

EXPERIMENT No-01

Experiment Name: Write a C program that will count the number of vowels in a string.

Objective:

- The objective of this program is to take a string input from the user and count how many vowels (a, e, i, o, u) are present in it. This helps in understanding string manipulation, character checking, and loop operations in C programming.

Problem Analysis:

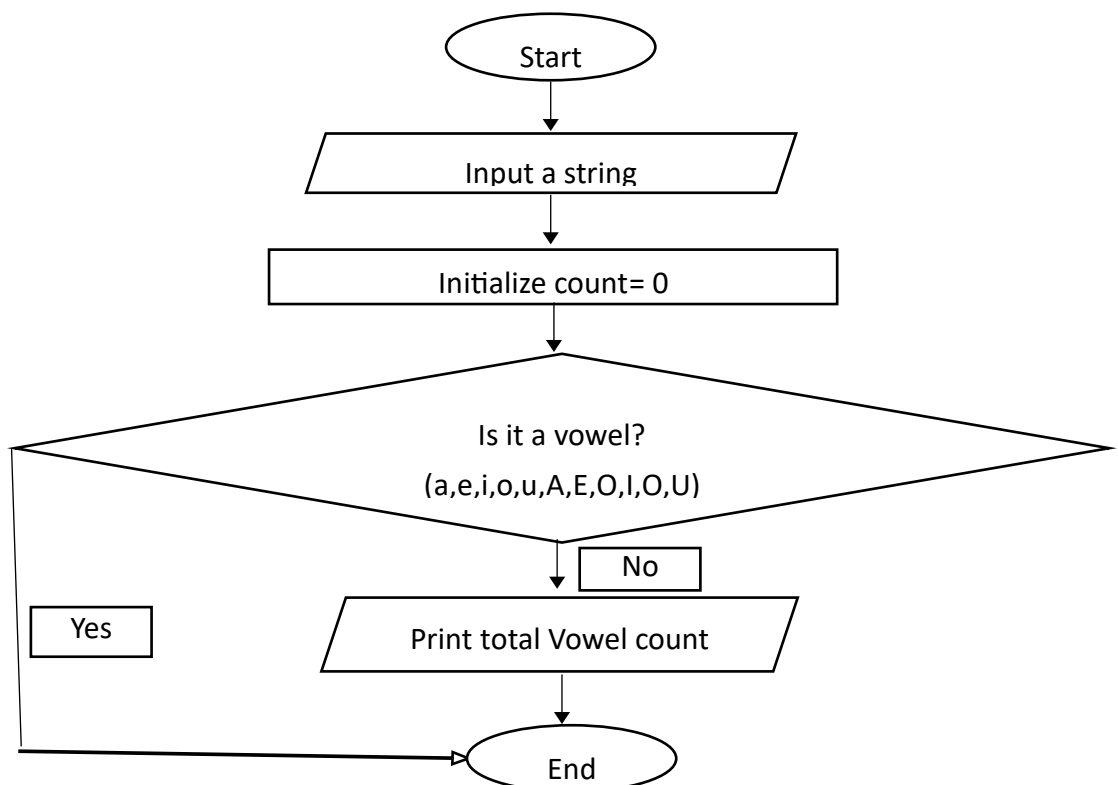
The problem is about counting the number of vowel in a string. This program is done by using a string. Again in this string the function of array has been notified. Therefore, the array is written as: stg[i]. here 'stg' stands for string. During the processing or calculation phase we don't need any extra parameter (variables) for this problem.

Input variable	Processing variable	Output variable	Header file
Stg[100]	Stg[i]=='vowel'	Count++	Include<Stdio.h>

Algorithm:

- Start
- Declare a string and initialize a counter to 0
- Input a string from the user
- For each character in the string:
 - If the character is a vowel (a, e, i, o, u or A, E, I, O, U), increment the counter
- Display the total number of vowels
- End

Flowchart:



Source Code:

```
lab6.c X
1  #include <stdio.h>
2  #include <string.h>
3
4  int main() {
5      char str[100];
6      int i, count = 0;
7
8      printf("Enter a string: ");
9      gets(str);
10
11     for(i = 0; str[i] != '\0'; i++) {
12         if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u' ||
13            str[i]=='A' || str[i]=='E' || str[i]=='I' || str[i]=='O' || str[i]=='U') {
14             count++;
15         }
16     }
17
18     printf("Number of vowels: %d\n", count);
19     return 0;
20 }
```

Output:

 D:\lab6\bin\Debug\lab6.exe

```
Enter a string: Md Atikur Rahaman
Number of vowels: 6
```

```
Process returned 0 (0x0)   execution time : 41.040 s
Press any key to continue.
```

Discussion:

The program correctly counts vowels by checking each character in the input string. It uses simple loops and conditions, making it easy for beginners to understand. Although `gets()` is used here for simplicity, using `fgets()` is safer. Overall, the program meets its goal and helps practice basic string handling in C.

