Strings

Abdallah Karakra

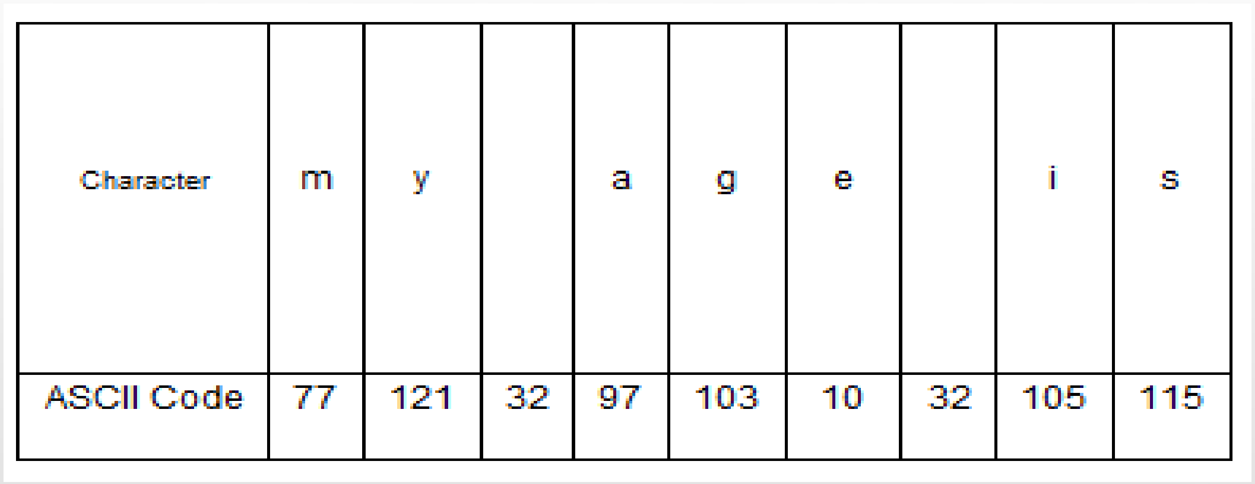
Computer Science Department

Comp 230

Sunday, January 10, 2016

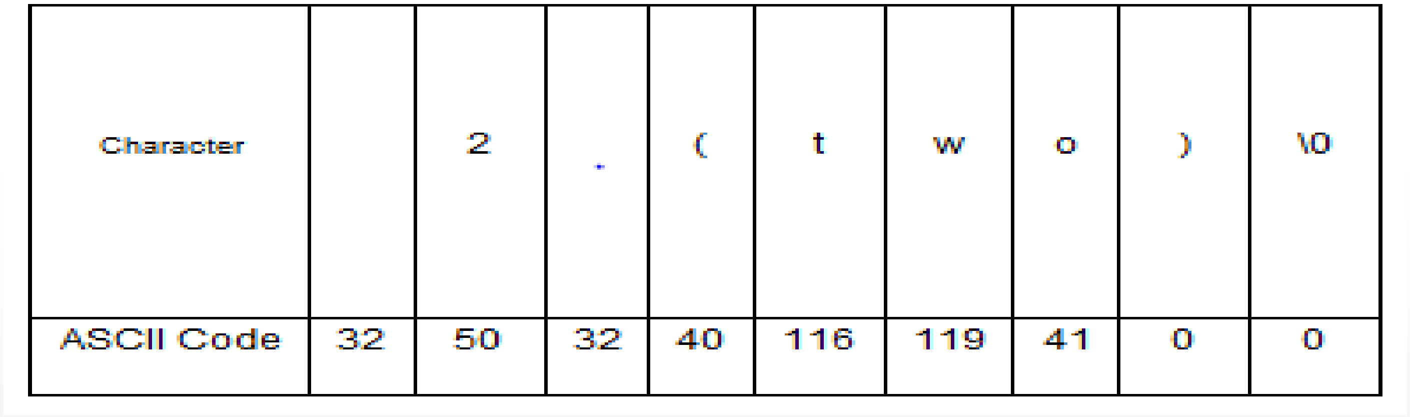
Strings

- A string is a sequence of characters (Array of characters).
- Strings are stored in memory as ASCII codes



Character	m	y		a	g	e		i	s
ASCII Code	77	121	32	97	103	10	32	105	115

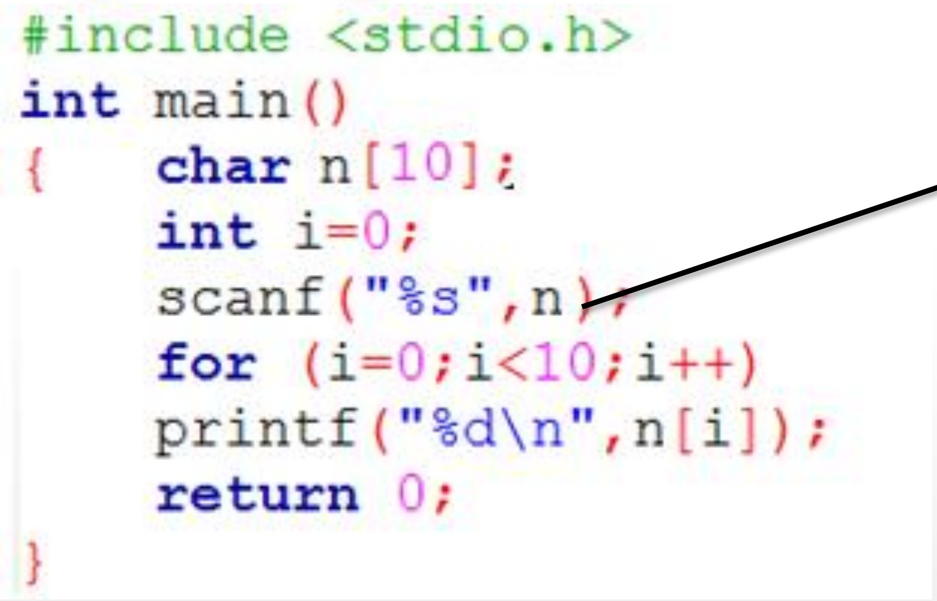
Strings



Character		2	.	(	t	w	o	)	\0
ASCII Code	32	50	32	40	116	119	41	0	0

- **The last character is the null character having ASCII value zero (character '\0' that marks the end of a string in C)**

Strings: Examples

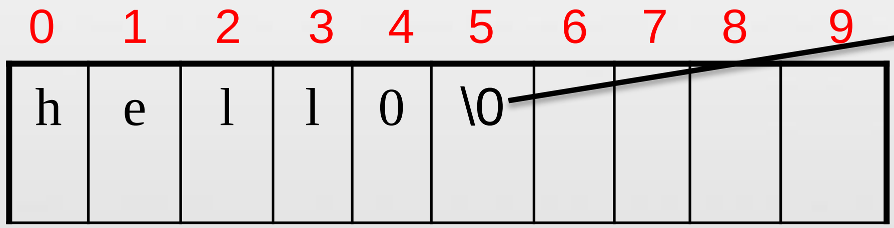


```
#include <stdio.h>
int main()
{     char n[10];
      int i=0;
      scanf("%s",n);
      for (i=0;i<10;i++)
      printf("%d\n",n[i]);
      return 0;
}
```

