



# C STRING

**BY:MOAYAD ALMASRY**

# String

```
char c[5] = {'a', 'b', 'c', 'd', '\0'};
```

```
char c[5] = "abcd";
```

طريقة تعريف ال string

هكذا يتم تخزينه داخل المصفوفة حيث  
يتم انهاء السترنج ب null

c[0]	c[1]	c[2]	c[3]	c[4]
a	b	c	d	\0

# String

```
char c[] = "abcd";  
char c[50] = "abcd";  
char c[] = {'a', 'b', 'c', 'd', '\\0'};  
char c[5] = {'a', 'b', 'c', 'd', '\\0'}
```

طرق لتعريف ال string

هكذا يتم تخزينه داخل المصفوفة حيث  
يتم انهاء السترنج ب null

c[0]	c[1]	c[2]	c[3]	c[4]
a	b	c	d	\\0

# Some more examples of string initialization

Initialization	Memory representation										
<code>char animal[]="Lion";</code>	<table><tr><td>L</td><td>i</td><td>o</td><td>n</td><td>'\0'</td></tr></table>	L	i	o	n	'\0'					
L	i	o	n	'\0'							
<code>char location[]="New Delhi";</code>	<table><tr><td>N</td><td>e</td><td>w</td><td></td><td>D</td><td>e</td><td>l</td><td>h</td><td>i</td><td>'\0'</td></tr></table>	N	e	w		D	e	l	h	i	'\0'
N	e	w		D	e	l	h	i	'\0'		
<code>char serial_no[]="A011";</code>	<table><tr><td>A</td><td>0</td><td>1</td><td>1</td><td>'\0'</td></tr></table>	A	0	1	1	'\0'					
A	0	1	1	'\0'							
<code>char company[]="Airtel";</code>	<table><tr><td>A</td><td>i</td><td>r</td><td>t</td><td>e</td><td>l</td><td>'\0'</td></tr></table>	A	i	r	t	e	l	'\0'			
A	i	r	t	e	l	'\0'					

# String


```
char name[20];
```

```
printf("Enter name: ");
```

```
scanf("%s", name);
```

```
printf("Your name is %s.", name);
```

```
return 0;
```



بهذه الطريقة يمكننا ادخال كلمة على السترينج  
وتخزينها فيه

# String

```
char name[20];
```

```
printf("Enter name: ");
```

```
scanf("%s", name);
```

```
printf("Your name is %s.", name);
```

```
return 0;
```

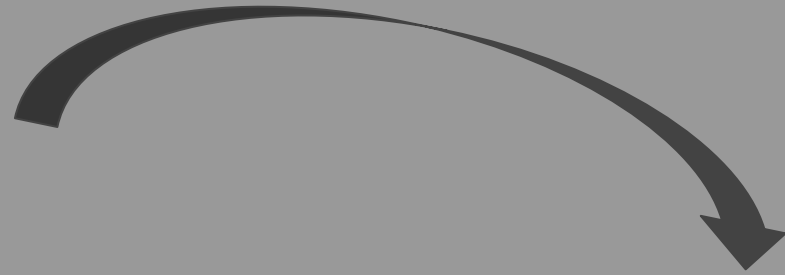
if input:c  
output:c

but if  
input:c cours  
output:c

لو لاحظنا فان scanf  
تتوقف عن القراءة عند  
وجود فراغ

# String

`gets (اسم السترينج) ;`



يعمل فنكشن gets نفس عمل scanf لكن هنا  
يقوم بقراءة السترينج كامل مع الفراغات

`puts (str) ;`



`printf("%s",str)` يعمل فنكشن puts نفس  
وكلاهما يطبع السترينج

`printf("%s",str);`

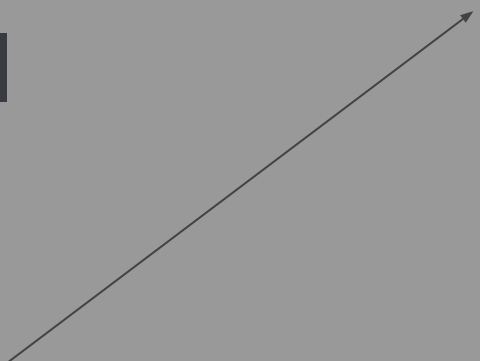
# String

```
#include <stdio.h>
void displayString(char str[]);
```

```
int main(){
    char str[50];
    printf("Enter string: ");
    gets(str);
    displayString(str);
    return 0;
}
```

```
void displayString(char str[]){
    printf("String Output: ");
    puts(str);
}
```

