STRINGS

BY: RICHA JAIN

Strings

- Just like group of integers can be stored in an integer array, similarly, a group of characters can also be stored in a character array.
- This array of characters is known as a string.
- Strings are used to used for manipulating and storing text such as words and sentences.

Fundamentals of strings

- Strings are arrays of characters
  - String is a **pointer** to first character (like array)
  - Value of string is the **address** of first character
- Each element of the string is stored in a **contiguous** memory locations.
- Terminated by a **null character**('\0') which is automatically inserted by the compiler to indicate the **end of string**.

- A string is always enclosed in double quotes and may include characters such as alphabets, numbers, escape sequence , blank space and special characters.
- Eg: “Good morning” , “41-B”,

  “121+31-43”,

  ”B”

  “ “

String Definition

- They are defined as

```
char array_name[size];
e.g. char carname[30];
or   char *carname;
```

- It defines an array name and reserves 30 bytes for storing characters and single character consumes 1 bytes each.
- Since the last byte is used for storing null character so total number of character specified by the user cannot exceed **29**.

String Initialization

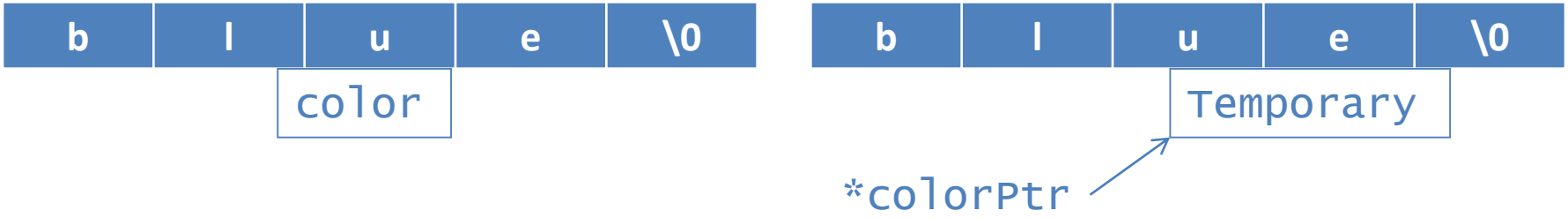
- String Initialization
  - Two ways:
  - Define as a character array or a variable of type char *

```
char color[] = "blue"; //char array
```

Or

```
char color[] = { 'b', 'l', 'u', 'e', '\0' };
char *colorPtr = "blue"; //pointer variable
```

  - Remember that strings represented as character arrays end with '\0'



```
char *colorPtr = "blue"; //pointer variable
printf("%s", colorPtr);
```

Is correct way to use pointer to char.

But following is wrong:

```
char *colorPtr; //pointer variable
scanf("%s", &colorPtr); /* invalid statement %s don't
   work with pointer to char */
```

Strings in C(Reading and Writing a string)

```
#include<stdio.h>
 #include<conio.h>
 int main()
 {
    char name[20];
    printf("enter the name of the person");
    scanf("%s",&name);
    printf("enter name is %s",name);
    getch();
    return 0;
    }
```

How?

- The last program will print only a single word not the sentences with white spaces?
- That is if input is

  Lovely Professional University

- Output will be:

  Lovely

- So how to print:

  Lovely Professional University

use gets and puts

Standard I/O Library Functions

- List of functions in `#include<stdio.h>`
- Used for string input/output functions.

Function	Description
gets(char *s);	Inputs characters from the standard input into the array s until a **newline or end-of-file** character is encountered. A terminating null character is appended to the array.
puts(const char *s);	Prints the string s followed by a newline character.

Program to print strings with white spaces using library functions

```
#include <stdio.h>
 main()
{

 char  name[100]; //string char array

 puts("\nEnter a string: ");
 gets(name); //to input string with space

 printf("\nString is: ")
 puts(name); //to output const string

}//end main
```

Output

```
Enter a string:
Lovely Professional University
String is:
Lovely Professional University
```

```c
#include <stdio.h>

main()
{

 char  name[]={"Lovely Professional
University"}; //string char array
 int i=0;

 while(name[i]!='\0') //untill null character
  {
   printf("%c", name[i]);
   i++;
  }//end while

}//end main
```

Program to print strings character by character using loop.

```
Lovely Professional University
```

Output

- If we are interested to read a string consisting of blank characters as well e..g we want to read a line of text, gets function is the easiest way to do it.
- Using header file string.h

Standard string functions

- Strlen :-determines length of string
- Strcpy :-copies a string from source to destination
- Strncpy:-copies char of string to another string upto specific length
- Strcmp:-compare char of 2 strings
- Stricmp/strcmpi:-compare 2 strings
- Strncmp:-compare char of 2 strings upto specific length
- Strnicmp:-compare char of 2 strings upto specific length .Ignore case.

Strlen function

It counts the number of characters in a given string.