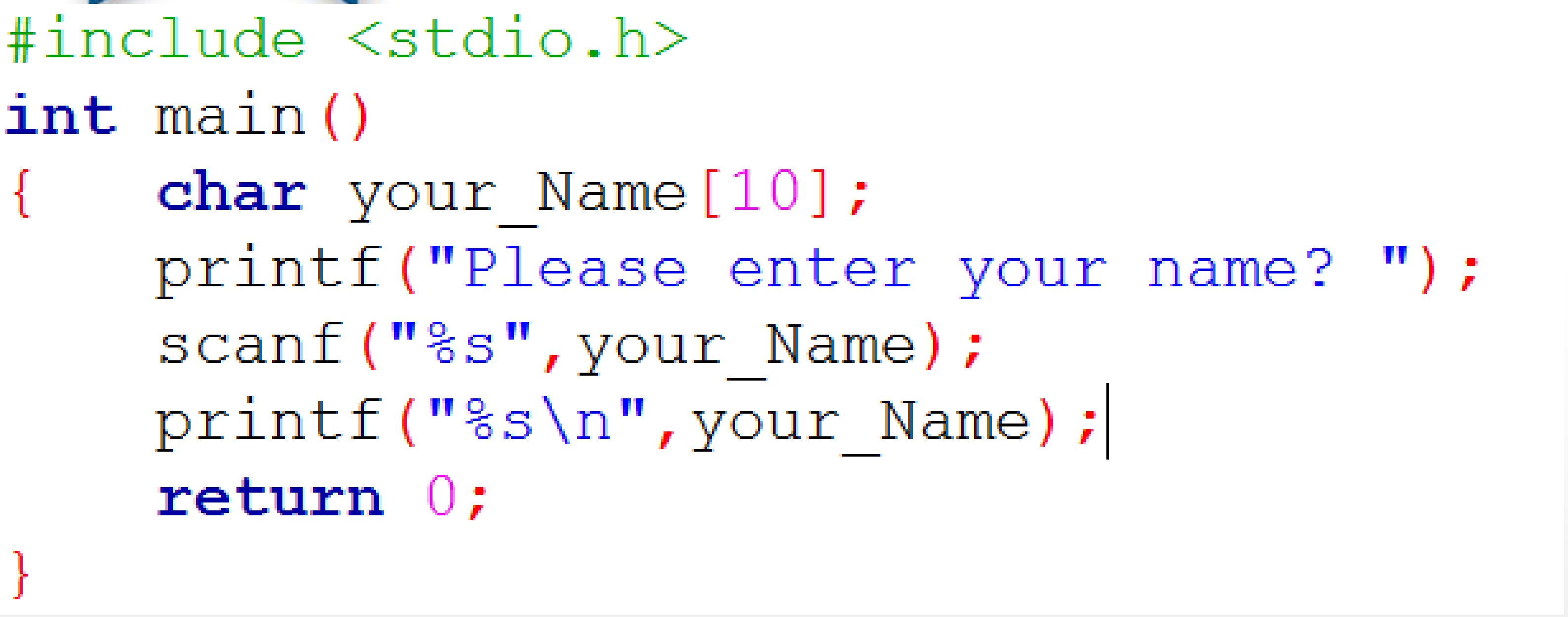
hello

Output:

```
104
101
108
108
111
0
2
0
0
0
```



Strings: Examples



```c
#include <stdio.h>
int main()
{    char your_Name[10];
     printf("Please enter your name? ");
     scanf("%s",your_Name);
     printf("%s\n",your_Name);
     return 0;
}
```

0	1	2	3	4	5	6	7	8	9
A	h	m	a	d	\0				

\0: null character, determines the end of the string

Strings: Examples

char your_Name[10]="Ahmad";

your_Name

0	1	2	3	4	5	6	7	8	9
A	h	m	a	d	\0	?	?	?	?

char your_Name[ ]="Ahmad";

your_Name

0	1	2	3	4	5
A	h	m	a	d	\0

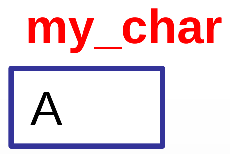
char your_Name[10]={'a','h','m','a','d'};
your_Name[5]='\0';

your_Name

0	1	2	3	4	5	6	7	8	9
A	h	m	a	d	\0	?	?	?	?

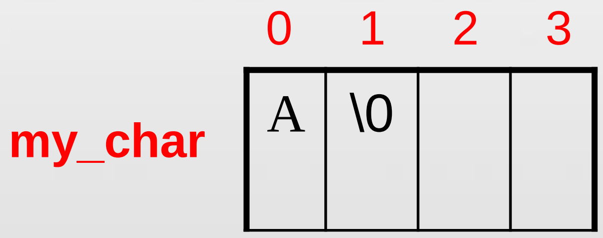
Strings: Common Errors

char my_char='A'; // correct



char my_char="A"; // error

char my_char [4]="A"; // correct



Strings: Array of strings

```
char week_days[7][13]={"Monday","Tuesday","Wednesday",...}
```

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	M	o	n	d	a	y	\0	?	?	?	?	?	?
1	T	u	e	s	d	a	y	\0	?	?	?	?	?
2	W	e	d	n	e	s	d	a	y	\0	?	?	?
3	.												
4	.												
5	.												
6													

Strings: Example

Write a program to read the names of 5 students and also their grades (three grades for each students),and save them.

Names[5][10]

Y	a	m	e	n	\0				
A	h	m	A	d	\0				
K	h	a	l	e	d	\0			
M	o	h	a	m	m	a	d	\0	
S	a	n	d	y	\0				

Grades[5][3]

99	98	100
80	90	50
70	78	60
88	90	70
70	90	92

Code

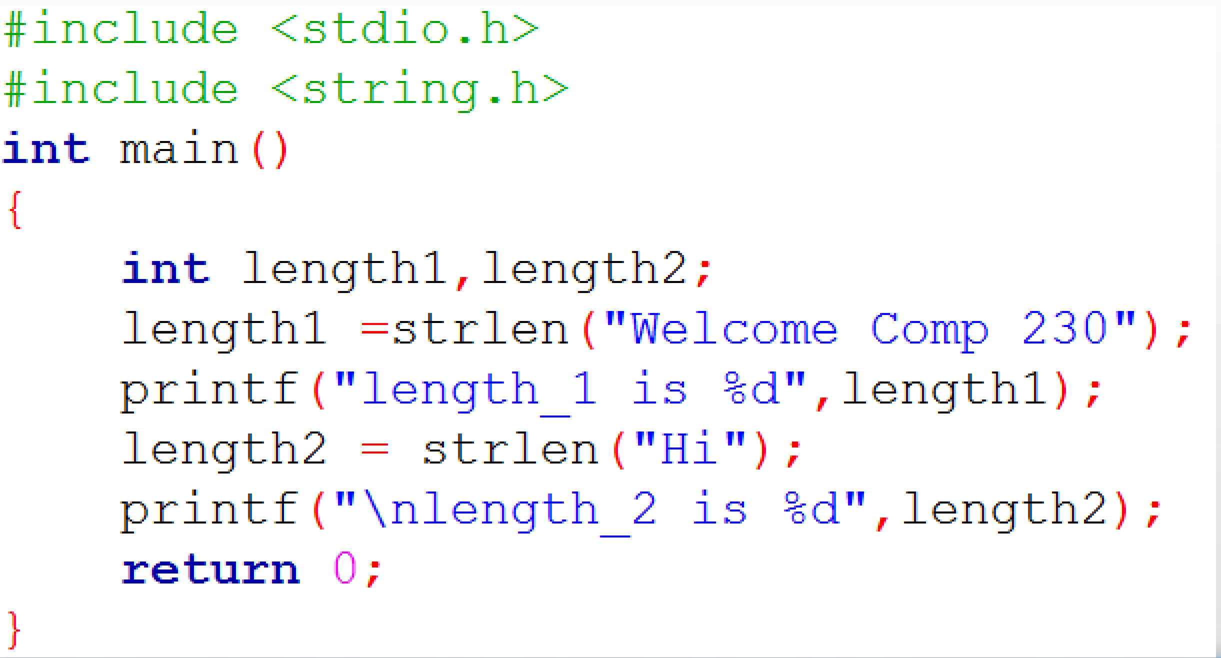
Strings Functions

include string.h library header file in the program

- **Length (number of characters in the string).**

  strlen() function

  Syntax n=strlen(string);



```
#include <stdio.h>
#include <string.h>
int main()
{
    int length1,length2;
    length1 =strlen("Welcome Comp 230");
    printf("length_1 is %d",length1);
    length2 = strlen("Hi");
    printf("\nlength_2 is %d",length2);
    return 0;
}
```

strlen()
Returns the number of characters in s , not counting the terminating null.
size_t strlen(const char *str)

```
length_1 is 16
length_2 is 2
```

