String in C Programming

The String can be defined as the one-dimensional array of ~~elements~~ characters terminated by a null ('\0'). The character array or the string is used to manipulate text such as word or sentences. Each character in the array occupies one byte of memory, and the last character must always be 0.

The termination character ('\0') is important in string since it is the only way to identify where the string ends.

String Decearation in C

(1) By char array

(2) By String Literal :- is a sequence of zero or more multibyte characters enclosed in double quotes, as in "abc". String literals are not modifiable.

① string By char array

```
char ch[10] = {'I', 'n', 'd', 'r', 'a', 'n', 'i', 'l', '\0'};
```

Index →	0	1	2	3	4	5	6	7	8
	I	n	d	r	a	n	i	l	\0
address →	100	101	102	103	104	105	106	107	108

② Assigning character by character without Size

we can assign character by character without size with the NULL character at the end. The size of the string is determined by the compiler automatically.

```
char ch[] = {'I', 'n', 'd', 'r', 'a', 'n', 'i', 'l', '\0'};
```

③ Assigning a String literal without Size

```
char name[] = "Indranil";
```

④ Assigning a String Literal with a Predefined Size

```
char name[100] = "Indranil";
```

```
#include<stdio.h>
#include<string.h>
int main(){
    char name1[12] = {'I', 'N', 'D', 'R', 'A', 'N', 'I', 'L', '\0'};
    char name2[15] = "INDRANIL";
    Printf("%s\n", name1);
    Printf("%s\n", name2);
    return 0;
}
```

name1

0	1	2	3	4	5	6	7	8	9	10	11
I	N	D	R	A	N	I	L	\0			

name2

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
I	N	D	R	A	N	I	L	\0						

Traversing String

Two ways to traverse a String

(1) By using the length of the String
(2) By using the null character

using the length of the String

```
/* C program to counting the number of Vowels in a String */
#include <stdio.h>
void main() {
    char s[11] = "Indranil";
    int i = 0;
    int count = 0;
    while (i < 11) {
        if (s[i] == 'a' || s[i] == 'e' || s[i] == 'i' || s[i] == 'o' || s[i] == 'u' ||
            s[i] == 'A' || s[i] == 'E' || s[i] == 'I' || s[i] == 'O' || s[i] == 'U') {
            count++;
        }
        i++;
    }
    printf("The number of Vowels = %d", count);
}
```

using the NULL character

Same program same line of code just minor change

```
while (s[i] != NULL);
```

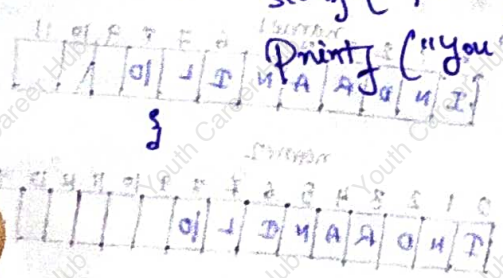
Accepting String as the input

```
#include <stdio.h>
void main() {
    char s[20];
    printf("Enter a String:\n");
    scanf("%s", s);
    printf("You entered %s", s);
}
```

the above code will not work for space separated strings. To make this code working for the space separated strings, the minor change required in the scanf function.

```
scanf("%d[^\n]s", s)
```

which instructs the compiler to store the string s while the new line ('\n') is encountered.



gets() and puts() functions in C

The gets() and puts() are declared in the header file stdio.h. Both the functions are involved in the input/output operations of the string.

C gets() function

The gets() function enables the user to enter some characters followed by the enter key. All the characters entered by the users get stored in a character array. The gets() allowed the user to enter the space-separated strings.

```
#include<stdio.h>
int main()
{
    char s[20];
    printf("Enter a string\n");
    gets(s);
    printf("%s", s);
    return 0;
}
```

* The gets() function is risky to use since it does not perform any array bound checking and keep reading the characters until the new line (enter) is encountered.

fgets() in c Language

For reading a string value with lines from the specified stream and stores it into the string pointed to by str. It stops when either (n-1) characters are read, the newline character is read, or the end of file is reached.

Syntax

```
char *fgets(char *str, int n, FILE *stream)
```

Example Let's say the maximum no of characters are 15 and input length is greater than 15 but still fgets() will read only 15 character and print it

```
#include<stdio.h>
int main(){
    char str[15];
    fgets(str, 15, stdin);
    printf("%s\n", str);
    return 0;
}
```

C puts() function

The puts() function is very much similar to printf() function. The puts() function is used to print the string on the console which is previously read by using gets() or scanf() function.

```
#include <stdio.h>
#include <string.h>
int main()
{
    char name[50];
    printf("Enter your name: \n");
    gets(name);                    //reads string from user
    printf("your name is: ");
    puts(name);                    //display string
    return 0;
}
```


C String Library functions

① Strlen(string_name) — Returns the length of a string

```
#include <stdio.h>
#include <string.h>
int main(){
    char ch[20]={'I','L','O','V','E','I','N','D','I','A','\0'};
    printf("Length of string %d", strlen(ch));
    return 0;
}
```

② strcpy(destination source) — copies the contents of source string to destination string

```
#include<stdio.h>
#include<string.h>
int main()
{
    char ch[90] = "";
    char ch2[90];
    strcpy(ch2, ch);
    printf("value of second string %d", ch2);
    return 0;
}
```

(3) strcat(first_string, second_string) — Concatenates and joins first string with second string.

```
#include<stdio.h>
#include<string.h>
int main()
{
    char ch[20]={'h','e','l','l','o','\0'};
    char ch2[10]={'W','I','L','L','\0'};
    strcat(ch,ch2);
    printf("Value of first string is %s", ch);
    return 0;
}
```

(4) strcmp(first_string, second_string) — Compares the first string with second string. If both strings are same, it return 0.

```
#include<stdio.h>
#include<string.h>
int main()
{
    char str1[20], str2[20];
    printf("Enter 1st string: ");
    gets(str1);
    printf("Enter 2nd string: ");
    gets(str2);
    if(strcmp(str1,str2)==0){
        printf("Strings are equal");
    }else{
        printf("Not equal");
    }
    return 0;
}
```

⑤ strrev(string) — Returns Reverse String

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str[20], rev_string;
    printf("Enter a String:");
    gets(str);
    printf("String %s", str);
    rev_string = strrev(str);
    printf("Reverse String %s", rev_string);
    return 0;
}
```

⑥ strlwr(string) - Returns string characters in Lower case

```
lower = strlwr(str);
```

⑦ strupr(string) - Returns String characters in upper case

```
upper = strupr(str);
```

V. V. I QUESTIONS

what is String in c? Explain with example? write a c program to reverse a String given as user input? Define String Literal? wright a short note about this following string function —

gets(), puts(), fgets(), strlen(), strcpy(), strcat(), strcmp(), strrev()