كما لاحظنا يمكننا تمرير  
السترينج الى فنكشن  
ومعاملته نفس معاملته  
ال array العاديه





# String

```
#include <stdio.h>
void displayString(char str[]);
```

```
int main(){
    char str[50];
    printf("Enter string: ");
    gets(str);
    displayString(str);
    return 0;
```

```
}
void displayString(char str[]){
    printf("String Output: ");
    puts(str);
}
```

if input:comp133



output is:  
String Output:comp133



# String

```
char name[] = "Harry Potter";  
printf("%c", *name);           // Output: H  
printf("%c", *(name+1));       // Output: a  
printf("%c", *(name+7));       // Output: o
```

```
char *namePtr;  
namePtr = name;
```

```
printf("%c", *namePtr);        // Output: H  
printf("%c", *(namePtr+1));    // Output: a  
printf("%c", *(namePtr+7));    // Output: o
```

يمكننا هكذا انشاء بويتر  
يؤشر على اول عنصر في  
السترينج.....

# String

```
*name تعادل name[0]  
*(name+1) تعادل name[1]  
*(name+7) تعادل name[7]
```

```
char name[] = "Harry Potter";
```

```
printf("%c", *name); // Output: H  
printf("%c", *(name+1)); // Output: a  
printf("%c", *(name+7)); // Output: o
```

```
char *namePtr;  
namePtr = name;
```

```
printf("%c", *namePtr); //Output: H  
printf("%c", *(namePtr+1)); //Output: a  
printf("%c", *(namePtr+7)); //Output: o
```

البويتر يؤشر على مكان  
تخزين اول عنصر بال  
array وللتنقل بين عناصر  
array يمكننا استخدام  
الشكل التالي:

```
*namePtr تعادل nameptr[0]  
*(namePtr+1) تعادل nameptr[1]  
*(namePtr+7) تعادل nameptr[7]
```

# String

طريقة اخرى لكتابة الكود

```
char name[] = "Harry Potter";
```

```
printf("%c", name[0]);    // Output: H
printf("%c", name[1]);    // Output: a
printf("%c", name[7]);    // Output: o
```

```
char *namePtr;
```

```
namePtr = &name[0];
printf("%c", namePtr[0]);    // Output: H
printf("%c", namePtr[1]);    // Output: a
printf("%c", namePtr[7]);    // Output: o
```

لو لاحظنا فان

```
namePtr=&name[0]
```

تعاادل

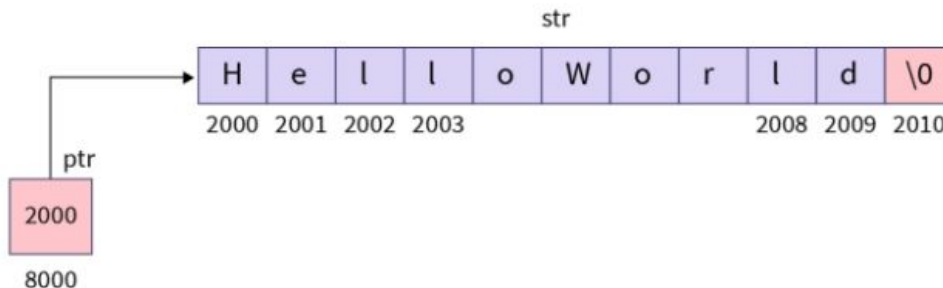
```
namePtr = name
```

# String

```
#include<stdio.h>
int main() {
    char str[11] = "HelloWorld";
    char *ptr = str;
    while (*ptr != '\0') {
        printf("%c", *ptr);
        ptr++;
    }
    return 0;
}
```

\0 : تمثل نهاية كل سترينج

output is: HelloWorld



كما موضح بالشكل البوینتر یؤشر علی  
مکان تخزين اول عنصر بال array  
وللتقل بین عناصر array  
نستخدم ptr++

# Two dimensional character arrays

الصيغة العامة

**data-type** array-name **[m][n];**

**m:** عدد strings

and

**n:** حجم كل string

# Initializing 2 D Strings

- We can initialize a 2 D string as follows:

```
char word[4][5]={“Cat”, “Boat”, “Mat”, ”Rate”};
```