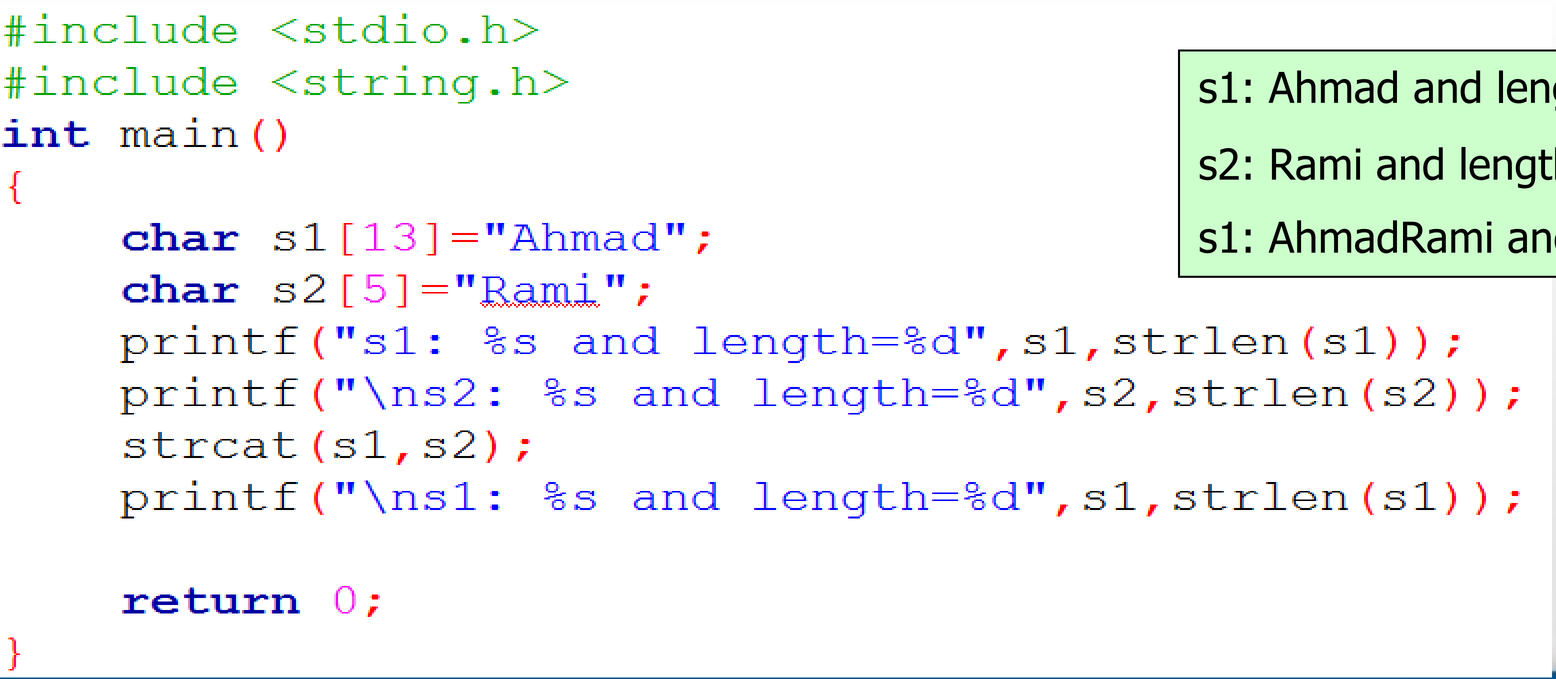
Strings Functions

include string.h library header file in the program

- **Joins 2 strings together**

strcat() function

Syntax strcat(string1,string2) ;



```c
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[5]="Rami";
    printf("s1: %s and length=%d",s1,strlen(s1));
    printf("\ns2: %s and length=%d",s2,strlen(s2));
    strcat(s1,s2);
    printf("\ns1: %s and length=%d",s1,strlen(s1));

    return 0;
}
```

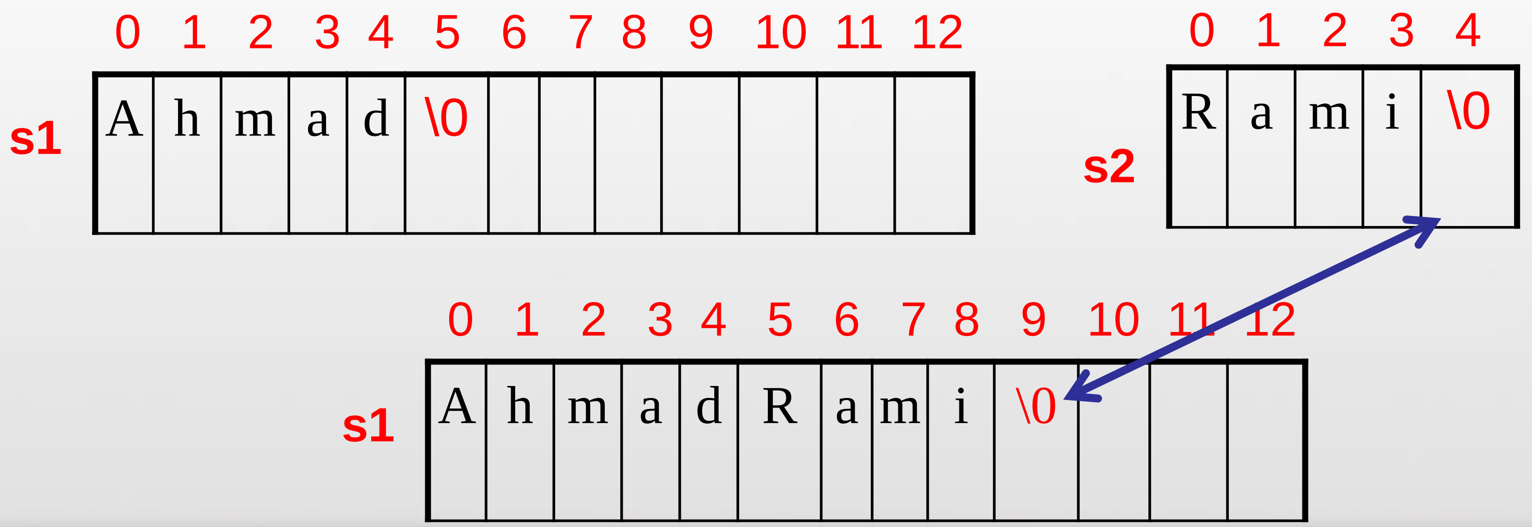
```
s1: Ahmad and length=5
s2: Rami and length=4
s1: AhmadRami and length=9
```

Strings Functions

include string.h library header file in the program

- **Joins 2 strings together (Appends source to the end of dest)**

char s1[13]="Ahmad";
char s2[5]="Rami";
strcat(s1,s2);



