PROGRAMMING IN C

ASSIGNMENT

ON

STRING FUNCTIONS

Submitted to :

Mrs. Linu Joseph

Submitted by :

Aneetta Paul
Roll no: 12112514
Dept: CS (1st yr BCA)

What is a string?

In C programming, a string is a sequence of characters terminated with a null character \0 or in other words, a string is a data type used in programming, such as an integer and floating point unit, but is used to represent text rather than numbers.

It consists of a set of characters that can also contain spaces and numbers. For example, the word "hamburger" and the phrase "I ate 3 hamburgers" are both strings. Even "12345" could be considered a string, if specified correctly. Typically, programmers must enclose strings in quotation marks for the data to recognized as a string and not a number or variable name.For example:

```
char c[] = "c string";
char c[] = “abc&123”;
```

When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default.

String handling functions

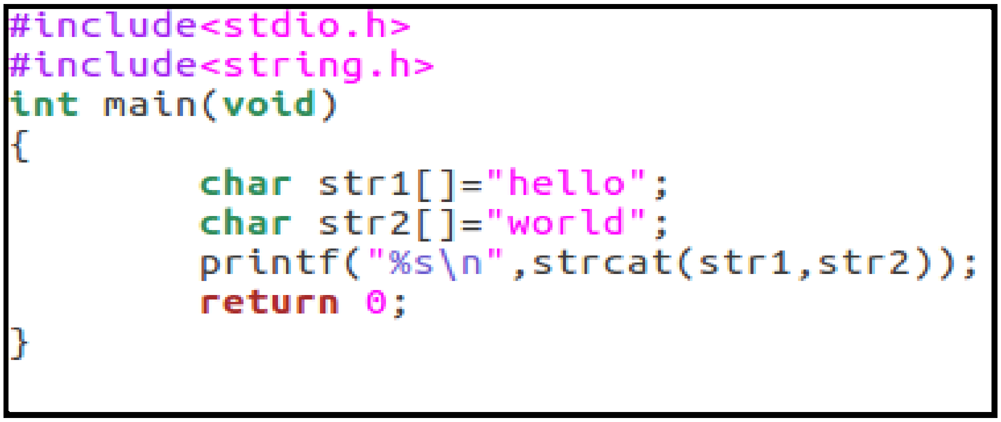
String handling functions can be used to carry out many of the string manipulations. These functions are packed in the **string.h** library. We have to include **string.h** in programs to use these functions. Mostly used string functions are :

1. strcat(): It is used to concatenate(combine) two strings.

Syntax : **`strcat(str1,str2)`**

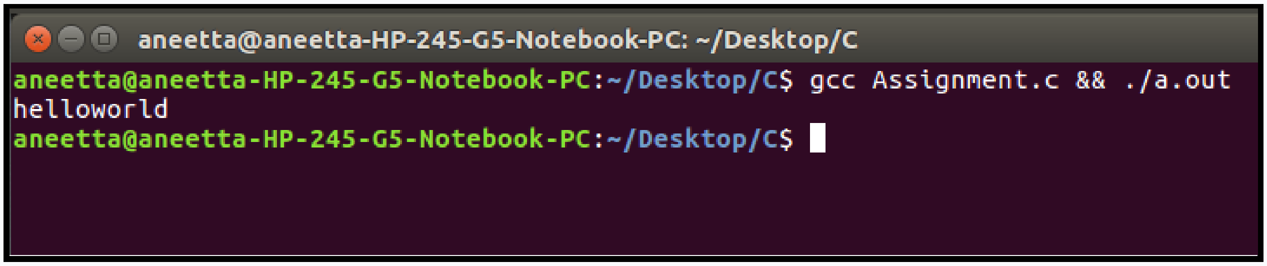
Example :

CODE:



```
#include<stdio.h>
#include<string.h>
int main(void)
{
	char str1[]="hello";
	char str2[]="world";
	printf("%s\n",strcat(str1,str2));
	return 0;
}
```

OUTPUT :



