



STRING

MODULE 3 PART B

String

- Sequence of characters that is treated as a single data item..
- String is represented using double quotation marks.

Examples : “Hello world”, “xyz123@”, “Good”

- Strings in C are represented by array of characters.
- The end of the string is marked with a special character, the null character, which is simply the character with the ASCII value 0.
- ‘\0’ represents the end of the string. It is also referred as String terminator & Null Character

Declaration of string

- General form for declaration of a string variable:

```
char string_name[size];
```

Example:

```
char city[10];
```

```
char name[30];
```

Initialization of string

- ▶ Two forms are there:

Form1 : `char city [9] = "NEW YORK";`

Size = 8+1

Form2: `char city [9] = {'N','E','W','Y','O','R','K','\0'};`

Initialization of string

char city [] = “NEW YORK”;

Size = 8+1 (Automatically determined by compiler)

char city [] = {‘N’,‘E’,‘W’,‘ ‘,‘Y’,‘O’,‘R’,‘K’,‘\0’};

Size = 8+1 (Automatically determined by compiler)

City



```
Char string[10] = "GOOD";
```

Other Declarations that results Error

Example 1: *char string[3] = "good";*

Example 2: *char string[5]; //Cannot Separate the initialization from declaration*
string = "good";

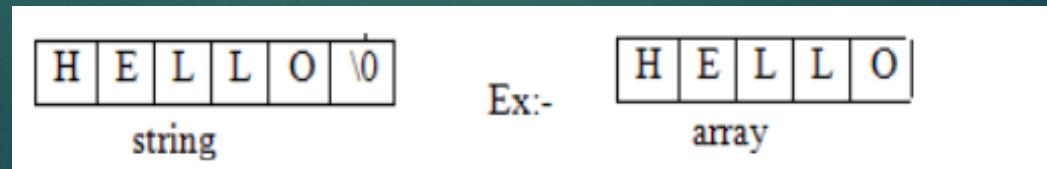
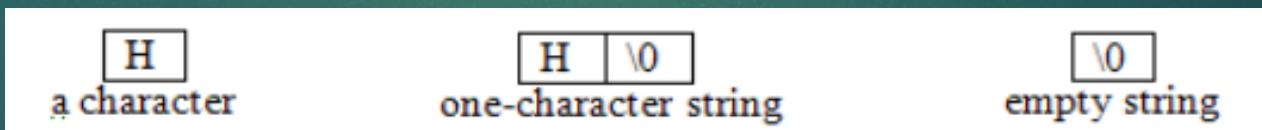
Example 3: *char s1[4] = "abc";*

char s2[4];

s2 = s1; //Array name cannot be used as left operand of assignment operator

Storing the strings in memory

- ▶ A string is stored in array, the name of the string is a pointer to the beginning of the string.
- ▶ The character requires only one memory location.
- ▶ If we use one-character string it requires two locations.
- ▶ The difference is shown below,



Reading strings from terminal

- ▶ String can be read from the user by using three ways:
 - a) `scanf()` function
 - b) `gets()` function
 - c) `getchar()` function

Using Scanf

- Used with %s format specification

```
char address[10]
scanf ("%s", address);
```

Here **don't** use “&” because name of string is a pointer to array.

The problem with scanf() is that it terminates its input on the first white space it finds.

```
#include<stdio.h>
void main()
{
    char name[10];
    printf("Enter the name:");
    scanf("%s",name);
    printf("Name is %s",name);
}
```

Enter the name: Dennis Richie
Name is Dennis

Using Scanf

- ▶ Used with %ws format specification

```
char address[10]  
scanf ("%ws", address);
```

If w is greater or equal than number of characters typed in, the entire string will be stored in string variable.

If w is less than number of characters typed in the string, the excess characters will be truncated and left unread.

```
#include<stdio.h>
void main()
{
    char name[10];
    printf("Enter the name:");
    scanf("%5s",name);
}
```

Enter the name: Dennis Richie

D	E	N	N	I	\0	?	?	?	?
---	---	---	---	---	----	---	---	---	---

```
#include<stdio.h>
void main()
{
char name[30];
printf("Enter the name:");
scanf("%[^\\n]", name);
printf("%s", name);
}
```

```
Enter the name: Hello World
Hello World
```

Using gets()

- ▶ gets() function takes the starting address of the string which will hold the input.
- ▶ string inputted using gets() is automatically terminated with a null character.
- ▶ The C gets function is used to read a line of text from a standard input device and store it in the String variable.
- ▶ When it reads the newline character, then the C gets function will terminate.

```
#include<stdio.h>
void main()
{
char name[20];
printf("Enter the name:");
gets(name);
printf("Name is %s",name);
}
```

Enter the name: Dennis Richie
Name is Dennis Richie

Using getchar()