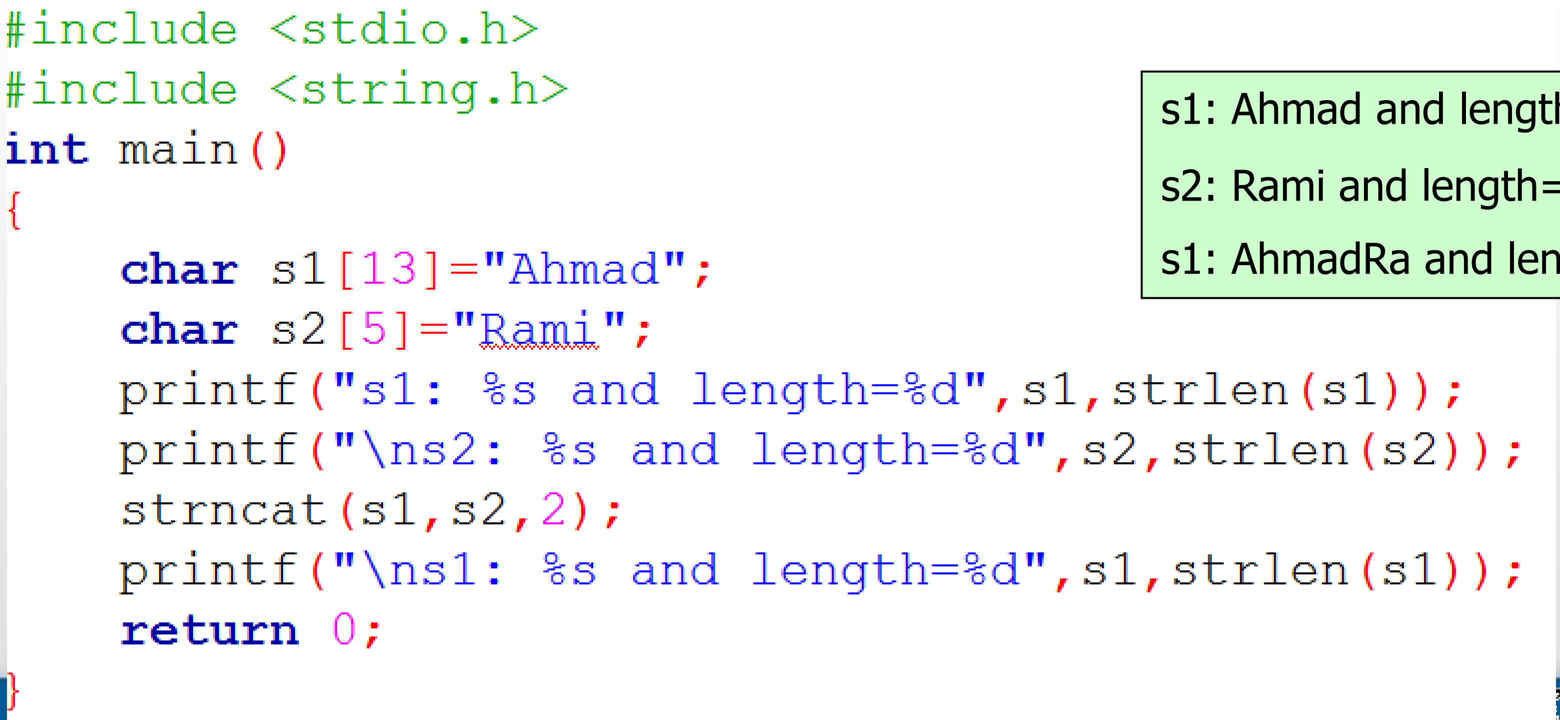
Strings Functions

include string.h library header file in the program

- **Joins 2 strings together (add a n characters from s2 to s1 plus a null character)**

strncat() function

Syntax strncat(string1,string2,n) ;



```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[5]="Rami";
    printf("s1: %s and length=%d",s1,strlen(s1));
    printf("\ns2: %s and length=%d",s2,strlen(s2));
    strncat(s1,s2,2);
    printf("\ns1: %s and length=%d",s1,strlen(s1));
    return 0;
}
```

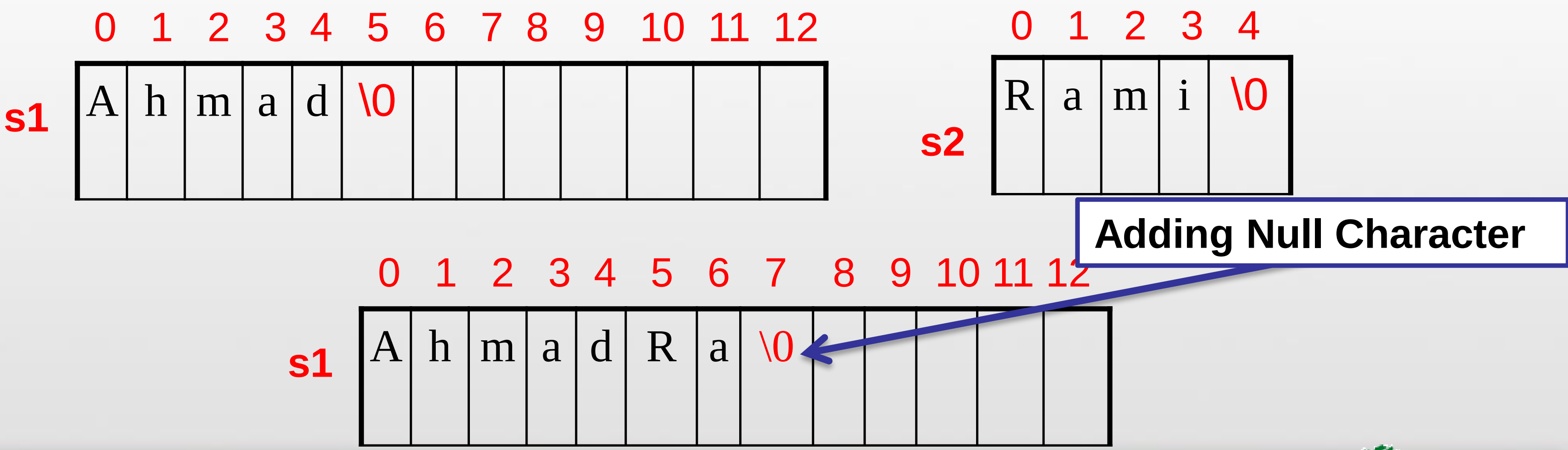
```
s1: Ahmad and length=5
s2: Rami and length=4
s1: AhmadRa and length=7
```

Strings Functions

include string.h library header file in the program

- **Joins 2 strings together (add a n characters from s2 to s1 plus a null character)**

```
char s1[13]="Ahmad";
char s2[5]="Rami";
strncat(s1,s2,2);
```



