**ABSTRACT**

Anagram strings are defined as a string t is called an anagram of the string s, if it is possible to rearrange letters in t so that it is identical to the string s. For example, the string "aab" is an anagram of the string "aba" and the string "ana" is not.We have to find the number of good substrings of the string s (identical substrings must be).

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE NO** |
| 1 | Introduction | 6 |
| 2 | Aim of the Project | 11 |
| 2.1 | Advantages & Disadvantages |  |
| 2.2 | Future Implementation |  |
| 3 | Software & Hardware Details |  |
| 4 | Data Flow Diagram |  |
| 5 | Implementation |  |
| 6 | Algorithm for each module |  |
| 7 | Integration and System Testing |  |
| 8 | Conclusion |  |

**INTRODUCTION**

.

**A glance on C:**

* C was originally developed at Bell Labs by Dennis Ritchie between 1972 and 1973 to construct utilities running on Unix. [C](https://www.geeksforgeeks.org/c-programming-language/) is a procedural programming language. It was initially developed by Dennis Ritchie in the year 1972. It was mainly developed as a system programming language to write an operating system.
* The main features of the C language include low-level memory access, a simple set of keywords, and a clean style, these features make C language suitable for system programming’s like an operating system or compiler development. Many later languages have borrowed syntax/features directly or indirectly from the C language
* C is an [imperative](https://en.wikipedia.org/wiki/Imperative_programming) [procedural](https://en.wikipedia.org/wiki/Procedural_programming) language. It was designed to be [compiled](https://en.wikipedia.org/wiki/Compiler) to provide [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) access to [memory](https://en.wikipedia.org/wiki/Computer_memory) and language constructs that map efficiently to [machine instructions](https://en.wikipedia.org/wiki/Machine_code), all with minimal [runtime support](https://en.wikipedia.org/wiki/Runtime_system). Despite its low-level capabilities, the language was designed to encourage cross-platform programming. A [standards](https://en.wikipedia.org/wiki/Specification_(technical_standard))-compliant C program written with [portability](https://en.wikipedia.org/wiki/Software_portability) in mind can be compiled for a wide variety of computer platforms and operating systems with few changes to its source code.

**Main type:**

* The C language provides the four basic arithmetic type specifiers char, int, float and double, and the modifiers signed, unsigned, short, and long. The following table lists the permissible combinations in specifying a large set of storage size-specific declarations.
* A header file is a file with extension . h which contains C function declarations and macro definitions to be shared between several source files. There are two types of header files: the files that the programmer writes and the files that comes with your compiler.

**The C language also exhibits the following characteristics**:

* The language has a small, fixed number of keywords, including a full set of [control flow](https://en.wikipedia.org/wiki/Control_flow) primitives: [if/else](https://en.wikipedia.org/wiki/Conditional_(computer_programming)), [for](https://en.wikipedia.org/wiki/For_loop), [do/while](https://en.wikipedia.org/wiki/Do_while_loop), [while](https://en.wikipedia.org/wiki/While_loop), and [switch](https://en.wikipedia.org/wiki/Switch_statement). User-defined names are not distinguished from keywords by any kind of [sigil](https://en.wikipedia.org/wiki/Sigil_(computer_programming)" \o "Sigil (computer programming)).
* It has a large number of arithmetic, bitwise, and logic operators: +,+=,++,&,||, etc.
* More than one [assignment](https://en.wikipedia.org/wiki/Assignment_(computer_science)) may be performed in a single statement.
* Data typing is [static](https://en.wikipedia.org/wiki/Static_typing), but [weakly enforced](https://en.wikipedia.org/wiki/Strong_and_weak_typing); all data has a type, but [implicit conversions](https://en.wikipedia.org/wiki/Implicit_conversion) are possible.
* [Declaration](https://en.wikipedia.org/wiki/Declaration_(computer_programming)) [syntax](https://en.wikipedia.org/wiki/C_syntax) mimics usage context. C has no "define" keyword; instead, a statement beginning with the name of a type is taken as a declaration. There is no "function" keyword; instead, a function is indicated by the presence of a parenthesized argument list.
* Low-level access to [computer memory](https://en.wikipedia.org/wiki/Computer_memory) is possible by converting machine addresses to typed [pointers](https://en.wikipedia.org/wiki/Pointer_(computer_programming)).
* [Procedures](https://en.wikipedia.org/wiki/Procedure_(computer_science)) (subroutines not returning values) are a special case of function, with an untyped return type void.