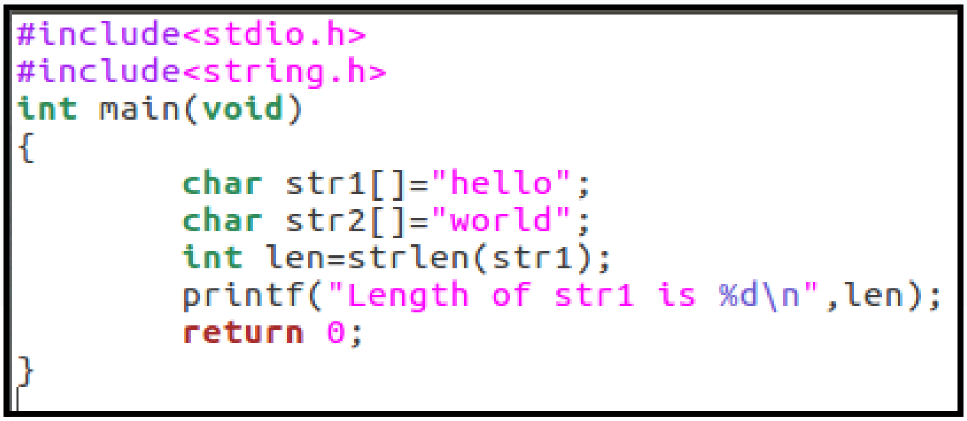
```
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ gcc Assignment.c && ./a.out
helloworld
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ 
```

2. strlen(): It is used to show the length of a string.

Syntax : `strlen(str1)`

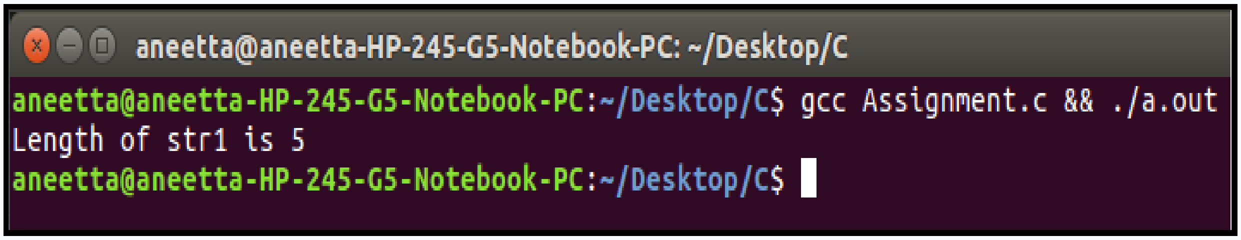
Example :

CODE:



```
#include<stdio.h>
#include<string.h>
int main(void)
{
	char str1[]="hello";
	char str2[]="world";
	int len=strlen(str1);
	printf("Length of str1 is %d\n",len);
	return 0;
}
```

OUTPUT :



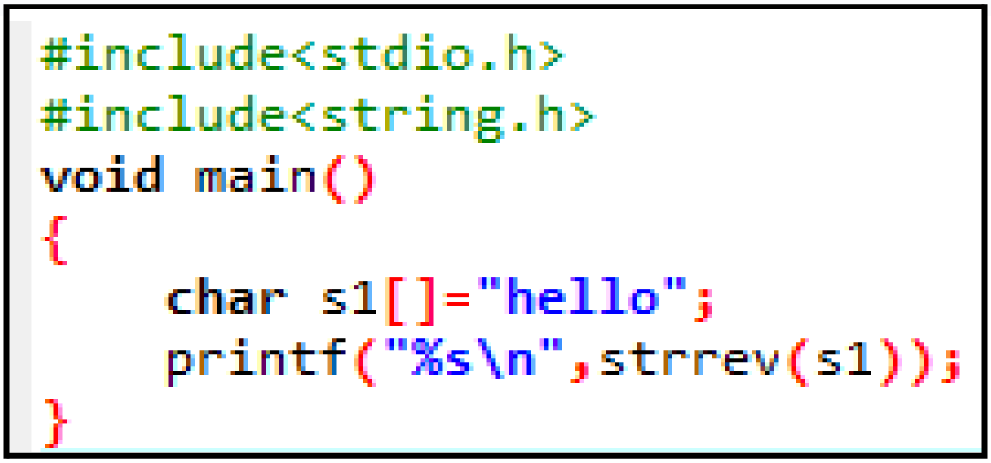
```
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ gcc Assignment.c && ./a.out
Length of str1 is 5
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ 
```

3. **strrev() :** It is used to show the reverse of a string.