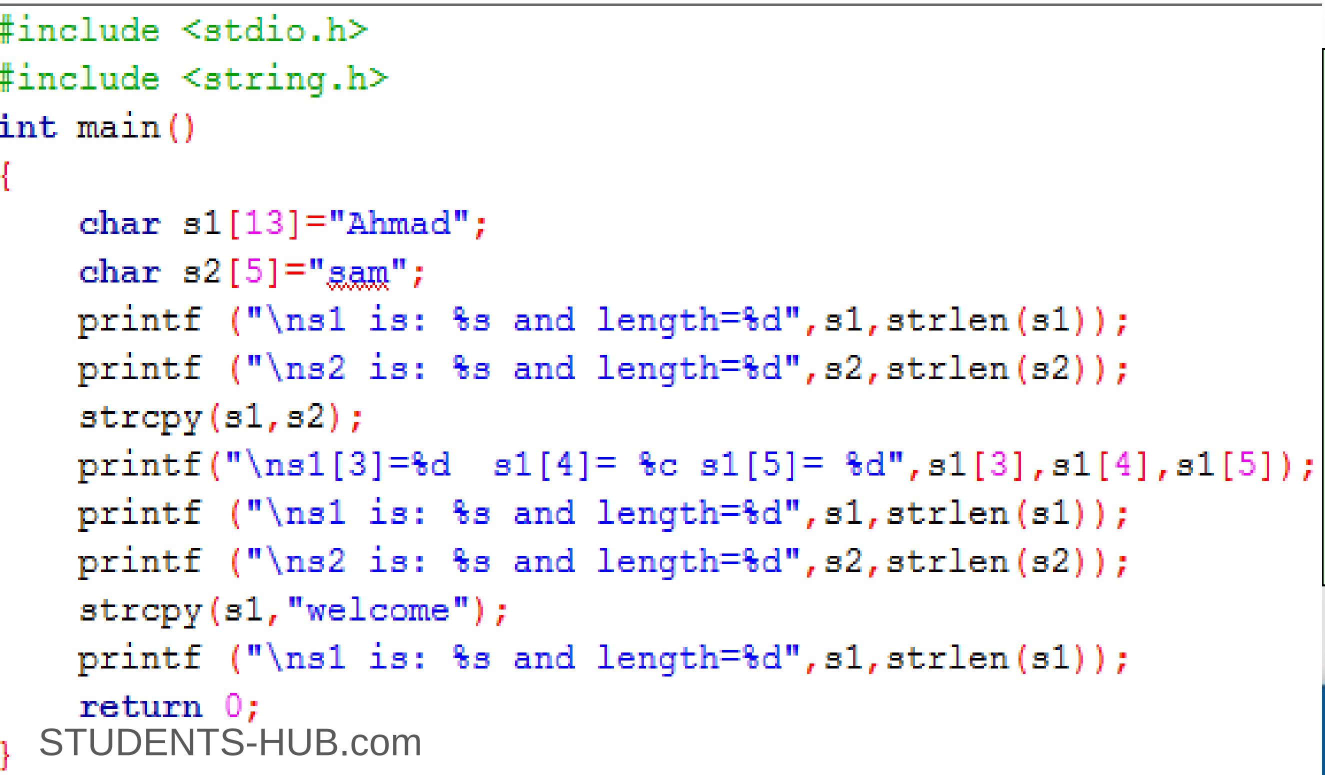
Strings Functions

include string.h library header file in the program

- **Assigns the contents of string2 to string1**

strcpy () function

Syntax strcpy(string1,string2);



```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[5]="sam";
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    printf ("\ns2 is: %s and length=%d",s2,strlen(s2));
    strcpy(s1,s2);
    printf("\ns1[3]=%d  s1[4]= %c s1[5]= %d",s1[3],s1[4],s1[5]);
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    printf ("\ns2 is: %s and length=%d",s2,strlen(s2));
    strcpy(s1,"welcome");
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    return 0;
}
```

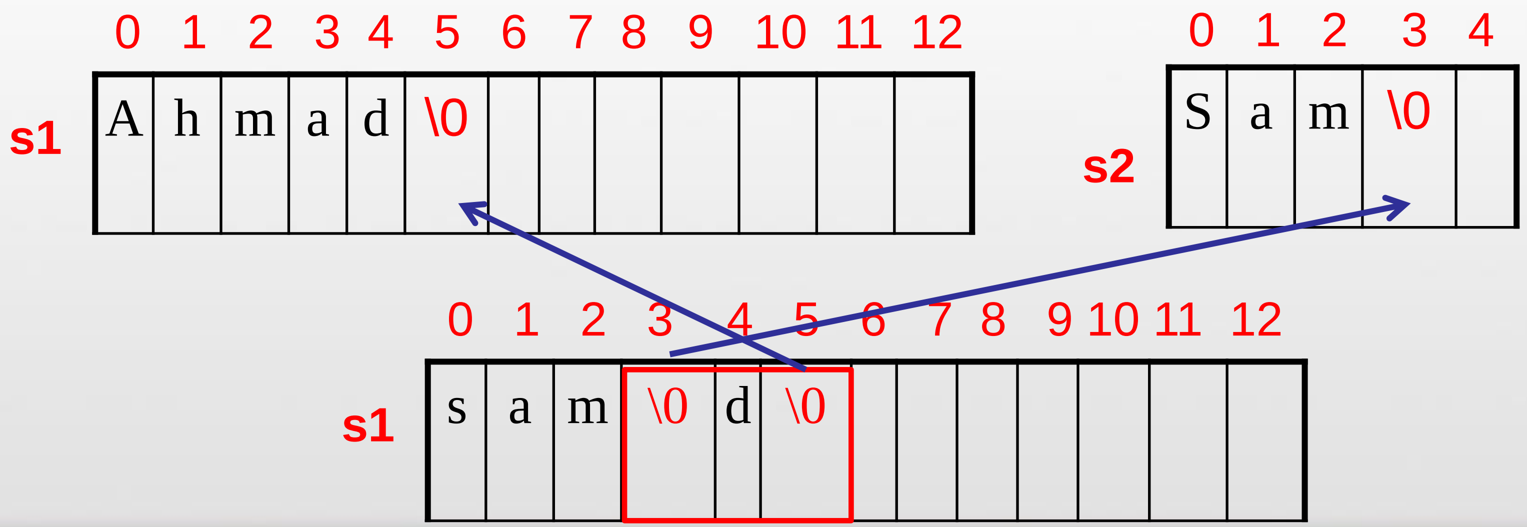
```
s1 is: Ahmad and length=5
s2 is: sam and length=3
s1[3]=0  s1[4]= d s1[5]= 0
s1 is: sam and length=3
s2 is: sam and length=3
s1 is: welcome and length=7
```

Strings Functions

include string.h library header file in the program

- **Assigns the contents of string2 to string1**

```
char s1[13]="Ahmad";
char s2[5]="sam";
strcpy(s1,s2);
```



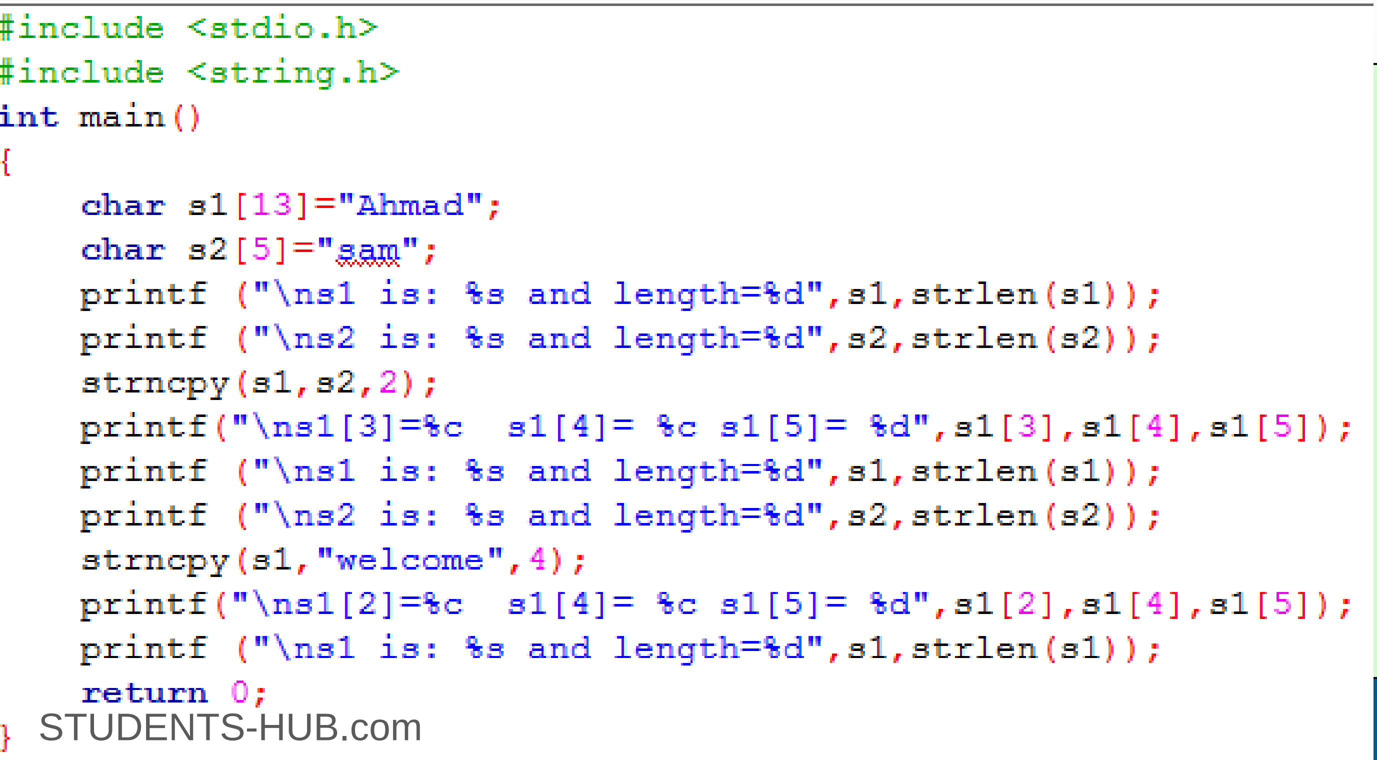
Strings Functions

include string.h library header file in the program

- **Makes a copy of up to n characters from string2 in string1 (does NOT add a null character)**