ترتيب السطرينج داخل  
array

word[0]

word[1]

word[2]

word[3]

word[3][0]=R

word[0][1]=a

coluns

0	1	2	3	4	
C	a	t	'\0'		0
B	o	a	t	'\0'	1
M	a	t	'\0'		2
R	a	t	e	'\0'	3

rows

# Accessing 2 D Strings: Example 1

```
char word[4][5]={"Cat", "Boat", "Mat", "Rate"};
```

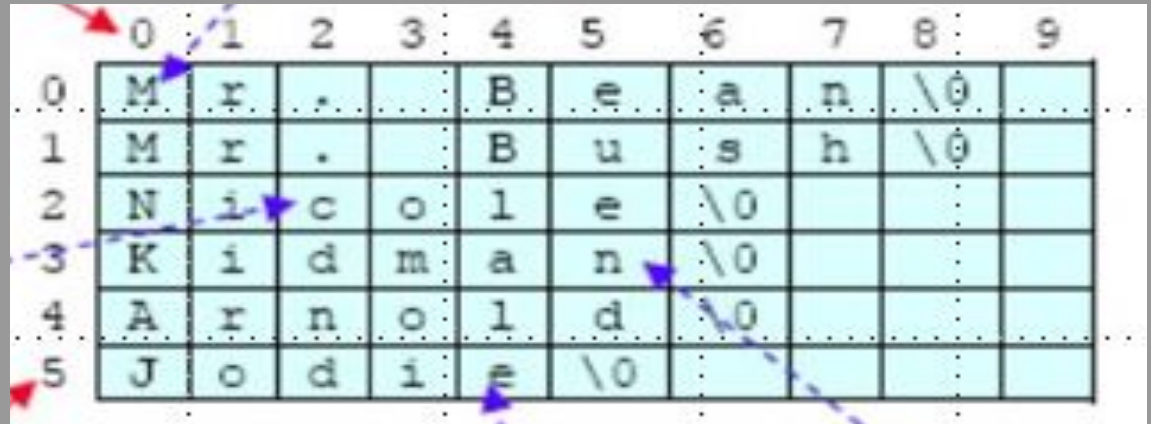
Statement	output
<code>printf("%c",word[2][0])</code>	M
<code>printf("%c",word[1][1]);</code>	o
<code>printf("%c",word[2]);</code>	Mat
<code>printf("%c",word[0]);</code>	Cat

	0	1	2	3	4
0	C	a	t	'\0'	
1	B	o	a	t	'\0'
2	M	a	t	'\0'	
3	R	a	t	e	'\0'



# String

```
char Name[6][10] = {"Mr. Bean", "Mr.Bush", "Nicole", "Kidman", "Arnold", "Jodie"};
```



A diagram illustrating the memory layout of the `Name` array. It is a 6x10 grid of characters. The columns are indexed 0 to 9, and the rows are indexed 0 to 5. The data is as follows:

	0	1	2	3	4	5	6	7	8	9
0	M	r	.		B	e	a	n	\0	
1	M	r	.		B	u	s	h	\0	
2	N	i	c	o	l	e	\0			
3	K	i	d	m	a	n	\0			
4	A	r	n	o	l	d	\0			
5	J	o	d	i	e	\0				

Red arrows point to the first character of each row (index 0). Blue dashed arrows point to the null terminator (\0) in each row (index 6).

Statement	output
<code>printf("%c", word[0]);</code>	Mr. Bean
<code>printf("%c", word[2][0]);</code>	e
<code>printf("%c", word[4]);</code>	Arnold
<code>printf("%c", word[5][2]);</code>	d
<code>printf("%c", word[1][8]);</code>	

# Input and displaying array of strings

- لإدراج قيم ل array سترينج نحن بحاجة الى loop كالتالي

```
char names[4][20];  
for(int i=0;i<4;i++)  
    gets(names[i]);
```

- لطباعه قيم ل array سترينج نحن بحاجة الى loop كالتالي

```
char names[4][20];  
for(int i=0;i<4;i++)  
    puts(names[i]);
```