- ▶ Read successive single characters from the input and place them into a character array.
- ▶ Entire line of text can be read and stored in an array.
- ▶ Reading is terminated when the newline character is entered and the null character is placed at the end of the string.

```
char ch;  
ch = getchar();
```

```
#include <stdio.h>
void main( )
{
char line[81], character;
int c;
c = 0;
printf("Enter text. Press <Return> at end\n");
do
{
character = getchar();
line[c] = character;
c++;
}
while(character != '\n');
c = c - 1;
line[c] = '\0';
printf("\n%s\n", line);
}
```

```
Enter text. Press <Return> at end
sneha sreedevi

sneha sreedevi
```

Copy one string into another and count the number of characters copied

```
#include <stdio.h>
void main( )
{
    int i;
    char string2[30],string1[30];
    printf("Enter a string \n");
    scanf("%s", string2);
    for( i=0 ; string2[i] != '\0'; i++)
        string1[i] = string2[i];
    string1[i] = '\0';
    printf("\n");
    printf("%s\n", string1);
    printf("Number of characters = %d\n", i );
}
```

Sneha

Sneha

Number of characters = 5

Writing Strings To Screen

Using printf()

Using puts()

Using putchar()

Using printf()

- ▶ Used with %s format specification

```
char address[10]  
printf ("%s", address);
```

Using puts()

- Used to print the strings including blank spaces

puts(str);

Example:

```
char message[20] = "Hello world";
puts(message);
```

```
9 #include <stdio.h>
10
11
12 void main()
13 {
14     {
15
16     char name[30];
17     puts("Enter a string ");
18     gets (name);
19     puts("Entered string is");
20     puts (name);
21 }
```

```
Enter a string
Ram is studying in fourth class
Entered string is
Ram is studying in fourth class
```

Using putchar()

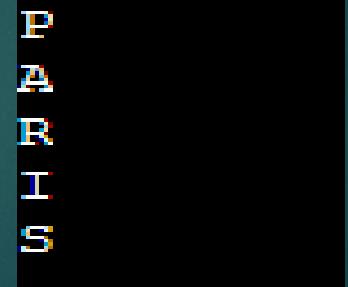
- To print a character on the screen.

char ch = 'A';

putchar (ch);

```
#include <stdio.h>

void main()
{
    char name[6] = "PARIS";
    int i;
    for (i=0; i<5; i++)
    {
        putchar(name[i]);
        putchar('\n');
    }
}
```



P
A
R
I
S

```
#include <stdio.h>

int main()
{
    char string[] = "Welcome to the world of C Programming\n";
    int i=0;
    while(string[i]!='\0')
    {
        putchar(string[i]);
        i++;
    }
    return 0;
}
```

Welcome to the world of C Programming

Putting Strings Together

- ▶ Just as we cannot assign one string to another directly, we cannot join two strings together by the simple arithmetic addition.
- ▶ That is, the statements such as

```
string3 = string1 + string2;  
string2 = string1 + "hello";
```

are not valid.

- ▶ The process of combining two strings together is called concatenation.

Comparison of Strings Together

- C does not permit the comparison of two strings directly. That is, the statements such as

```
if(name1 == name2)  
if(name == "ABC")
```

are not permitted.

- It is therefore necessary to compare the two strings to be tested, character by character.
- The comparison is done until there is a mismatch or one of the strings terminate into a null character, whichever occurs first.

```
#include <stdio.h>

int main()
{
    char str1[30];
    char str2[30];
    int i=0;
    printf("Enter the string1");
    gets(str1);
    printf("Enter the string2");
    gets(str2);
    while(str1[i] == str2[i] && str1[i] != '\0'&& str2[i] != '\0')
    {
        i = i+1;
    }
    if (str1[i] == '\0' && str2[i] == '\0')
        printf("strings are equal\n");
    else
        printf("strings are not equal\n");
    return 0;
}
```

Enter the string1 Pallavi Sneha
Enter the string2Pallavi Padmesh
strings are not equal

Enter the string1 Hello World
Enter the string2 Hello World
strings are equal

String Handling Functions

- ▶ C supports a number of string handling functions.
- ▶ All of these built-in functions are aimed at performing various operations on strings and they are defined in the header file `string.h`.
 - ▶ `strlen()`
 - ▶ `strcpy()`
 - ▶ `strcat()`
 - ▶ `strcmp()`

strlen()

- ▶ Counts and returns the number of characters in a string excluding null character.
- ▶ It takes the form

$$n = \text{strlen}(\textit{string})$$

Example:

```
char str1[] = "WELCOME";
```

```
int n;
```

```
n = strlen(str1);
```

```
#include <stdio.h>
#include<string.h>
int main( )
{
char string[50];
int length;
printf("Enter any string: ");
gets(string);
length=strlen(string);
printf("The length of string=%d", length);
return 0;
}
```

Enter any string: sneha sreedevi
The length of string=14

strcpy()

- This function is used to copy one string to the other.
- Its syntax is as follows:

strcpy(string1,string2);

- where string1 and string2 are one-dimensional character arrays.
- This function copies the content of string2 to string1.