strncpy() function

Syntax strncpy(string1,string2,n) ;



```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[5]="sam";
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    printf ("\ns2 is: %s and length=%d",s2,strlen(s2));
    strncpy(s1,s2,2);
    printf("\ns1[3]=%c  s1[4]= %c s1[5]= %d",s1[3],s1[4],s1[5]);
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    printf ("\ns2 is: %s and length=%d",s2,strlen(s2));
    strncpy(s1,"welcome",4);
    printf("\ns1[2]=%c  s1[4]= %c s1[5]= %d",s1[2],s1[4],s1[5]);
    printf ("\ns1 is: %s and length=%d",s1,strlen(s1));
    return 0;
}
```

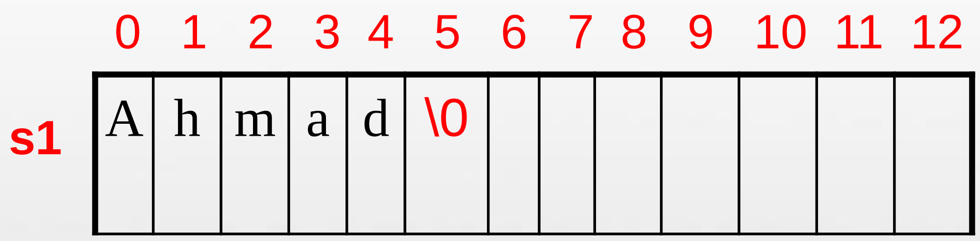
```
s1 is: Ahmad and length=5
s2 is: sam and length=3
s1[3]=a  s1[4]= d s1[5]= 0
s1 is: samad and length=5
s2 is: sam and length=3
s1[2]=l  s1[4]= d s1[5]= 0
s1 is: welcd and length=5
```

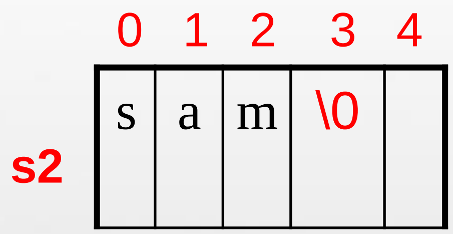
Strings Functions

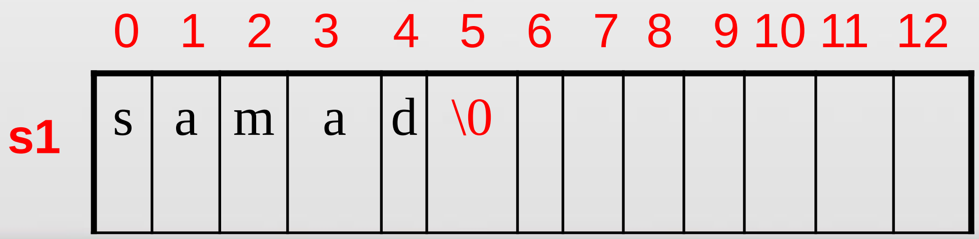
include string.h library header file in the program

- **Makes a copy of up to n characters from string2 in string1** **(does NOT add a null character)**

```
char s1[13]="Ahmad";
char s2[5]=“sam";
strncpy(s1,s2,2);
```







Strings Functions

include string.h library header file in the program

- **which returns a zero if 2 strings are equal, or a non zero number if the strings are not the same.**

strcmp() function

Syntax strcmp(string1,string2) ;

int result= strcmp (string1,string2);

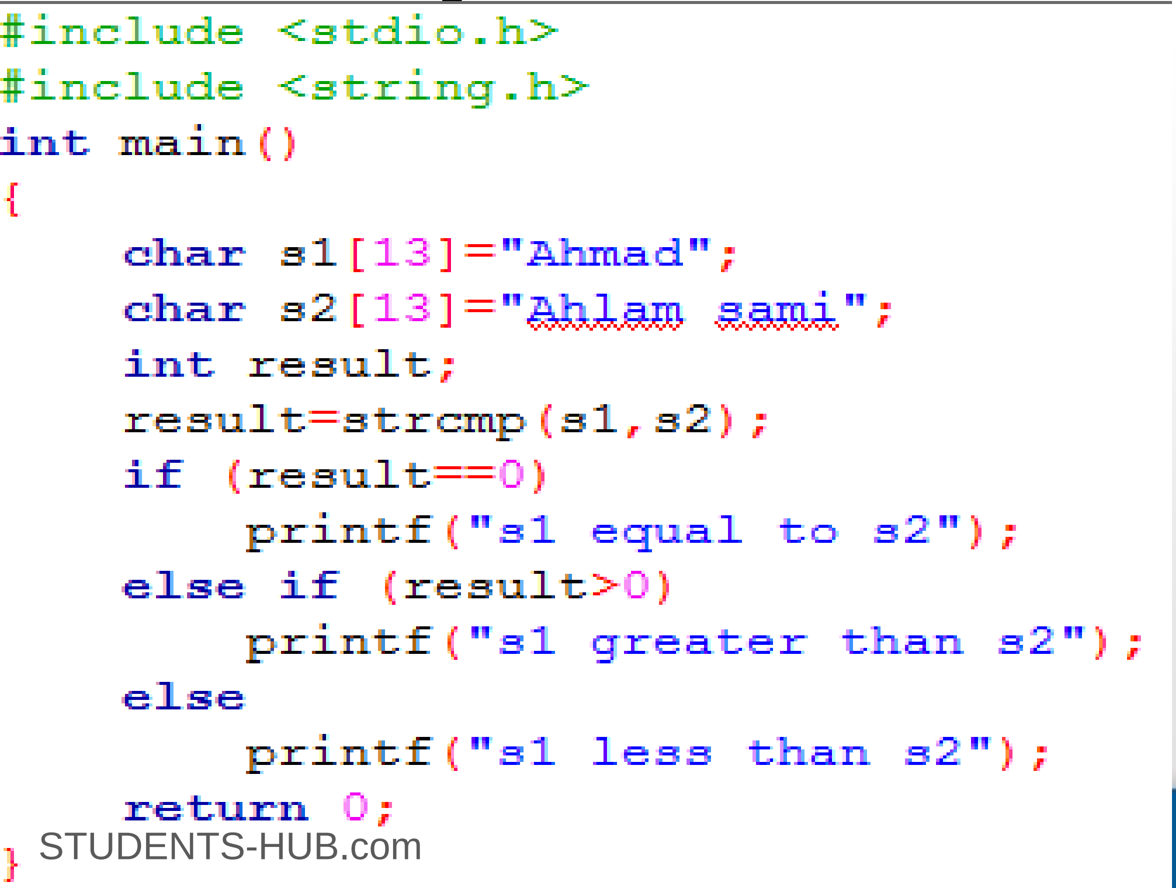
result=0,   if string1 equal string2
result>0 , if string1 greater than string2
Result<0 , if string1 less than string2

Strcmp uses ASCII values to compare between two strings.