Syntax : `strrev(str1)`

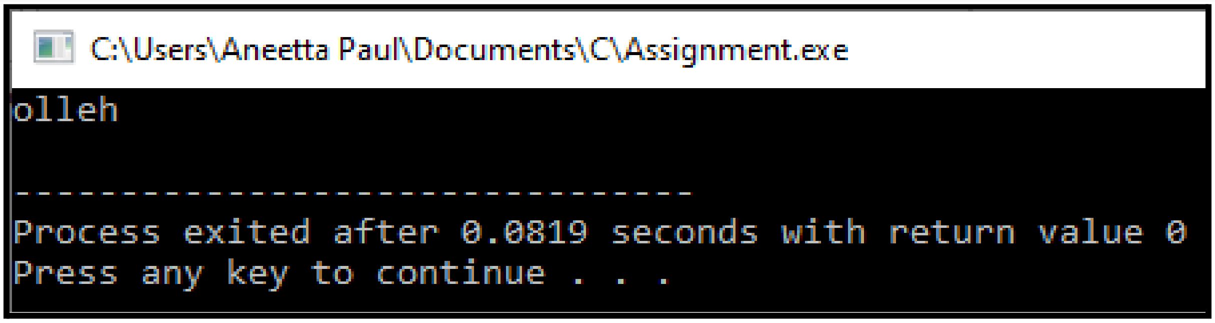
Example :

CODE :



```c
#include<stdio.h>
#include<string.h>
void main()
{
	char s1[]="hello";
	printf("%s\n",strrev(s1));
}
```

OUTPUT :



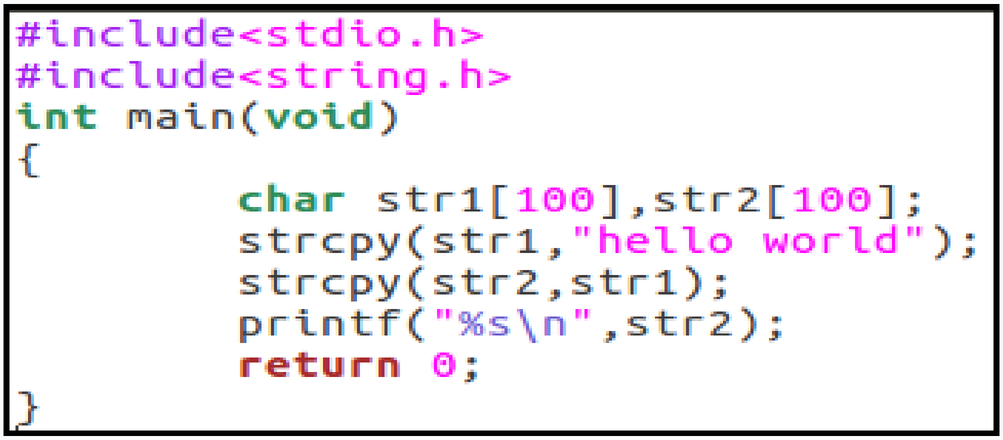
```
olleh

--------------------------------
Process exited after 0.0819 seconds with return value 0
Press any key to continue . . .
```

4. strcpy() : It copies one string into another.

Syntax : `strcpy(str1,str2)`

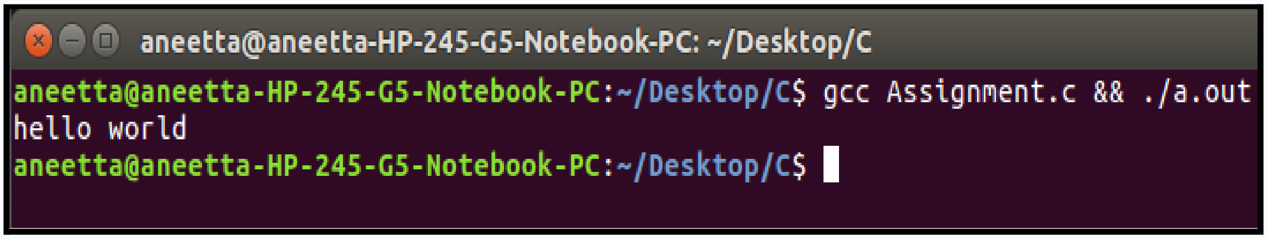
Example :

