STRINGs .....

❖ For use STRING functions must include the string headers# include <string.h>

❖ Declaration and allocation of string

charvariable_name [size]

size is the number of characters in the string

❖ String Functions

1. Concatenate strings :

General form: **strcat(destination, source)**

Description :

Appends a copy of the *source* string to the *destination* string. The new string formed by the concatenation of both in *destination*.

Example :*str1* = "Information "

str2= "Technology"

strcat (***str1***, *str2*) →*str1*="InformationTechnology"

2. Append characters from string :

General form: **strncat(destination, source, num)**

Description:

Appends the first *num* characters of *source* to *destination*.

Example :*str1* = "Information"

str2 = "Technology"

strncat (***str1***, *str2* , 3) →*str1*= "InformationTec"

3.Compare two strings :

General form: **strcmp(str1, str2)**

Description:

Compares the string *str1* to the string *str2*. This function starts comparing the first character of each string. If they are equal to each other, it continues with the following pairs until the characters differ. This function returns an integral value(<0 or >0 =0) indicating the relationship between the strings

0→*ifstr1* ≡ *str2*, **positive**→*ifstr1* > *str2* , **negative**→*ifstr1* < *str2*

Example1 :*str1* = "I like Iraq"

str2 = "I like My country Iraq"

x = strcmp(*str1*, *str2*) →*x* = -4not match

Example2 :*str1* = "I like my country Iraq "

str2 = "I like Iraq"

x = strcmp(*str1*, *str2*) →*x* =36not match

*4.*Compare characters of two strings :

General form : strncmp(str1, str2, num)

Description:

Compares up to *num* characters of the string *str1* to those of the string *str2*.

This function starts comparing the first character of each string. If they are equal to each other, it continues with the following pairs until the characters differor until *num* characters match in both strings, whichever happens first. This function returns an integral value (>0 or <0 or =0) indicating the relationship between the strings.

0→*ifstr1* ≡ *str2,* **positive**→*ifstr1 > str2 ,* **negative**→*ifstr1 < str2*

Example1 :*str1* = "computer "

str2 = " compare "

x = strncmp(*str1*, *str2, 3*) →*x* = 0 match

Example2 :*str1* = "I like Iraq "

str2 = "I likeIraq"

x = strcmp(*str1*, *str2, 8*) →*x* = 41not match

5. Locate first occurrence of character in string :

General form: strchr(str, chr)

Description:

Returns a pointer to the first occurrence of character *chr* in the string *str*.

The terminating null-character is considered part of the string therefore, it can also be located in order to retrieve a pointer to the end of a string.

The return value is the **first occurrence** of character *chr* in the string *str*. If the *chr* is not found, the function returns a null pointer.

Example :*char * x*

str1 = " *computer* "
ch = "*c* "
ch1 = *"s"*
x = strchr(*str1*, *ch*) →*x* = not NULLmatch
x = strchr(*str1*, *ch1*) →*x* = NULLnot match

6. Locate last occurrence of character in string :

General form : strrchr(str, chr)

Description:

Returns a pointer to the last occurrence of character *chr* in the string *str*.

The terminating null-character is considered part of the string.

Therefore, it can also be located to retrieve a pointer to the end of a string.

Example :*char * x*

str1 = " *I Love Iraq too much*"
ch = "*I* "
ch1 = *"O"*
x = strrchr(*str1*, *ch*) →*x* = not NULL match
x = strrchr(*str1*, *ch1*) →*x* = NULL not match

7. Split string into tokens (strtok) :

General form: strtok(str, delimiters)

Description:

This function split *str* into tokens, which are sequences of contiguous characters separated by any of the characters that are part of *delimiters*.

On a first call, the function expects a string as argument for *str*, whose first character is used as the starting location to scan for tokens.

In subsequent calls, the function expects a NULL pointer and uses the position right after the end of the last token as the new starting location for scanning.

To determine the beginning and the end of a token, the function first scans from the starting location for the first character not contained in *delimiters* (which becomes the *beginning of the token*). And then scans starting from this *beginning of the token* for the first character contained in *delimiters*, which becomes the *end of the token*. The scan also stops if the terminating *null character* is found.

The point where the last token was found is kept internally by the function to be used on the next call.

Example-1 :*char * x*

str1 = " *I, Love . Iraq :too : much*"
ch = ": "
ch1 = *"a"*
x = strtok(*str1*, *ch*) →*x* = "*I , Love . Iraq*"
x = strtok(*str1*, *ch1*) →*x* = "*I , Love . Ir*"

Example-2 :*char * x*

str1 = " *I , Love . Iraq : too : much*"
ch = ": "
ch1 = *"a"*
x = strtok(*str1*, *ch*) →*x* = "*I , Love . Iraq*"
x = strtok(*NULL*, *ch1*) →*x* = "*too*"

8. Get string length :

General form: strlen(str)

Description:

Returns the length or number of character for string str.

Example :

str1 = "Structure Programming"
x = strlen(*str1*) →*x* = 21

9. copy string :

General form : strcpy (destination, source)

Description:

Copies the string *source* into the string *destination*, including the terminating null character.(the result in the destination string).

Example:

str1 = " *Good*"
str2 = "*Nice* "
strcpy (*str1*, *str2*) →*str1* = "*Nice* " , *str2* = "*Nice* "

10. Copy characters from string :

General form: strncpy(destination, source, num)

Description :

Copies the first *num* characters of *source* to *destination*.
If the end of the *source* string (which is signaled by a null-character) is found before *num* characters have been copied, *destination*is padded with zeros until a total of *num* characters have been written it. (the result in the destination string).

Example :

str1 = " *Ahmed*"
str2 = " *Haydir* "

strcpy (*str1*, *str2, 2*) → *str1* = “*Hamed*” , *str2* = “*Haydir*”