**Anagram:**

* An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once. For example, the word anagram itself can be rearranged into naga ram, also the word binary into brainy and the word adobe into abode.
* While anagramming is certainly a recreation first, there are ways in which anagrams are put to use, and these can be more serious, or at least not quite frivolous and formless. For example, psychologists use anagram-oriented tests, often called "anagram solution tasks".
* To check given strings are anagram or not for that purpose we have to match each character of the string.
* And also we have to count the occurrence of each characters from both the strings.
* To implement it first we will arrange both strings in the same order.
* And then compare each character of both string position wise.
* If all positions have the same character in both strings then it means strings are anagram otherwise it is not an anagram.

**String:**

* In C, strings are one special kind of array: a string is an array of char values: char name[7]; I introduced the char type when I introduced types, but in short it is commonly used to store letters of the ASCII chart. Strings are such an important and useful datatype that they are implemented in nearly every [programming language](https://en.wikipedia.org/wiki/Programming_language). In some languages they are available as [primitive types](https://en.wikipedia.org/wiki/Primitive_type) and in others as [composite.](https://en.wikipedia.org/wiki/Composite_type)

Syntax - char str\_name[size];

**Strlen( ):**

* The strlen() function calculates the length of a given string.
* The strlen() function takes a string as an argument and returns its length. The returned value is of type size\_t (the unsigned integer type).
* It is defined in the [<string.h>](https://www.programiz.com/c-programming/library-function/string.h) header file.

Syntax - int strlen(const char \*str);

**Strcmp( ):**

strcmp() compares the two strings lexicographically means it starts comparison character by character starting from the first character until the characters in both strings are equal or a NULL character is encountered.

Syntax*-* strcmp(char s1,char s2 )

**strlen() :**

The strlen() function in C is used to calculate the length of a string.strlen() calculates the length of a string up to, but not including, the terminating null character.

Return Value: The function returns the length of the string passed to it.

Syntax - int strlen(const char \*str);

# 

# **String pointers:**

* Character datatypes are used to hold only 1 byte of character. It holds only one character in a variable. But we need to have more features from this character datatype as we have words / sentences to  be used in the programs. In such cases we create array of characters to hold the word / string values and add null character ‘\0’ to indicate the end of the string

**Array:**

* An array is a collection of data items, all of the same type, accessed using a common name.
* A one-dimensional array is like a list; A two dimensional array is like a table; The C language places no limits on the number of dimensions in an array, though specific implementations may.
* Some texts refer to one-dimensional arrays as vectors, two-dimensional arrays as matrices, and use the general term arrays when the number of dimensions is unspecified or unimportant.

*Example:* int a[10]

**Double dimensional array:**

* The two-dimensional array can be defined as an **array of arrays**. The 2D array is organized as matrices which can be represented as the collection of rows and columns. However, 2D arrays are created to implement a relational database lookalike data structure.

*Example:* data\_type array\_name[rows][columns];

**If-else:**

The if-else statement in C is used to perform the operations based on some specific condition. The operations specified in if block are executed if and only if the given condition is true. . It is also called two-way selection statement

**5***Example:*

1. if(condition1)
2. {
3. //code to be executed if condition1 is true
4. }else if(condition2)
5. {
6. //code to be executed if condition2 is true
7. }
8. else if(condition3)
9. {
10. //code to be executed if condition3 is true
11. }
12. Else
13. {
14. //code to be executed if all the conditions are false
15. }

**For loop:**

* The init step is executed first, and only once. This step allows you to declare and initialize any loop control variables. ...
* After the body of the 'for' loop executes, the flow of control jumps back up to the increment statement. ...
* The condition is now evaluated again.

*Example:*

For(intializationstatement ; testexperssion ; updatestatement)

{

\\ Statements inside the body of loop

}

**AIM**

A string t is called an anagram of the string s, if it is possible to rearrange letters in t so that it is identical to the string s. For example, the string "aab" is an anagram of the string "aba" and the string "ana" is not.

The string t is called a substring of the strings if it can be read starting from some position in the strings. For example, the string "aba" has six substrings: "a", "b", "a", "ab". "ba", "aba". You are given a string s, consisting of lowercase Latin letters and characters "?", You are also given a string p, consisting of lowercase Latin letters only. For example, if the string p = <aba», then the string "a??" is good, and the string «<?bc> is not.Your task is to find the number of good substrings of the string s (identical substrings must be

Input

The first line is non-empty string s, consisting of no more than 105 lowercase Latin letters and characters "?". The second line is non-empty string p. consisting of no more than 105 lowercase Latin letters. Please note that the length of the string p can exceed the length of the string s.

Output

Print the single number representing the number of good substrings of strings.Two substrings are considered different in their positions of occurrence are different.