CODE :



```
#include<stdio.h>
#include<string.h>
int main(void)
{
        char str1[100],str2[100];
        strcpy(str1,"hello world");
        strcpy(str2,str1);
        printf("%s\n",str2);
        return 0;
}
```

OUTPUT :



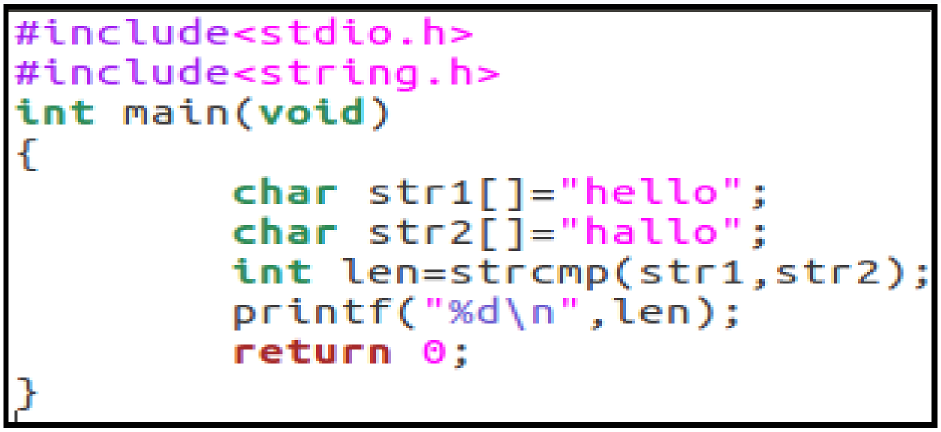
```
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ gcc Assignment.c && ./a.out
hello world
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ 
```

5. strcmp() : It is used to compare two strings. The strcmp() compares two strings character by character. If the strings are equal, the function returns 0, greater than 0 if the first non-matching character in str1 is greater (in ASCII) than that of str2, less than 0 if the first non-matching character in str1 is lower (in ASCII) than that of str2.

Syntax : `strcmp(str1,str2)`

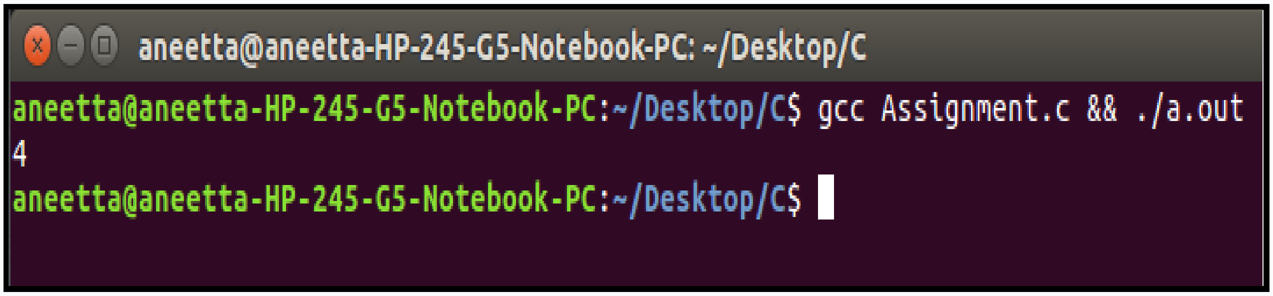
Example :

CODE :



```c
#include<stdio.h>
#include<string.h>
int main(void)
{
        char str1[]="hello";
        char str2[]="hallo";
        int len=strcmp(str1,str2);
        printf("%d\n",len);
        return 0;
}
```

OUTPUT :