Example:

```
char str1[ ] = "WELCOME";
char str2[ ] = "HELLO";
strcpy(str1,str2);
```

```
#include <stdio.h>
#include<string.h>
int main( )
{
    char city[15];
    strcpy(city, "BANGALORE");
    puts(city);
    return 0;
}
```

BANGALORE

- A program to copy one string to another using strcpy() function

```
#include<stdio.h>
#include<string.h>
int main()
{
char string1[30],string2[30];
printf("Enter first string:");
gets(string1);
printf("\nEnter second string:");
gets(string2);
strcpy(string1,string2);
printf("\nFirst string=%s",string1);
printf("\nSecond string=%s",string2);
return 0;
}
```

```
Enter first string: Hello World

Enter second string: Hai all

First string= Hai all
Second string= Hai all
```

strcat ()

- This function is used to concatenate two strings. i.e., it appends one string at the end of the specified string.
- Its syntax as follows:

strcat(string1,string2);

- where string1 and string2 are one-dimensional character arrays.
- This function joins two strings together.

Example: Example:

```
char str1[20 ] = "HELLO";
char str2[20] = "WORLD";
strcat(str1,str2);
```

```
#include<stdio.h>
#include<string.h>
int main()
{
char string1[30],string2[15];
printf("\n Enter first string:");
gets(string1);
printf("\n Enter second string:");
gets(string2);
strcat(string1,string2);
printf("\n Concatenated string=%s",string1);
return 0;
}
```

Enter first string: Hai all

Enter second string:Welcome to C programming

Concatenated string= Hai allWelcome to C programming

strcmp ()

- ▶ Compares two strings character by character (ASCII comparison) and returns one of three values {-1,0,1}.

Return value	Description
0	When both are equal
<0	If ASCII value of a character of the first string is less than the ASCII value of the character of the second string then function will return negative value
>0	If ASCII value of a character of the first string is greater than the ASCII value of the character of the second string then function will return positive value

Example :

```
int n;  
char city[20] = "MADRAS";  
char town[20] = "MANGALORE";  
n = strcmp(city, town);
```

//ASCII value of D = 68

//ASCII value of N = 78

```
#include<string.h>
int main( )
{
char a[100], b[100];
printf("Enter the first string\n");
gets(a);
printf("Enter the second string\n");
gets(b);
if( strcmp(a,b) == 0 )
printf("Entered strings are equal.\n");
else
printf("Entered strings are not equal.\n");
return 0;
}
```

```
Enter the first string
Delhi
Enter the second string
New Delhi
Entered strings are not equal.
```

strrev()

- strrev() function reverses a given string in C language.

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str[40]; // declare the size of character string
    printf ("\n Enter a string to be reversed: ");
    scanf ("%s", str);

    // use strrev() function to reverse a string
    printf ("\n After the reverse of a string: %s ", strrev(str));
    return 0;
}
```

Enter a string to be reversed: AMBULANCE

After the reverse of a string: ECNALUBMA

Reverse of a string

```
#include<stdio.h>          for(i=0;i<len/2;i++)  
#include<string.h>          {  
int main()                temp = string[i];  
{                           string[i]=string[len-i-1];  
int len,i,j,temp;         string[len-i-1]=temp;  
char string[50];           }  
printf("Enter the string:"); printf("Reverse of the string is %s", string);  
scanf("%[^\\n]",string);   }  
len = strlen(string);
```

Table Of Strings

C	h	a	n	d	i	g	a	r	h
M	a	d	r	a	s				
A	h	m	e	d	a	b	a	d	
H	y	d	e	r	a	b	a	d	
B	o	m	b	a	y				

```
char city[ ] [ ]
{
    "Chandigarh",
    "Madras",
    "Ahmedabad",
    "Hyderabad",
    "Bombay"
}
```

Sorting a string

```
#include<stdio.h>
#include<string.h>
int main()
{
char str[10][50],temp[50];
int i,j,n;
printf("Enter the no of Words to be entered:\n");
scanf("%d",&n);
printf("Enter the words:");
for(i=0;i<n;i++) //Reading
scanf("%s[\^\\n]",str[i]);
```

```
//Sorting
for(i=0;i<n-1;i++)
{
    for(j=i+1;j<n;j++)
    {
        if(strcmp(str[i],str[j])>0)
        {
            strcpy(temp,str[i]);
            strcpy(str[i],str[j]);
            strcpy(str[j],temp);
        }
    }
}
```

```
//Printing  
printf("\n\n lexicographical order: \n");  
for(i=0;i<n;i++)  
    puts(str[i]);  
return 0;  
}
```

```
Enter the no of Words to be entered:  
4  
Enter the words:heap  
stack  
hello  
queue  
  
In lexicographical order:  
heap  
hello  
queue  
stack
```