ADVANTAGESANDDISADVANTAGES **:**

**“THERE WILL BE MANY WAYS TO ANSWER A QUESTION”**

**YOU DO YOUR WAY**

FUTURE EVOLUTION:

**SYSTEM REQUIREMENTS**

* **SOFTWARE REQUIREMENTS:**

The major software requirements of the project are as follows:

Language : Turbo-C

Operating system**:** Windows Xp or later.

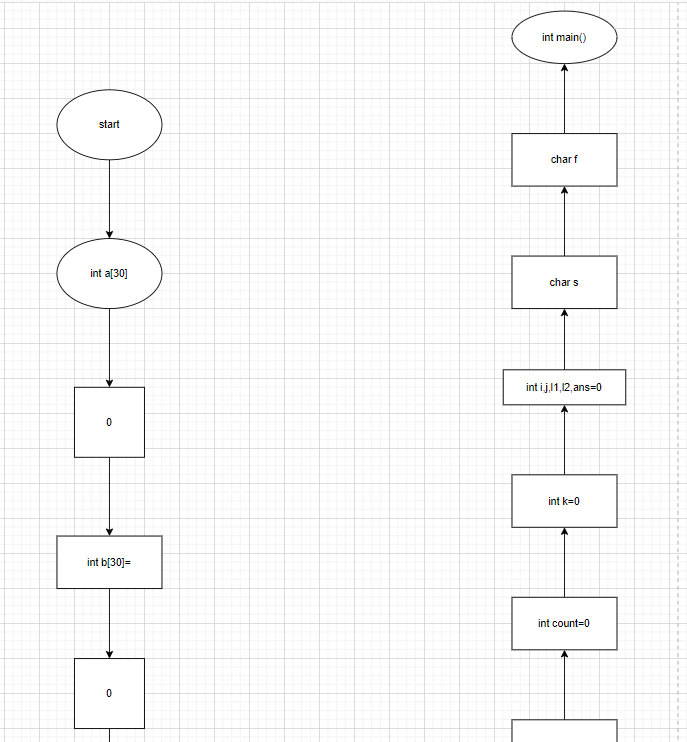
* **HARDWARE REQUIREMENTS:**

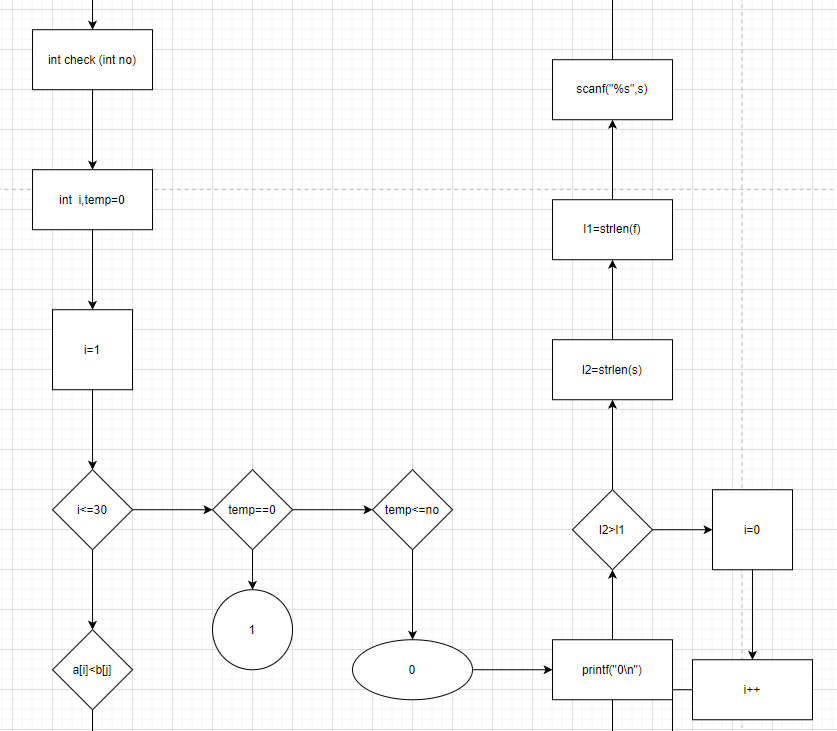
The hardware requirements that map towards the software are as follows:

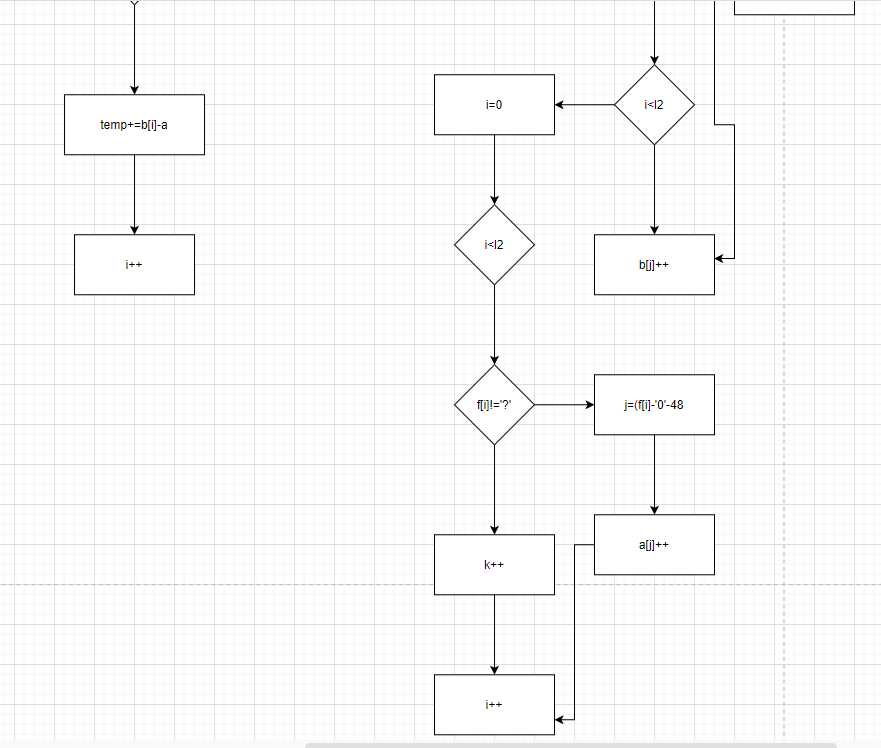
RAM :  Adequate memory depends highly on your applications and data. ...

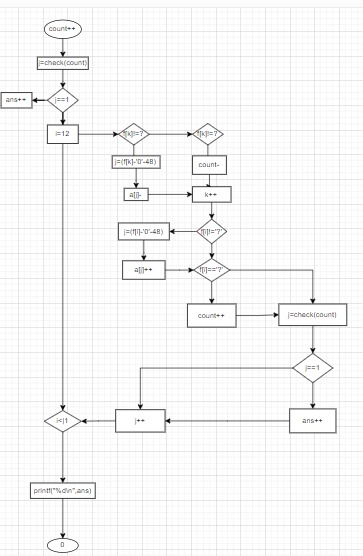
Processor : our software will run on any processor. Computation time scales with the processor speed, multi-core processors will make the software run faster. Quad or higher core processors are highly recommended.

**DATA FLOW DIAGRAM**









**ALGORITHM**

Step 1: Start

Step 2: Declare char variables int variables,I,j,i1,i2,ans=0,count=0

Step 3: Read two strings

Step 4: calculate string length of two strings

Step 5: if

Step 5.1: l2 greater than l1

Step 5.2 : display 0

Step 6: for

Step 6.1: I equal to zero (i=0)

Step 6.2: I less than l2 (i<l2)

Step 6.3: I increment increment

Step 7: j=(s[i]-‘0’-48)

Step 7.1: b(j)++

Step 8: for

Step 8.1: i equals to zero

Step 8.2: I less than l2

Step 8.3: I increment increment

Step 9: if

Step 9.1: f(i) not equals to?

Step 10: j=f(i)-‘0’-48

Step 10.1: a[j]++

Step 11: for

Step 11.1: i equals to one

Step 11.2: i less than thirty

Step 11.3: i increment increment

Step 12: if

Step 12.1: a[i] less than b[j]

Step 12.2: temp+=b[i]-a[i]

Step 13: if

Step 13.1: temp double equals to zero

Step 14: return1

Step 15: if

Step 15.1: temp less than or equals to no

Step15.2: return 1

Step 16: if

Step 16.1: j double equals to one

Step 16.2: ans++

Step 17: for

Step 17.1: i equals to l2

Step 17.2: i less than l1

Step 17.3: i increment increment

Step 18: if

Step 18.1: f(k) not equals to ?

Step 18.2: j=f(k)-‘0’-48

Step 18.3: a[j]- -

Step 19: else if

Step 19.1: f(k) double equals to ?

Step 19.2: Count - -

Step 20: if

Step 20.1: f(k) not equals to ?