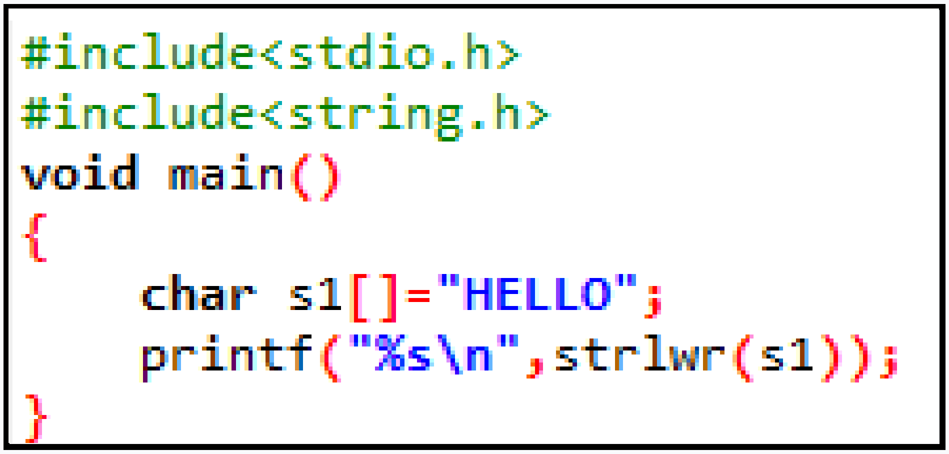
```
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ gcc Assignment.c && ./a.out
4
aneetta@aneetta-HP-245-G5-Notebook-PC:~/Desktop/C$ 
```

6. strlwr() : It is used to convert the input to lowercase.

Syntax : `strlwr(str1)`

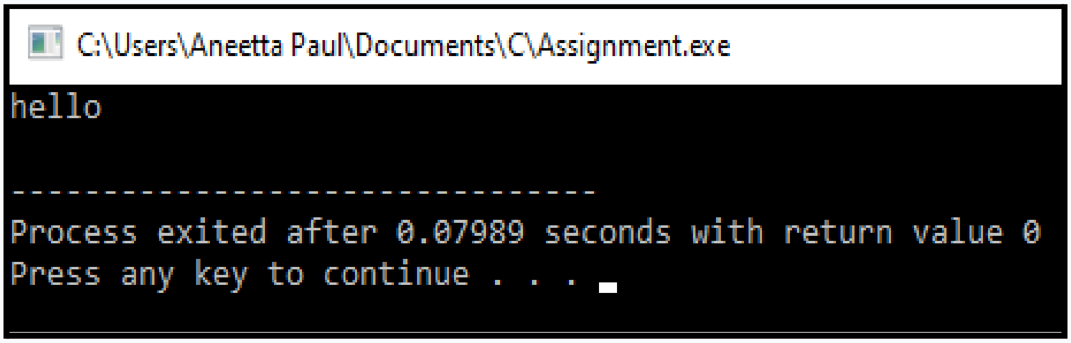
Example :

CODE :



```
#include<stdio.h>
#include<string.h>
void main()
{
	char s1[]="HELLO";
	printf("%s\n",strlwr(s1));
}
```

OUTPUT :



C:\Users\Aneetta Paul\Documents\C\Assignment.exe

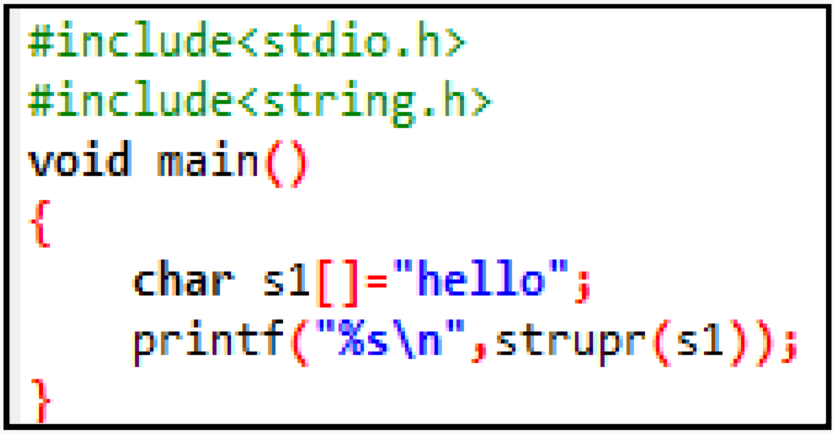
```
hello

--------------------------------
Process exited after 0.07989 seconds with return value 0
Press any key to continue . . .
```

7. strupr() : It is used to convert the input to uppercase.