Strings Functions

include string.h library header file in the program

- **which returns a zero if 2 strings are equal, or a non zero number if the strings are not the same.**



```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[13]="Ahlam sami";
    int result;
    result=strcmp(s1,s2);
    if (result==0)
        printf("s1 equal to s2");
    else if (result>0)
        printf("s1 greater than s2");
    else
        printf("s1 less than s2");
    return 0;
}
```

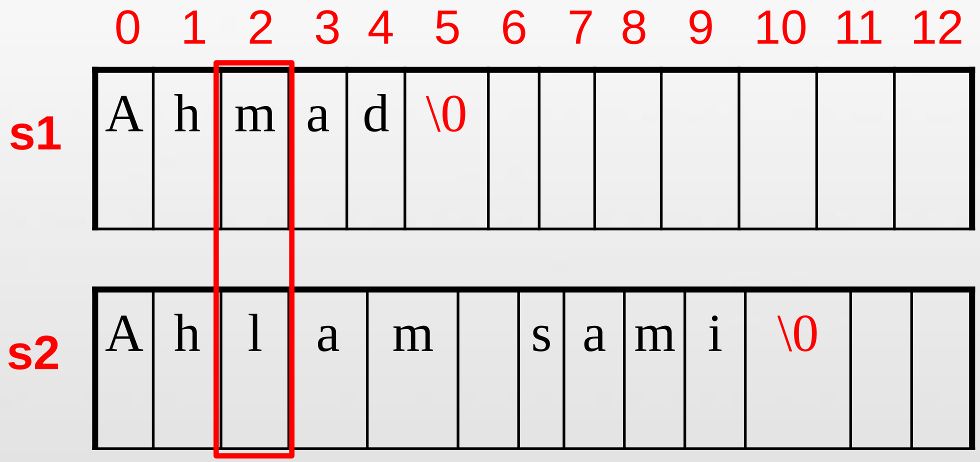
```
s1 greater than s2
```

Strings Functions

include string.h library header file in the program

- **which returns a zero if 2 strings are equal, or a non zero number if the strings are not the same.**

```
char s1[13]="Ahmad";
char s2[13]="Ahlam sami";
strcmp(s1,s2);
```



A equal A

h equal h

m greater than l (109 greater than 108)

→ s1 greater than s2

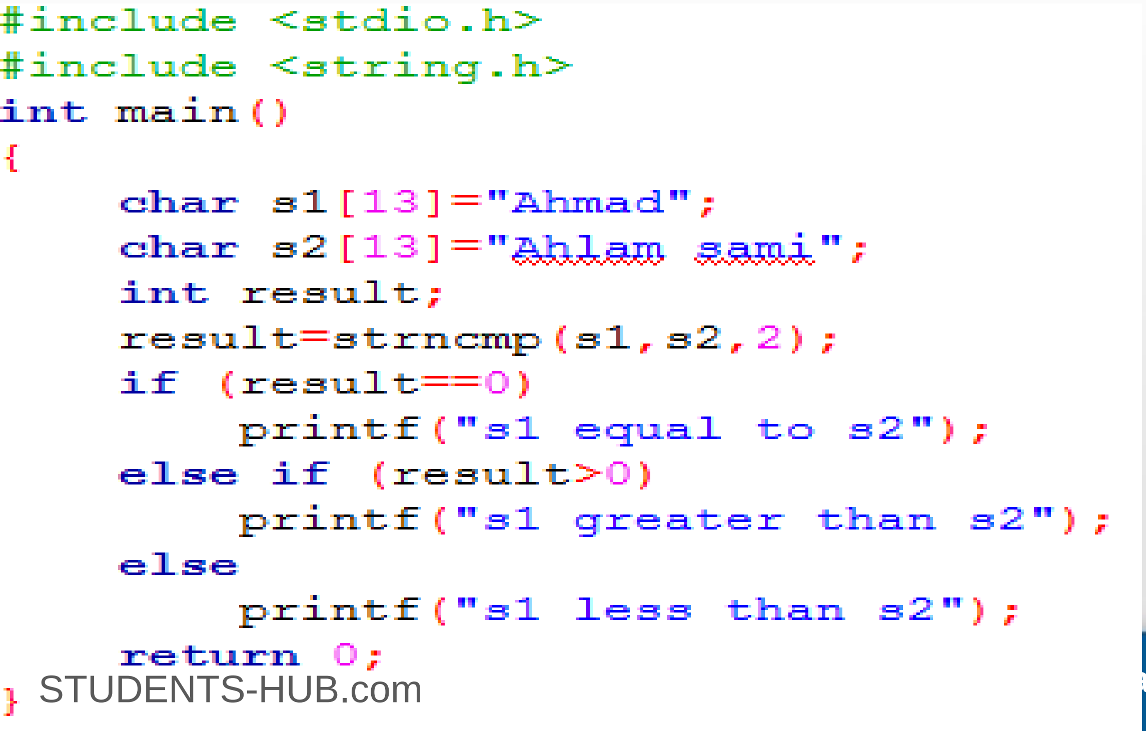
Strings Functions

include string.h library header file in the program

- **Compares the first n characters of s1 and s2**

strcmp() function

Syntax strcmp(string1,string2,n) ;



```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[13]="Ahmad";
    char s2[13]="Ahlam sami";
    int result;
    result=strncmp(s1,s2,2);
    if (result==0)
        printf("s1 equal to s2");
    else if (result>0)
        printf("s1 greater than s2");
    else
        printf("s1 less than s2");
    return 0;
}
```

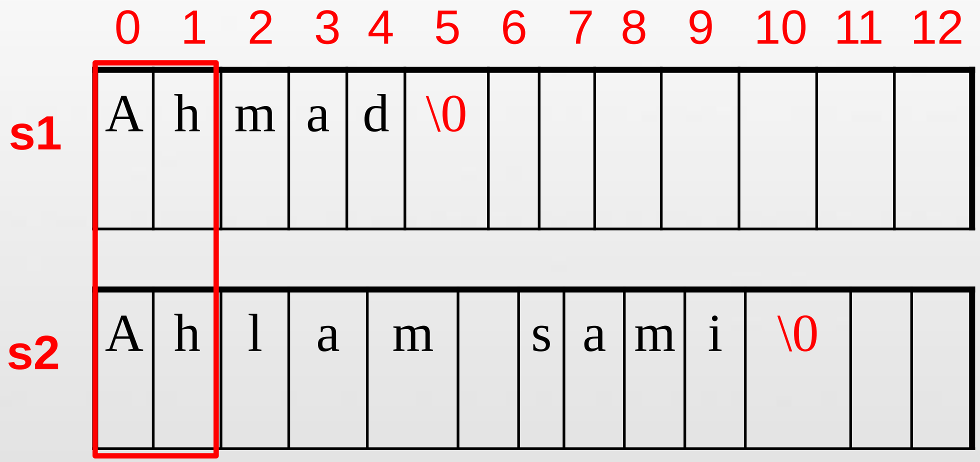
```
s1 equal to s2
```

Strings Functions

include string.h library header file in the program

- **which returns a zero if 2 strings are equal, or a non zero number if the strings are not the same.**

```
char s1[13]="Ahmad";
char s2[13]="Ahlam sami";
srncmp(s1,s2,2);
```