# Program to input and display the names of n cities.

```
char city[20][25];  
  
int n;  
  
printf("\nHow many names do you wish to  
input: ");  
  
scanf("%d", &n);  
  
for(int i = 0; i < n; i++) {  
    printf("\nCity %d: ", i + 1);  
    gets(city[i]);  
    printf("\nDisplaying names of cities:\n");  
    for (int i = 0; i < n; i++) {  
        puts(city[i]);  
    }  
}
```

## output:

How many names do you wish  
to input: 3

City 1:Ramallah

City 2:amman

City 3:birzeit

Displaying names of cities:

Ramallah

amman

birzeit

# String

لا يمكن استخدام هذه الطريقة

لوضع قيمه سترينج داخل اخر

```
str1=str2
```



```
strcpy(str1, str2)
```

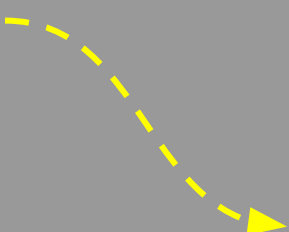
هذا الفنكشن يقوم بوضع

قيمه str2 داخل str1

# String

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

```
int main() {
    char str1[20] = "C programming";
    char str2[20];
    strcpy(str2, str1);
    puts(str2);
    return 0;
}
```



output:

C programming

# String

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

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

```
int main()
```

```
{
```

```
    char a[20]="Program";
```

```
    char b[20]={'P','r','o','g','r','a','m','\0'}
```

```
    printf("Length of string a = %zu \n",strlen(a));
```

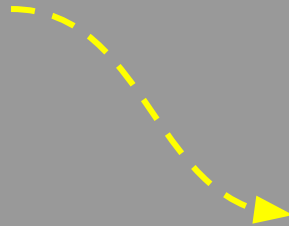
```
    printf("Length of string b = %zu \n",strlen(b));
```

```
    return 0;
```

```
}
```

strlen(str1)

هذا فنكشن يقوم بحساب  
حجم السترينج



output:

Length of string a = 7

Length of string b = 7

# String

```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100] = "This is ",
    str2[] = "programiz.com";
```

```
    strcat(str1, str2);
```

```
    puts(str1);
```

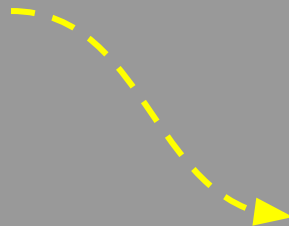
```
    return 0;
```

```
}
```

```
strcat(str1, str2);
```

هذا فنكشن يقوم باضافة

str1 الى str2



output:

This is programiz.com

# String

```
strcmp(str1, str2);
```

يقوم بمقارنة اول حرف من  
السترينج الاول مع المقابل له في  
السترينج الثاني بناا على ASCII  
اذا اول حرفين متشابهات يتقل  
ليقارن العنصر الثاني مع المقابل له

```
strcmp(str1, str2);
```

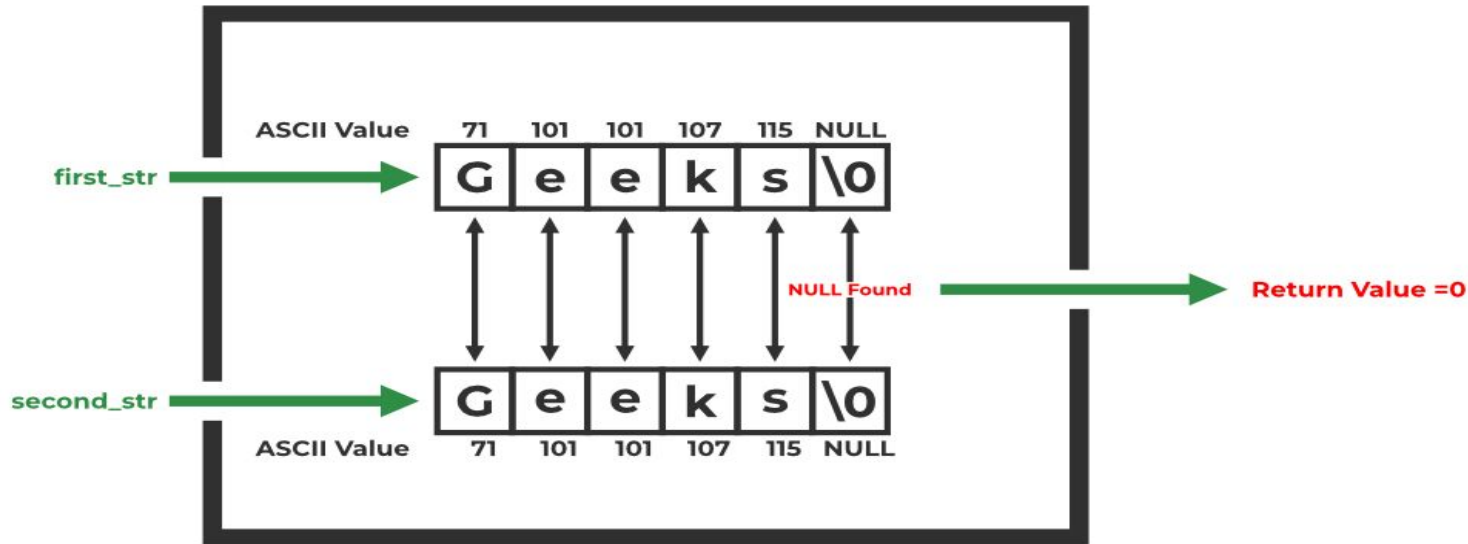
هذا فنكشن يقوم بمقارنه  
جملتين ويستخدم للترتيب غالبا