Syntax : `strupr(str1)`

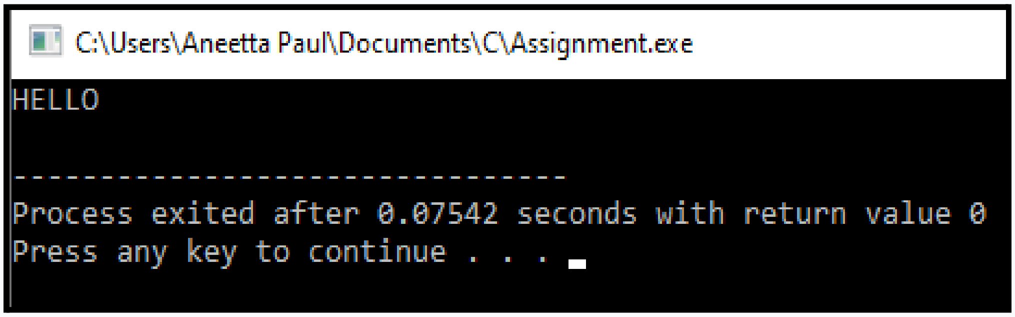
Example :

CODE :



```
#include<stdio.h>
#include<string.h>
void main()
{
	char s1[]="hello";
	printf("%s\n",strupr(s1));
}
```

OUTPUT :



C:\Users\Aneetta Paul\Documents\C\Assignment.exe

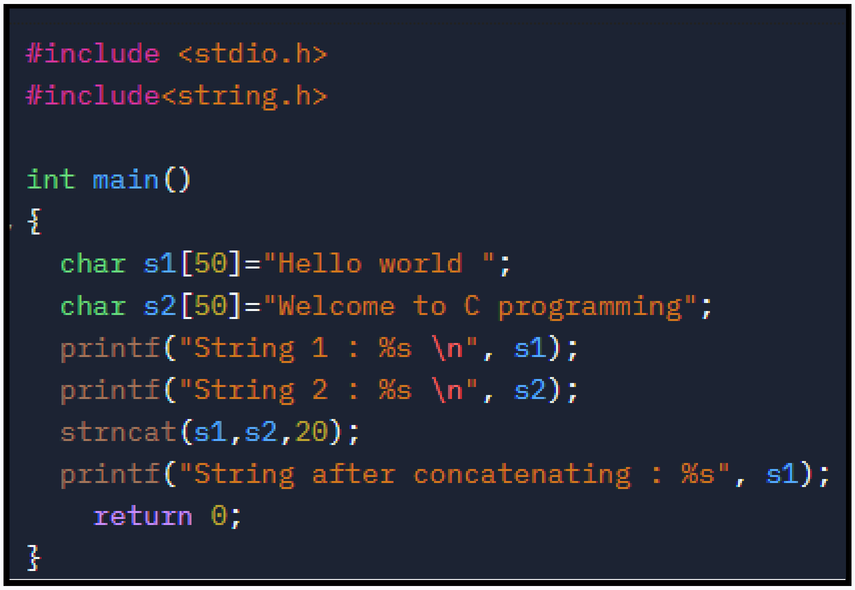
```
HELLO

--------------------------------
Process exited after 0.07542 seconds with return value 0
Press any key to continue . . .
```

8. strncat() : It is used to concatenate n characters of second string to first string.

Syntax : `(str1, str2, n)`

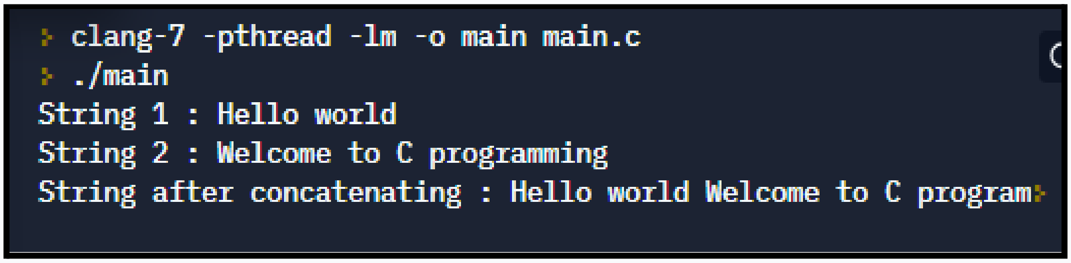
Example :