EXPERIMENT No-02

Experiment Name: Write a C program that will read two strings. If given two strings are same then print "Given strings are same". If given two strings are not same then print "Given strings are not same".

Objectives:

- To learn how to read two strings using C.
- To understand how to compare two strings using strcmp() function.
- To apply conditional statements (if-else) in string comparison.

Problem analysis:

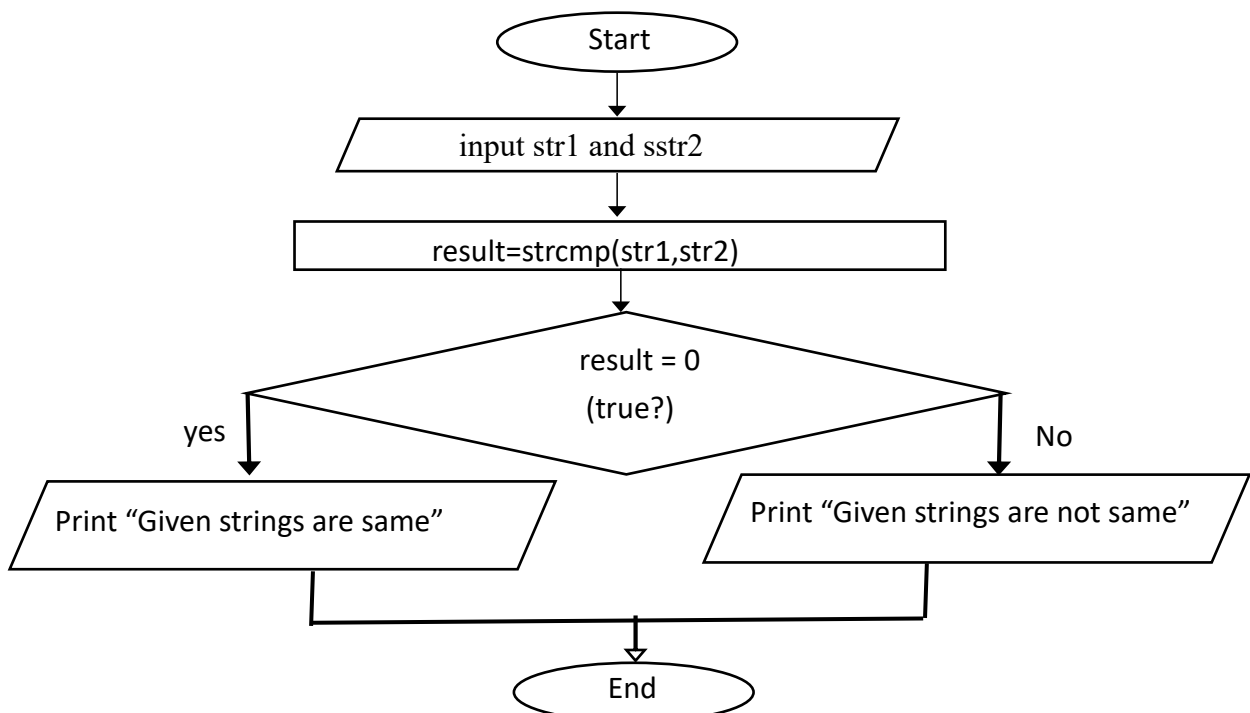
The problem describes how to detect two strings having same strings or different strings. This program is done by using if-else statement. The output of the program is displayed in any one of two ways where the result will show either 'string' are same or 'string' are not same. During the processing or calculation phase, we don't need any extra parameters for this problem.

Input variable	Processing variable	Output variable	Header file
Stg 1[50]	result=strcmp(stg1,stg2)	"Given string is same"	#include<stdio.h>
Stg2[50]		"Given string is not same"	

Algorithm:

1. Start
2. Declare two strings str1 and str2 and at
3. Input string str1 and str2
4. result = strcmp (str1, str2)
5. If the result is 0, print "Given strings are same"
6. Else, print "Given strings are not same"
7. End

Flowchart:



Source code:

```
1  #include <stdio.h>
2  #include <string.h>
3
4  int main() {
5      char str1[100], str2[100];
6
7      printf("Enter the first string: ");
8      fgets(str1, sizeof(str1), stdin);
9
10     printf("Enter the second string: ");
11     fgets(str2, sizeof(str2), stdin);
12
13
14     str1[strcspn(str1, "\n")] = '\0';
15     str2[strcspn(str2, "\n")] = '\0';
16
17     if (strcmp(str1, str2) == 0) {
18         printf("Given strings are same\n");
19     } else {
20         printf("Given strings are not same\n");
21     }
22
23     return 0;
24 }
25
```