To count no of chars in a given string

```
main()
{
char str[10];
int length;
printf(“enter string”);
gets(str);
length=strlen(str);
printf(“length of string=%d”,length);
}
```

Strcpy function

This function copies the contents of 1 string to another.

strcpy(s2,s1);

S1 =source string

S2 =destination string

S1 is copied to s2.

To copy contents of 1 string to other

```
main()
{
char s1[10],s2[10];
printf(“enter string”);
gets(s1);
strcpy(s2,s1);
printf(“first string”,s1);
printf(“second string”,s2);
}
```

Copy contents upto a specific length

```
main()
{                                                    ///strncpy fun// str1[10],str2[10];
int n;
printf(“enter source string”);
gets(str1);
printf(“enter destination string”);
gets(str2);
printf(“enter no. of char to be replaced”);
scanf(“%d”,&n);
strncpy(str2,str1,n);
printf(“first string”,str1);
printf(“second string”,str2);
}
```

Strcmp function

- This function compares 2 strings.it compares 2 strings and also check the upper case and lower case. if strings are same then it returns to 0 otherwise non-zero value

```
diff =strcmp(str1,str2);
if(diff= =0)
puts(“strings are equal”);
else
puts(“strings are not equal”);
```

Strncmp function

strncmp(source,target,argument)

```
main()
{
char sor[10],tar[10];
int n,diff;
printf(“enter first string”);
gets (sor);
printf(“enter second string”);
gets(tar);
printf(“enter length upto which comp is made”);
scanf(“%d”,&n);
diff =strncmp(sor,tar,n);
if (diff = =0)
puts(“strings are same upto %d characters”,n);
else
puts(“two strings are different”);
}
```

Strlwr and strupr

```
main()
{
char a[15];
printf("enter string in upper
    case");
gets(a);
printf("in lower case string is:-
    %s",strlwr(a));
}
```

```
main()
{
char a[15];
printf("enter string in lower
    case");
gets(a);
printf("in upper case string is:-
    %s",strupr(a));
}
```

Strchr function

It returns the pointer to a position in the first occurrence of the char in given string.

```
main()
{
char str[20],ch,*p;
printf("enter text");
gets(str);
printf("enter text to find");
ch=getchar();
p=strchr(str,ch);
if(p)
printf("char %c found in string",ch);
else
printf("char %c not found in string",ch);
```

Strcat,strncat function

```
main()
{
char s1[10],s2[10];
puts(“enter text 1”);
gets(s1);
puts(“enter text 2”);
gets(s2);
strcat(s1,s2);
printf(“%s”,s1);
}
```

```
main()
{
char s1[10],s2[10],n;
puts(“enter text 1”);
gets(s1);
puts(“enter text 2”);
gets(s2);
puts(“no of char to add”);
gets(n);
strcat(s1,” “);
strncat(s1,s2,n);
printf(“%s”,s1);
}
```

Searching : strstr

- ***Function strstr*** **<string.h>**

```
char *strstr(const char *s1, const char *s2);
```

- Function **strstr searches for the first occurrence of its second string argument in its first** string argument.
- If the second string is found in the first string, a pointer to the location of the string in the **first string** argument is returned.
- Otherwise, a **NULL pointer** is returned

```
#include <stdio.h>
#include <string.h>

main()
{
 char string1[] = "abcdefabcdef"; /* string to search */
 char string2[] = "def"; /* string to search for */

 printf( "string1 = %s string2 = %s", string1, string2);
 printf( "The remainder of string1 beginning with the
first occurrence of string2 is: %s",
strstr(string1, string2));

getch();
}
```

Program searches for the first occurrence of string2 in string1.

```
string1 = abcdefabcdef
string2 = def

The remainder of string1 beginning with the
first occurrence of string2 is: defabcdef
```

Strrev()

- Strrev()
- To reverse the string s1.

```
strrev(s1);
```

returns the value of s1

Function prototype	Function description
strcpy(s1, s2)	Copies string s2 into array s1. The value of s1 is returned.
strncpy(s1, s2, n)	Copies at most n characters of string s2 into array s1. The value of s1 is returned.
strcat(s1, s2)	Appends string s2 to array s1. The first character of s2 overwrites the terminating null character of s1. The value of s1 is returned.
strncat(s1,s2, n)	Appends at most n characters of string s2 to array s1. The first character of s2 overwrites the terminating null character of s1. The value of s1 is returned.
x = strcmp(s1,s2)	Compares string s1 to s2, Returns a negative number if s1 < s2, and zero if s1 == s2 and a positive number if s1 > s2
x = strncmp(s1,s2,n)	Compares first n characters s1 to s2, Returns a negative number if s1 < s2, and zero if s1 == s2 and a positive number if s1 > s2
l = strlen(s1);	returns the number of characters in the string excluding null character.

Count Length Of String

```
#include<stdio.h>
#include<conio.h>
void main()
{
    char ch[10];
    int i=0;
    gets(ch);
    while(ch[i]!='\0')
    {
        i++;
    }
        printf("%d",i);
        getch();    }
```