CODE :



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

int main()
{
  char s1[50]="Hello world ";
  char s2[50]="Welcome to C programming";
  printf("String 1 : %s \n", s1);
  printf("String 2 : %s \n", s2);
  strncat(s1,s2,20);
  printf("String after concatenating : %s", s1);
    return 0;
}
```

OUTPUT :



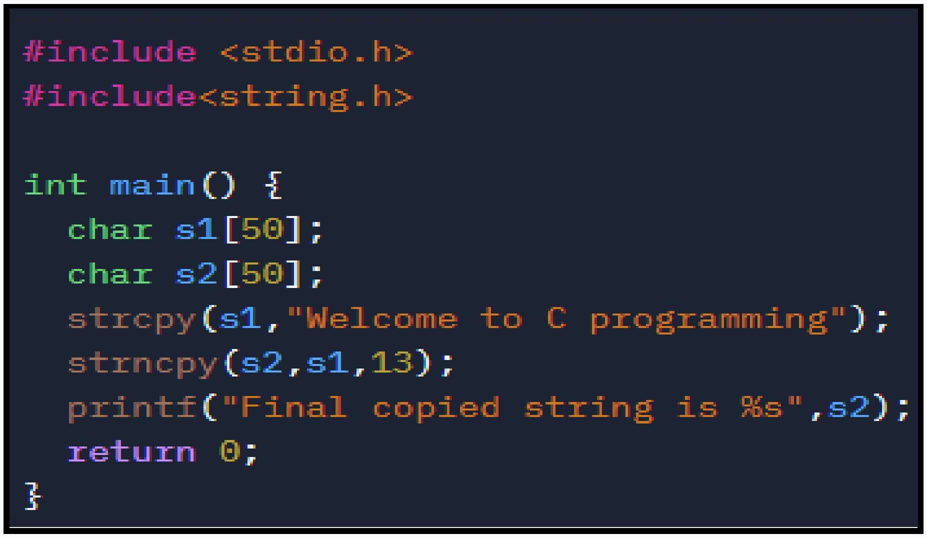
```
› clang-7 -pthread -lm -o main main.c
› ./main
String 1 : Hello world
String 2 : Welcome to C programming
String after concatenating : Hello world Welcome to C program›
```

9. strncpy() : It copies a given number of characters of one string into another.

Syntax : `strncpy(str1, str2, n)`

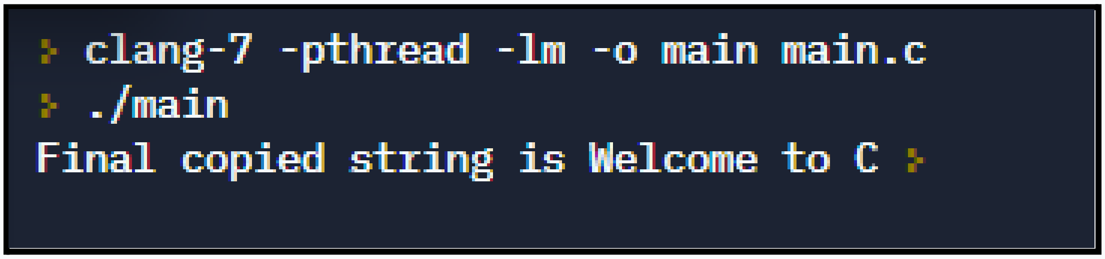
Example :

CODE :



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

int main() {
  char s1[50];
  char s2[50];
  strcpy(s1,"Welcome to C programming");
  strncpy(s2,s1,13);
  printf("Final copied string is %s",s2);
  return 0;
}
```

OUTPUT :