Step 20.2: j=f(i)-‘0’-48

Step 20.3: a[j]++

Step 21: if

Step 21.1: j double equals to one

Step 21.2: ans++

Step 23: stop

**IMPLEMENTATION**

#include <stdio.h>

#include <stdlib.h>

#include<string.h>

int a[30]={0};

int b[30]={0};

int check(int no)

{

int i,temp=0;

for(i=1;i<=30;i++)

{

if(a[i]<b[i])

temp+=b[i]-a[i];

}

if(temp==0)

{

return 1;

}

else if(temp<=no)

{

return 1;

}

return 0;

}

int main()

{

char f[100001];

char s[100001];

int i,j,l1,l2,ans=0;

int k=0;

int count=0;

scanf("%s",f);

scanf("%s",s);

l1=strlen(f);

l2=strlen(s);

if(l2>l1)

{

printf("0\n");

return 0;

}

else

for(i=0;i<l2;i++)

{

j=(s[i]-'0'-48);

b[j]++;

}

for(i=0;i<l2;i++)

{

if(f[i]!='?')

{

j=(f[i]-'0'-48);

a[j]++;

}

else

count++;

}

j=check(count);

if(j==1)

ans++;

for(i=l2;i<l1;i++)

{

if(f[k]!='?')

{

j=(f[k]-'0'-48);a[j]--;

}

else if(f[k]=='?')

{

count--;

}

k++;

if(f[i]!='?')

{

j=(f[i]-'0'-48);

a[j]++;

}

if(f[i]=='?')

{

count++;

}

j=check(count);

if(j==1)

ans++;

}

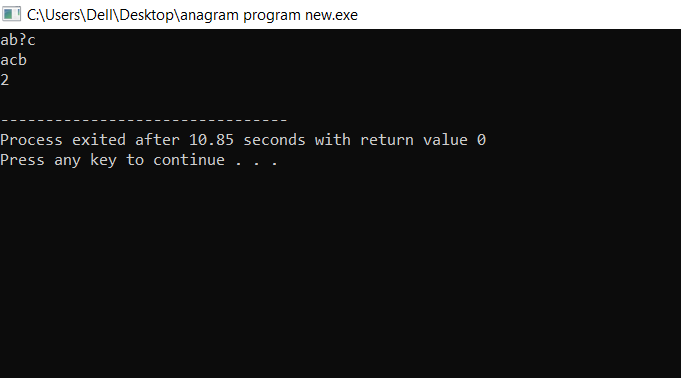
printf("%d\n",ans);

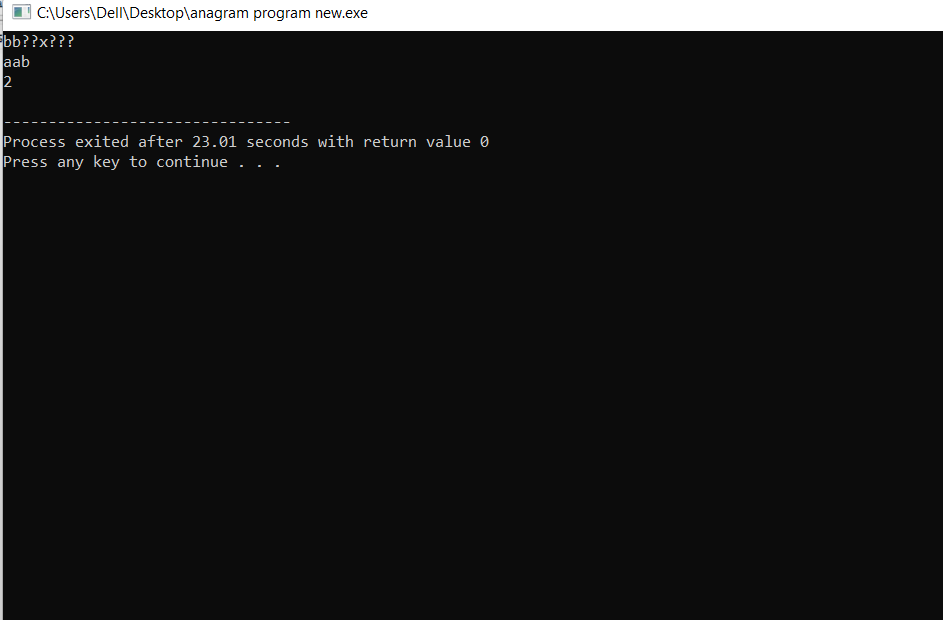
return 0;

}

**INTEGRATION AND SYSTEM TESTING**

**OUTPUTS**





**CONCLUSION**

A word is said to be an anagram of another when it can be formed by rearranging the letters of the other word using each letter just once.

E.g listen is an anagram of silent.