strcmp(str1, str2)	result
if str1 > str2	>0 like 1
if str1 < str2	<0 like -1
if str1 = str2	0



# String

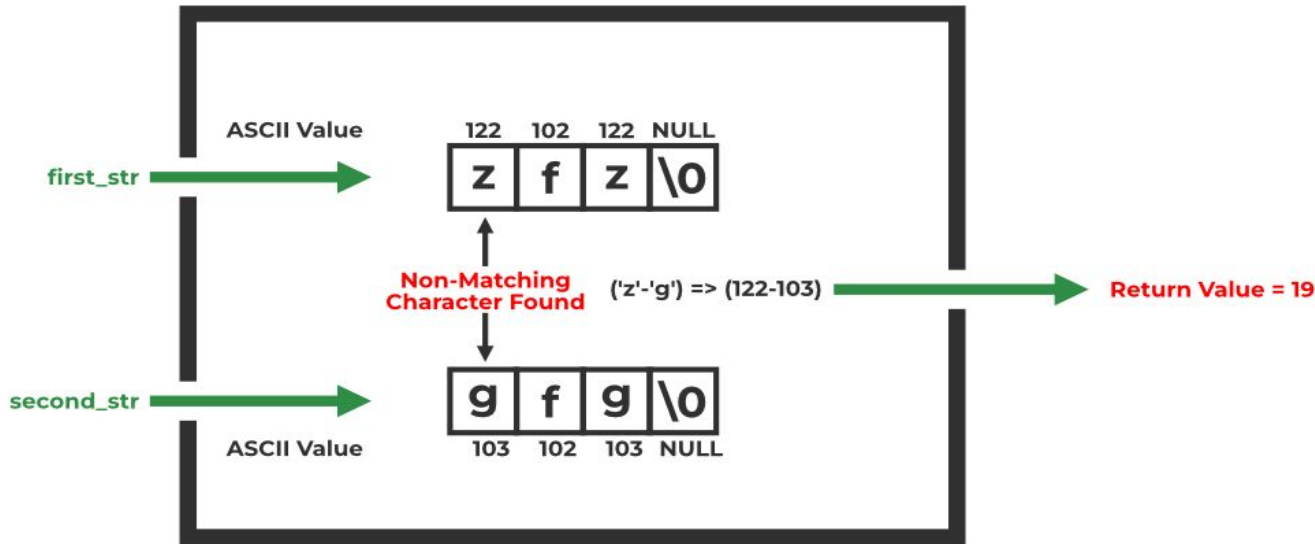
`strcmp("Geeks", "Geeks");`



<code>strcmp(str1, str2)</code>	result
if <code>str1 &gt; str2</code>	>0 like 1
if <code>str1 &lt; str2</code>	<0 like -1
if <code>str1 = str2</code>	0