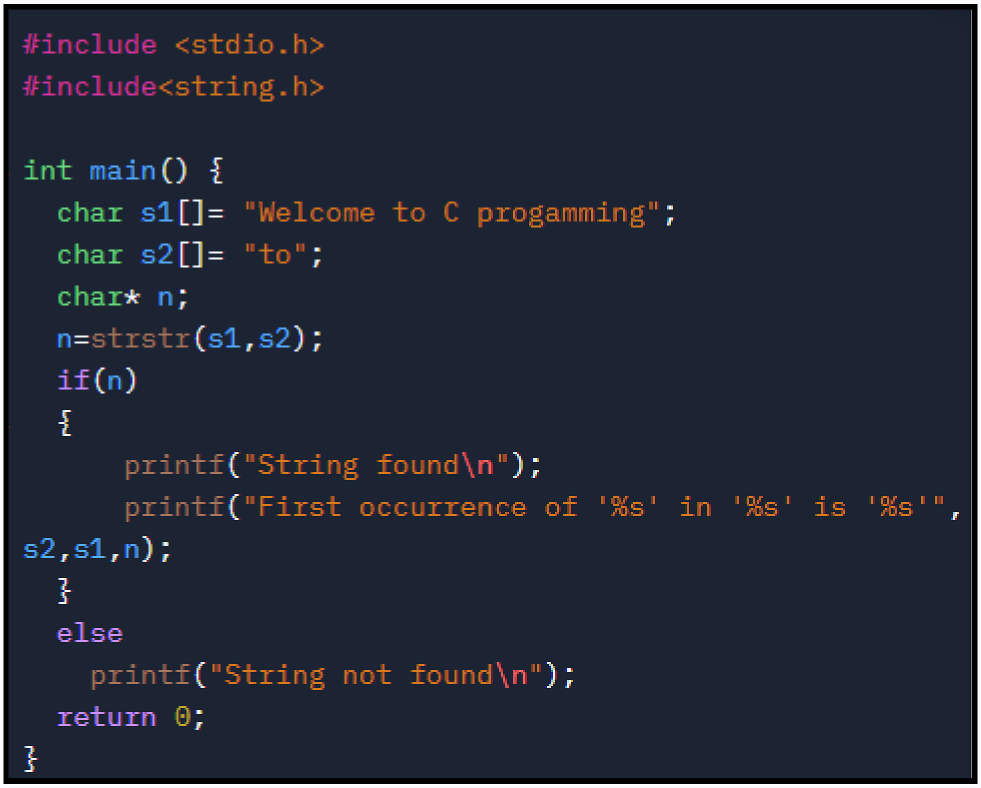
```
› clang-7 -pthread -lm -o main main.c
› ./main
Final copied string is Welcome to C ›
```

10. strstr() : It returns the pointer of the first occurrence of str2 in str1.

Syntax : `strstr(str1,str2)`

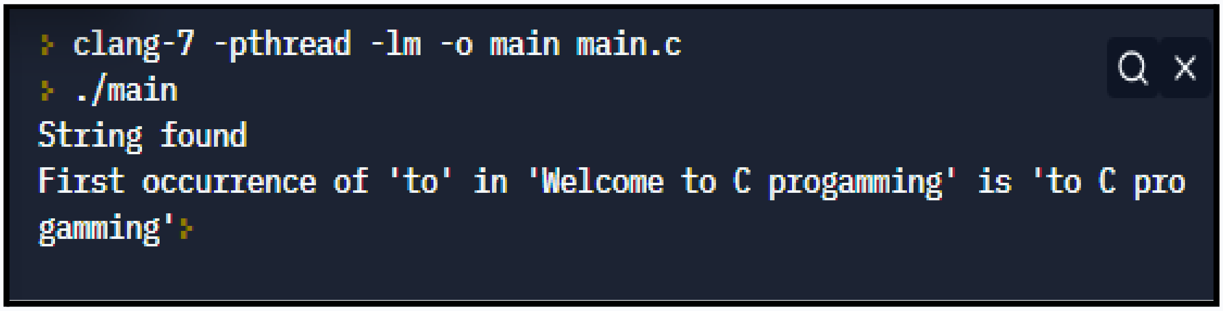
Example :

CODE :



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

int main() {
  char s1[]= "Welcome to C progamming";
  char s2[]= "to";
  char* n;
  n=strstr(s1,s2);
  if(n)
  {
      printf("String found\n");
      printf("First occurrence of '%s' in '%s' is '%s'",
s2,s1,n);
  }
  else
    printf("String not found\n");
  return 0;
}
```

OUTPUT :



```
› clang-7 -pthread -lm -o main main.c
› ./main
String found
First occurrence of 'to' in 'Welcome to C progamming' is 'to C pro
gamming'›
```

****