A equal A

h equal h

→ s1 equal s2

Summary

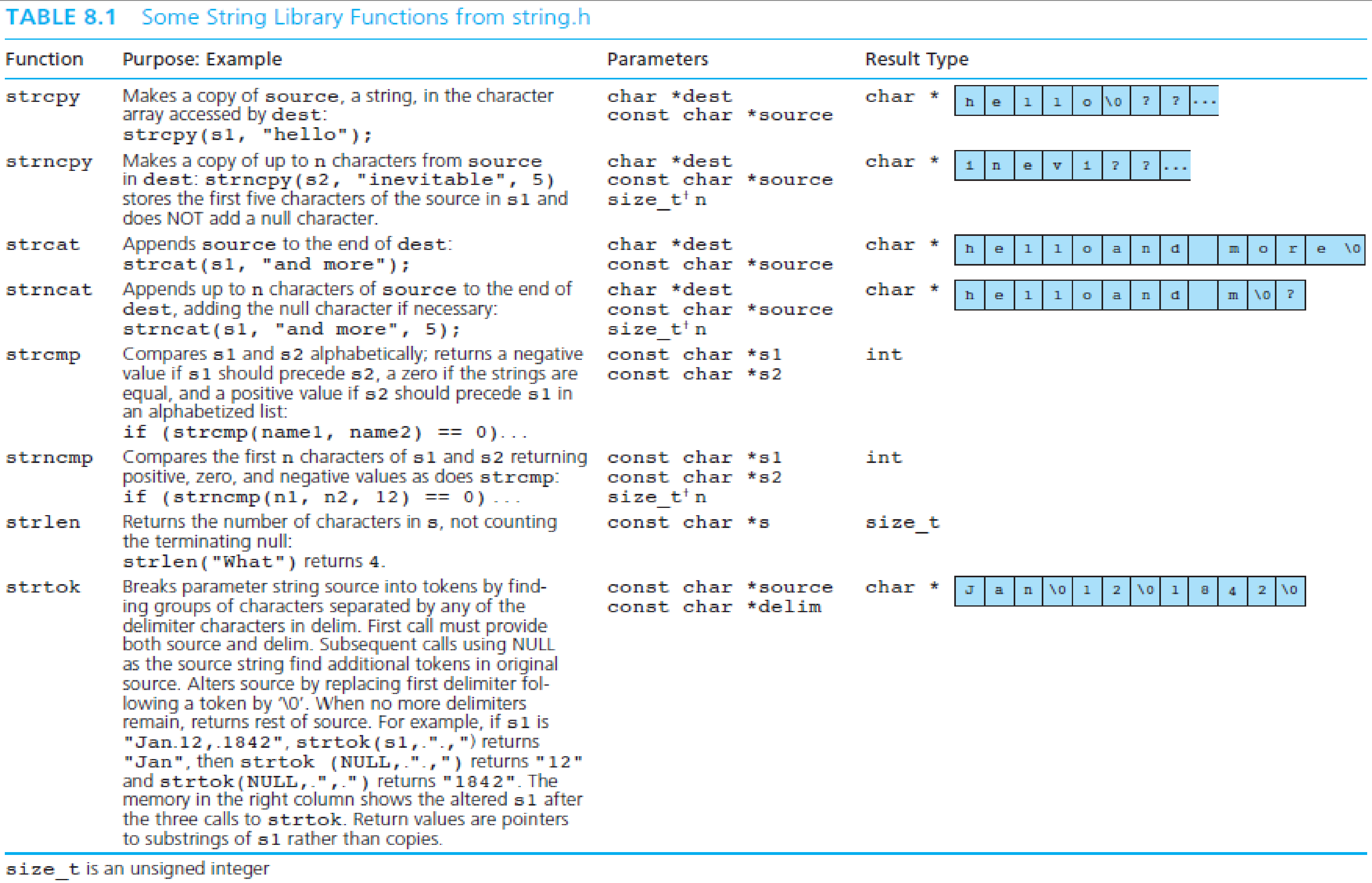


TABLE 8.1 Some String Library Functions from string.h

Function	Purpose: Example	Parameters	Result Type

`strcpy`	Makes a copy of `source`, a string, in the character array accessed by `dest`: `strcpy(s1, "hello");`	`char *dest`
		`const char *source`
			`char *` `h e l l o \0 ? ? ...`

`strncpy`	Makes a copy of up to `n` characters from `source` in `dest`: `strncpy(s2, "inevitable", 5)` stores the first five characters of the source in `s1` and does NOT add a null character.	`char *dest`
		`const char *source`
		`size_t† n`
			`char *` `i n e v i ? ? ...`

`strcat`	Appends `source` to the end of `dest`: `strcat(s1, "and more");`	`char *dest`
		`const char *source`
			`char *` `h e l l o a n d   m o r e \0`

`strncat`	Appends up to `n` characters of `source` to the end of `dest`, adding the null character if necessary: `strncat(s1, "and more", 5);`	`char *dest`
		`const char *source`
		`size_t† n`
			`char *` `h e l l o a n d   m \0 ?`

`strcmp`	Compares `s1` and `s2` alphabetically; returns a negative value if `s1` should precede `s2`, a zero if the strings are equal, and a positive value if `s2` should precede `s1` in an alphabetized list: `if (strcmp(name1, name2) == 0)...`	`const char *s1`
		`const char *s2`
			`int`

`strncmp`	Compares the first `n` characters of `s1` and `s2` returning positive, zero, and negative values as does `strcmp`: `if (strncmp(n1, n2, 12) == 0) ...`	`const char *s1`
		`const char *s2`
		`size_t† n`
			`int`

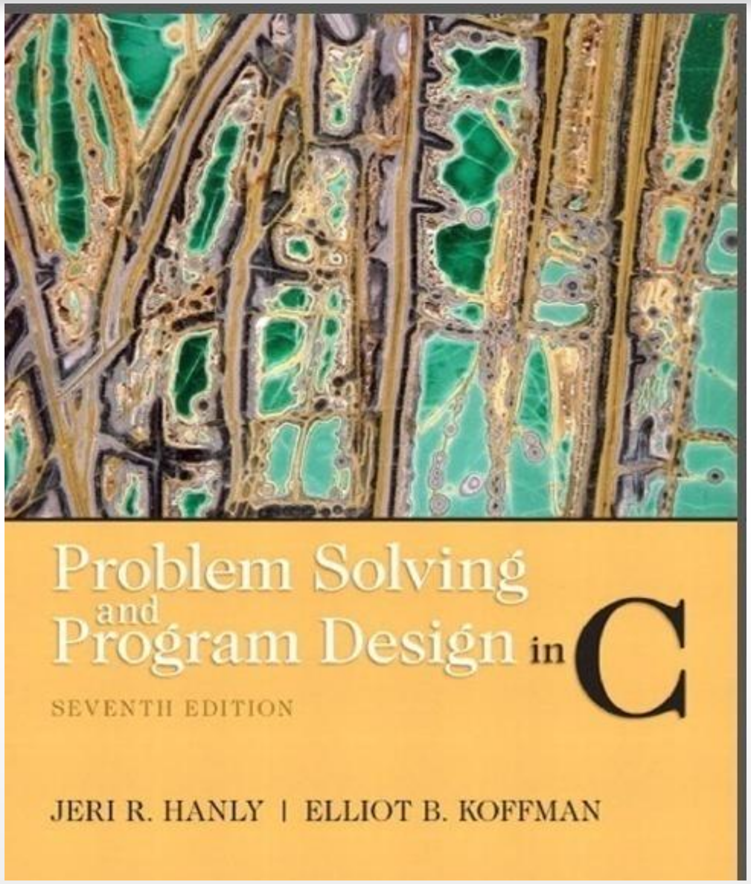
`strlen`	Returns the number of characters in `s`, not counting the terminating null: `strlen("What")` returns `4`.	`const char *s`	`size_t`

`strtok`	Breaks parameter string source into tokens by finding groups of characters separated by any of the delimiter characters in delim. First call must provide both source and delim. Subsequent calls using NULL as the source string find additional tokens in original source. Alters source by replacing first delimiter following a token by '\0'. When no more delimiters remain, returns rest of source. For example, if `s1` is `"Jan.12,.1842"`, `strtok(s1,".,")` returns `"Jan"`, then `strtok (NULL,".,")` returns `"12"` and `strtok(NULL,",.")` returns `"1842"`. The memory in the right column shows the altered `s1` after the three calls to `strtok`. Return values are pointers to substrings of `s1` rather than copies.	`const char *source`
		`const char *delim`
			`char *` `J a n \0 1 2 \0 1 8 4 2 \0`

† `size_t` is an unsigned integer

Question?

GOOD LUCK


Problem Solving and Program Design in C

SEVENTH EDITION

JERI R. HANLY | ELLIOT B. KOFFMAN

References:

Problem Solving & Program Design in C (main reference)