# String

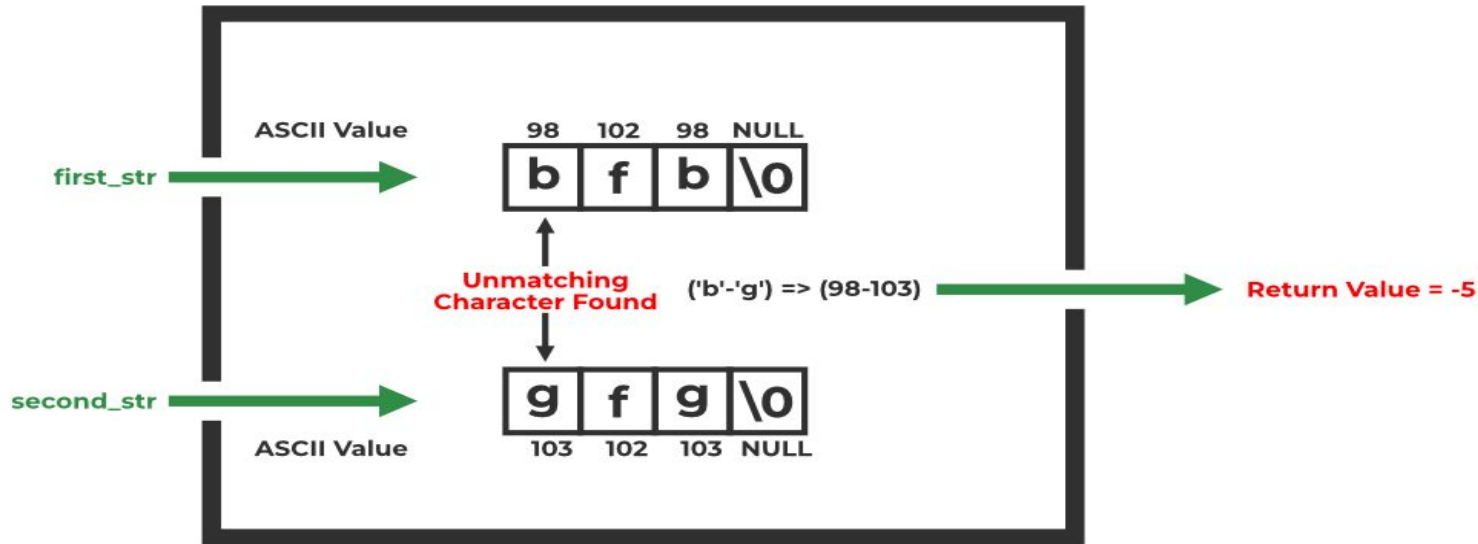
`strcmp("zfg", "gfg");`



<code>strcmp(str1, str2)</code>	result
<code>if str1 &gt; str2</code>	<code>&gt;0</code> like 1
<code>if str1 &lt; str2</code>	<code>&lt;0</code> like -1
<code>if str1 = str2</code>	0

# String

`strcmp("bfb", "gfg");`



<code>strcmp(str1, str2)</code>	result
if <code>str1 &gt; str2</code>	<code>&gt;0</code> like 1
if <code>str1 &lt; str2</code>	<code>&lt;0</code> like -1
if <code>str1 = str2</code>	0

# String

```
char str[] = "comp133-by-moayad";
```

قام بحجز العتاصر بالذاكرة كما تلاحظ

main																			
str	array																		
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
		char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char
		'c'	'o'	'm'	'p'	'1'	'3'	'3'	'-'	'b'	'y'	'-'	'm'	'o'	'a'	'y'	'a'	'd'	'\0'
token	pointer to char																		
	?																		

# String

```
char str[] = "comp133-by-moayad";  
char *token;  
    token = strtok(str, " - ");  
printf(" % s\n", token); //OUTPUT:COMP133
```

كما نلاحظ قمنا بانشاء بويتر  
يؤشر على اول عنصر بالسترنج

وباستخدام strtok قمنا  
بتجزئه السترنج عند - الاولى  
وفيت الثانيه - ووضع مكانها  
NULL او \0

main

array

str

token

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char
'c'	'o'	'm'	'p'	'1'	'3'	'3'	'\0'	'b'	'y'	'-'	'm'	'o'	'a'	'y'	'a'	'd'	'\0'