Output:

```
D:\cce251046\bin\Debug\cce251046.exe
Enter the first string: MD
Enter the second string: MD
Given strings are same

Process returned 0 (0x0)   execution time : 10.346 s
Press any key to continue.

D:\cce251046\bin\Debug\cce251046.exe
Enter the first string: MD
Enter the second string: ATIK
Given strings are not same

Process returned 0 (0x0)   execution time : 7.624 s
Press any key to continue.
```

Discussion:

This program uses the `strcmp()` function to accurately compare two strings in C. It demonstrates basic string input and decision-making using `if-else`. By checking the return value of `strcmp()`, the program determines if the strings are identical or not. This simple logic is essential for applications like user input validation, searching, and string filtering. Overall, the program effectively shows how to handle string comparison in C.

EXPERIMENT No-03

Experiment Name: Write a C program that will concatenate one string to the end of another string and display it as output.

Objective:

- To write a C program that takes two strings as input, concatenates the second string to the end of the first, and displays the resulting combined string.

Problem Analysis:

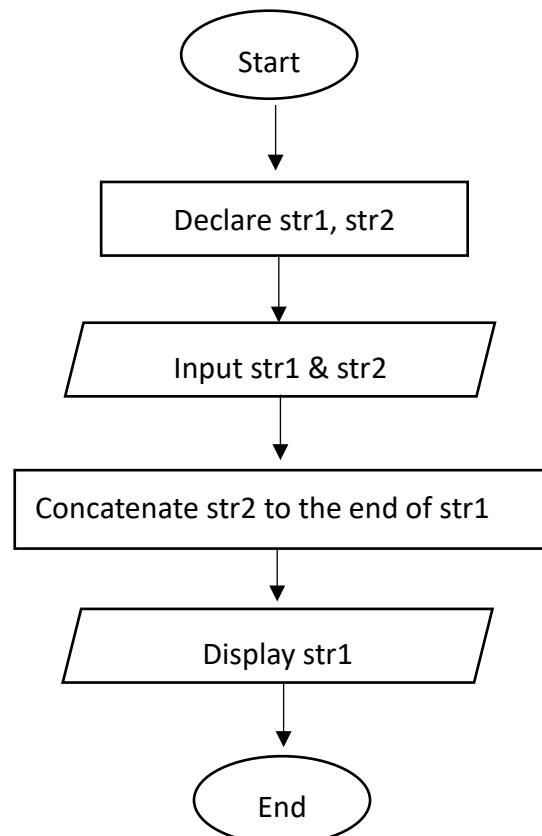
The problem describes about how to concatenate one string to the end of another string using 'strcat()' function by having its imports parameters identified as:stg1[50] and stg2 [50] (char type). The output of the program is to concatenate the two strings into one string. During the processing or calculation phase. We don't need any extra parameters for this problem.

Input variable	Processing variable	Output variable	Header file
Stg 1[50]	Strcat(stg1,stg2)	"Concatenate of two strings"	#include<stdio.h>
Stg2[50]			

Algorithm:

1. Start the program.
2. Declare two character arrays str1[100] and str2[100].
3. Ask the user to input two strings.
4. Use strcat() to append str2 to the end of str1.
5. Print the concatenated result stored in str1.
6. End the program.

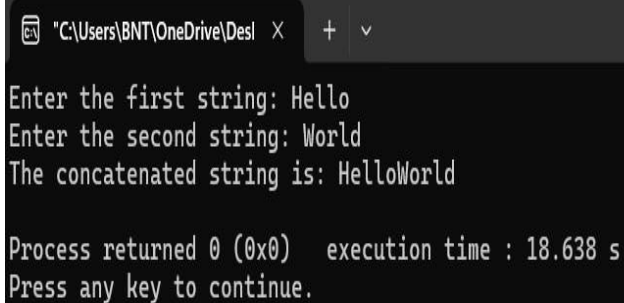
Flowchart:



Source Code:

```
1  #include <stdio.h>
2  #include <string.h>
3
4  int main() {
5      char str1[100], str2[100];
6
7      printf("Enter first string: ");
8      gets(str1);
9
10     printf("Enter second string: ");
11     gets(str2);
12
13     strcat(str1, str2);
14
15     printf("Concatenated string: %s\n", str1);
16
17     return 0;
18 }
```

Output:



```
"C:\Users\BNT\OneDrive\Desktop" X + v
Enter the first string: Hello
Enter the second string: World
The concatenated string is: HelloWorld

Process returned 0 (0x0)   execution time : 18.638 s
Press any key to continue.
```

Discussion:

This program successfully demonstrates how to concatenate two strings in C using the `strcat()` function. It takes user input, combines the strings, and displays the result. Proper memory allocation and safe input methods are important for handling larger or unexpected inputs. Overall, the task shows basic string handling and user interaction in C programming.