

#13 Strings

By Saurabh Shukla | 2017©mysirg.com

String

The string in C programming language is managed in a one-dimensional array of characters. String is a sequence of characters terminated by a nul character '\0'. Nul character is a non-printing symbol with ASCII code 0.

The following declaration creates a char array and initializing it with a single word string "Bhopal". To hold the nul character at the end of the array, the size of the character array should be at least one more than the number of characters in the word "Bhopal"

- `char city[7]={'B','h','o','p','a','l','\0'};`

Another way to initialize a string in an array is as follows:

- `char city[7]="Bhopal";`

Example

```
1  #include<stdio.h>
2  int main()
3  {
4      char city[7] = "Bhopal";
5      int i=0;
6      while(city[i]!='\0')
7      {
8          printf("%c",city[i]);
9          i++;
10     }
11     return(0);
12 }
13
```

- String constant is enclosed in double quotes.
- Char array of size 7 is created with the name city.
- The variable i is initialized to 0.
- `city[i]` is i^{th} variable in the array, where i is index.
- While loop iterate the `printf` and `i++` statements until nul character encounters in the array. Initially `city[i]` is 'B' as the value of i is 0. The value of i is incremented every time after printing the character stored in `city[i]`.
- Nul character is automatically stored at the 6th index of array city.

- The output is
Bhopal

About %s format specifier

Just like %d, %c, %f, you have another format specifier %s. It is used to specify the format of data as string. C style string is also known as nul terminated string.

Example

```
1  #include<stdio.h>
2  int main()
3  {
4      char text[6] = "Hello";
5      printf("%s",&text[0]);
6      return(0);
7  }
8
```

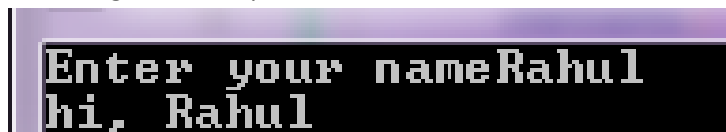
- text is a char array, initialized with a string "Hello".
- In printf, %s format specifier is used, which is responsible to tell printf that the data is string.
- In the previous example, format specifier was %c, which is used to print one character. To print a string, you need to print all characters of the string one by one, so loop was required. This time format specifier %s is used, which prints the whole string in one go.
- Note that in printf, you have to pass address of very first block of the array when %s is used.
- You can write simply printf("%s",text); (text in place of &text[0])
- Name of the array always represents the address of the first block of the array.
- Why you have to pass an address in printf with %s? You will learn about addresses in the pointers chapter. For now, address is a logical number (integer), which is nothing but a sequence number of bytes in the program's memory. Array variables always consume memory in a sequence. &text[0] or just text, is the representation of address of first byte of the array. One more to this address is the address of next byte, that is &text[1]. Function printf has to print all characters of the string and not just one character. You have given an address of the first block of the array to the function printf, which can be used to access text[0]. The best part is, printf can increment the address so that it can access the next block of the array. This continues until nul character is encountered.

Taking input from user

Example

```
1  #include<stdio.h>
2  int main()
3  {
4      char text[20];
5      printf("Enter your name");
6      scanf("%s",&text[0]);
7      printf("hi, %s",&text[0]);
8      return(0);
9  }
10
```

- User is asked to enter his name.
- Scanf is used to take string as an input in the same way, you used it for integers or float values.
- Last printf is used to verify, whether input is properly stored in the array or not.
- Following is the sample run



```
Enter your nameRahul
hi, Rahul
```

- Again



```
Enter your nameRahul Gupta
hi, Rahul
```

- This time we have entered "Rahul Gupta" which is a two word string. Printf function prints only the first word that is the name but not the second word, why?
- It is because scanf is not capable to input multiword string. Scanf reads characters from input buffer, character by character. Scanf consider three characters as delimiter or data separator symbols, they are space, tab and new line character. Rahul is followed by a space character which forces scanf to stop scanning from the buffer.
- To get rid of this problem, use function gets() in place of scanf.
- gets() can input only string constants, so no need to mention %s specifier.
- gets() can input multiword string. The only delimiter it understands is new line character.
- gets() can input only one string at a time. So if you want to input two strings, you have to call gets() two times.
- Next example is the same program but replaces the use of scanf() with gets()

Example

```
1  #include<stdio.h>
2  int main()
3  {
4      char text[20];
5      printf("Enter your name");
6      gets(&text[0]);
7      printf("hi, %s",&text[0]);
8      return(0);
9  }
```

Predefined functions

There are several predefined functions declared in string.h header file can be used to manipulate strings. Discussing some of them is of worth use.