pointer to char

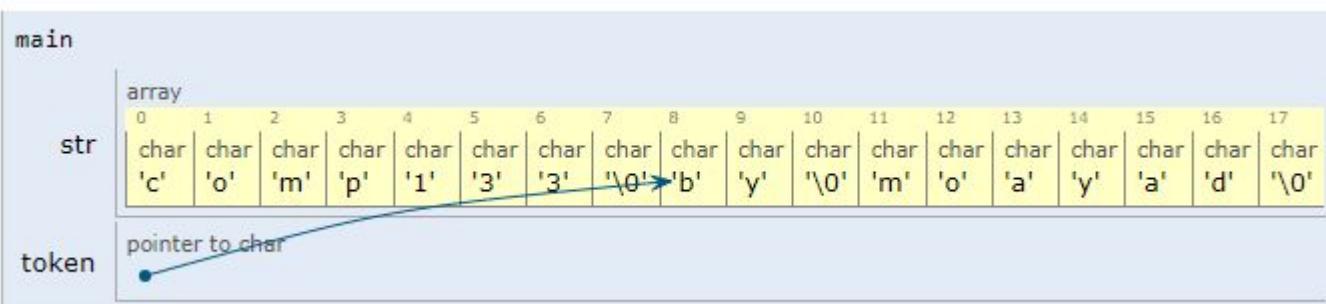


# String

```
char str[] = "comp133-by-moayad";  
char* token = strtok(str, " - ");  
printf(" % s\n", token); //OUTPUT:COMP133
```

```
token = strtok(NULL, " - ");  
printf(" % s\n", token); //OUTPUT:BY  
return 0;
```

كما نلاحظ هنا اصبح  
البوینتر یؤشر بعد null  
وسيقوم باستبدال  
بالnull



# String

```
char str[] = "comp133-by-moayad";  
char* token = strtok(str, " - ");  
printf(" % s\n", token);
```

```
token = strtok(NULL, " - ");
```

```
printf(" % s\n", token);  
token = strtok(NULL, " - ");  
printf(" % s\n", token);
```

OUTPUT:  
comp133  
by  
moayad

main

array

str

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char
'c'	'o'	'm'	'p'	'1'	'3'	'3'	'\0'	'b'	'y'	'\0'	'm'	'o'	'a'	'y'	'a'	'd'	'\0'

token

pointer to char



# String

```
char str[] = "comp133-by-moayad";  
char* token = strtok(str, " - ");  
printf(" % s\n", token);
```

```
token = strtok(NULL, " - ");
```

```
printf(" % s\n", token);  
token = strtok(NULL, " - ");  
printf(" % s\n", token);
```

```
token = strtok(NULL, " - ");  
printf(" % s\n", token);
```

output:  
comp133  
by  
moayad  
(NULL)

بعد طباعه moayad  
وقمنا بالتقطيع مره  
اخرى لا يوجد قيم  
بالسترينج لتقطيعها اذن  
اصبحت قيمة البوینتر  
null

main

array

str

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char	char
'c'	'o'	'm'	'p'	'1'	'3'	'3'	'\0'	'b'	'y'	'\0'	'm'	'o'	'a'	'y'	'a'	'd'	'\0'

token

pointer to char  
NULL (0x0)



# String

## Find the Frequency of Characters

```
#include <stdio.h>
int main() {
    char str[1000], ch;
    int count = 0;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Enter a character: ");
    scanf("%c", &ch);
    for (int i = 0; str[i] != '\0'; ++i)
    {
        if (ch == str[i])
            ++count;
    }
    printf("Frequency of %c = %d", ch, count);
    return 0;
}
```

output:

Enter a string:moayad

Enter a character:a

Frequency of a = 2

# String

## Revers string

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

```
void reverseString(char str[]) {
    int length = strlen(str);
    for (int i = 0; i < length / 2; i++) {
        char temp = str[i];
        str[i] = str[length - i - 1];
        str[length - i - 1] = temp;
    }
}
```

```
int main() {
    char str[100];
    printf("Enter a string: ");
    gets(str);
    reverseString(str);
    printf("Reversed string: %s\n", str);
```

```
    return 0;
}
```

output:

Enter a string: moayad

Reversed string: dayaom

# String

## Revers string

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

void reverseString(char str[]) {
    int length = strlen(str);
    for (int i = 0; i < length / 2; i++) {
        char temp = str[i];
        str[i] = str[length - i - 1];
        str[length - i - 1] = temp;
    }
}

int main() {
    char str[100];
    printf("Enter a string: ");
    gets(str);
    reverseString(str);
    printf("Reversed string: %s\n", str);

    return 0;
}
```

output:  
Enter a string: moayad  
Reversed string: dayaom