- strlen()
 - prototype: int strlen(char *);
 - It calculates the length of the given string

```
1  #include<stdio.h>
2  int main()
3  {
4      int len;
5      char str[20];
6      printf("Enter a string");
7      gets(str);
8      len=strlen(str);
9      printf("Length is %d",len);
10     return(0);
11 }
```

- strrev()
 - prototype: char* strrev(char*);
 - It reverses the given string

```
1  #include<stdio.h>
2  int main()
3  {
4      char str[20];
5      printf("Enter a string");
6      gets(str);
7      printf("You entered %s",str);
8      strrev(str);
9      printf("Reverse is %s", str);
10     return(0);
11 }
```

- `strlwr()`
 - prototype: `char* strlwr(char*);`
 - It converts the given string into its corresponding lower case

```
1  #include<stdio.h>
2  int main()
3  {
4      char str[20];
5      printf("Enter a string");
6      gets(str);
7      strlwr(str);
8      printf("String in lower case is %s",str);
9      return(0);
10 }
```

- `strupr()`
 - prototype: `char* strupr(char*);`
 - It converts the given string into its corresponding upper case

```
1  #include<stdio.h>
2  int main()
3  {
4      char str[20];
5      printf("Enter a string");
6      gets(str);
7      strupr(str);
8      printf("String in upper case is %s",str);
9      return(0);
10 }
```

- `strcpy()`
 - prototype: `char* strcpy(char*, char*);`
 - It copies the second string into first

```
1  #include<stdio.h>
2  int main()
3  {
4      char str1[20], str2[20];
5      printf("Enter a string");
6      gets(str1);
7      strcpy(str2,str1);
8      printf("String1=%s and String2 = %s",str1,str2);
9      return(0);
10 }
```

- `strcmp()`
 - prototype: `int strcmp(char*, char*);`
 - It compares two strings and return -1, 0 or 1, indicative values. Return 0 if strings are identical. Returns -1 if first string appears before the second string in the dictionary. Returns 1 if first string appears after the second string in the dictionary.

```
1  #include<stdio.h>
2  int main()
3  {
4      char str1[20], str2[40];
5      printf("Enter two strings");
6      gets(str1);
7      gets(str2);
8      strcat(str2, str1);
9      printf("String1=%s and String2 = %s", str1, str2);
10     return(0);
11 }
```

- `strcat()`
 - prototype: `char* strcat(char*, char*)`;
 - It simply appends or concatenates the two strings.

```
1  #include<stdio.h>
2  int main()
3  {
4      int r;
5      char str1[20], str2[20], str3[20];
6      printf("Enter three strings");
7      gets(str1);
8      gets(str2);
9      gets(str3);
10     r=strcmp(str2, str1);
11     printf("%d", r);
12     r=strcmp(str1, str3);
13     printf("%d", r);
14     return(0);
15 }
```

Two dimensional character arrays

Two dimensional char array can be used to handle multiple strings. One char array of single dimension is used to store one string. Two dimensional arrays is nothing but array of one dimensional array

Example

```
1  #include<stdio.h>
2  int main()
3  {
4      int i;
5      char str[5][10];
6      printf("Enter 5 strings");
7
8      for(i=0;i<5;i++)
9          gets(str[i]);
10
11     for(i=0;i<5;i++)
12         printf("\nString %d = %s ",i+1,str[i]);
13 }
```

- In the above program, we have created a two dimensional char array, which is a collection of 5 one dimensional char arrays, each of length 10.
- Printf is used to ask user to enter 5 strings.
- Function gets() needs to be called 5 times to input five strings. Thus we put gets() in a loop.
- str[i] denotes the address of ith one dimensional array (i starts from 0 to 4).
- Last printf is executed 5 times as we put it in the loop. It prints the string number and string.

References:

YouTube video links

- Lecture 13 Strings in C part-1
 - <https://youtu.be/9HCOoGtCPQI?list=PL7ersPsTyYt2Q-SqZxTA1D-melSfqBRMW>
- Lecture 13 Strings in C part-2
 - <https://youtu.be/jMEXa6BBj00?list=PL7ersPsTyYt2Q-SqZxTA1D-melSfqBRMW>

Exercise

- 1) Write a program to count the occurrence of a character in a given string. String and character is taken from user.
- 2) Write a program to find number of vowels in a given string.
- 3) Write a program to count number of words in a given string
- 4) Write a program to check whether a given string is palindrome or not.
- 5) Write a program to calculate length of string.(with and without strlen())
- 6) Write a program to reverse a string. (with and with and without strlen()).

- 7) Write a program to convert string into lower case (with and without `strlwr()`)
- 8) Write a program to convert string into upper case (with and without `strupr()`).
- 9) Write a program to copy a string into another char array (with and without `strcpy()`).
- 10) Write a program to concatenate a string to another string (with and without `strcat()`)
- 11) Write a program to compare two strings (with and without `strcmp()`).
- 12) Write a program to perform case insensitive comparison of two strings.
- 13) Write a program to check whether a given string is alphanumeric or not
- 14) Write a program to remove adjacent duplicate characters from a string
- 15) Write a program that takes 5 strings from user, store them in two dimension char array, now find the number of occurrence a specific character (entered by user) in 5 entered strings
- 16) Write a program to find index of first occurrence of given character in a given string
- 17) Write a program to reverse a string word wise.
- 18) Write a program to remove extra spaces from a given string
- 19) Write a program to search a given pattern in a given string
- 20) Write a program that takes 5 strings from user, store them in two dimension char array, now display them in dictionary order
- 21) Write a program to abbreviate the name of a person. Example Rohit Kumar Bhargav should become R. K. Bhargav
- 22) Write a program to convert digits of a number into words. [For example 234 should become Two Four Three]