STRINGS

Tuesday, 2. August 2022 00:08

- Strings is a seauence of characters, terminated at null character. null character = '\0'
- Strings are handled in char array.
  char str [10];

INITIALIZING CHAR ARRAY DURING DECLARATION

```
int main()
{
  char str [10] = { 'P', 'A', 'T', 'N', 'A' }
  int i ;
  for( i=0 ; i<9 ; i++)
     printf(" %c" , str [i]);
  return 0;
}
```

0	1	2	3	4	5	6	7	8	9 10
P	A	T	N	A	\0	\0	\0	\0	\0 \0

⇓

80	65	84	78	65	0	0	0	0	0

NOTE:- we are entering an alphabet as character but computer or system is taking it as ASCII code.

PRINTING STRING

* we will not use printf in order to print, because it will check the condition more times, the character has to print.

```
int main ()
{
  char str [10] = { 'P', 'A', 'T', 'N', 'A' };
  int i;
  for ( i=0 , str [i], i++ )
      printf(" %c " , str [i] );
  return 0;
}
```

⇓
How this executes?
First i=0 ∴ str [0] = P and as we have already know that str [0] = P takes its ascii code so 80 and every non-zero value is true or 1. So loop continues.

i	str [i]	ASCII	loop
0	P	80	✓
1	A	65	✓
2	T	84	✓
3	N	78	✓
4	A	65	✓
5	\0	0	✗ False

So, loop will terminate here.

USE OF NULL CHARACTER

```
int main ()
{
  char str [10] = { 'P', 'A', 'T', 'N', 'A' }
  return 0;
}
```

P	A	T	N	A	\0	\0	\0	\0	\0

* wherever you see null character. string ends.

```
int main ()
{
  char str [10] = { 'A', 'S', 'H', 'I', 'S', 'H' };
  printf ( "%s", str );
}
```

∴ Here str means address of string indirectly & str [i] , str are same.

CALCULATING LENGTH OF STRING

```
int main ()
{
  char str [10] = " PATNA" ;
  int i ;
  for ( i=0 ; str[i]; i++ );
printf (" length of string is %d" , i );
```

* writing each character with single quote is very time consuming so we will write full string with double quote.

STRING CONSTANT

"PATNA" → string constant or string literal

TAKING INPUT FROM USER

- scanf is not capable to input scanf() multiword string. because space, tab, new line characters are delimiters.
- we will not use scanf for string input.
- gets ( ) is a capable to input multi word string. but gets() takes illegal memory.
  like if you make an array of 10 elements and want to take input from user by gets() to store in your arrays, gets() will never warn if user writes more than 10 elements so it will acquire illegal memory.

fgets ( )

we use fgets ( ) function.

fgets (arrayname , size of array , stdin)
↳ for keyboard

∴ fgets (str , 5 , stdin)

MEMORY CONCEPT

1 byte = 8 bit

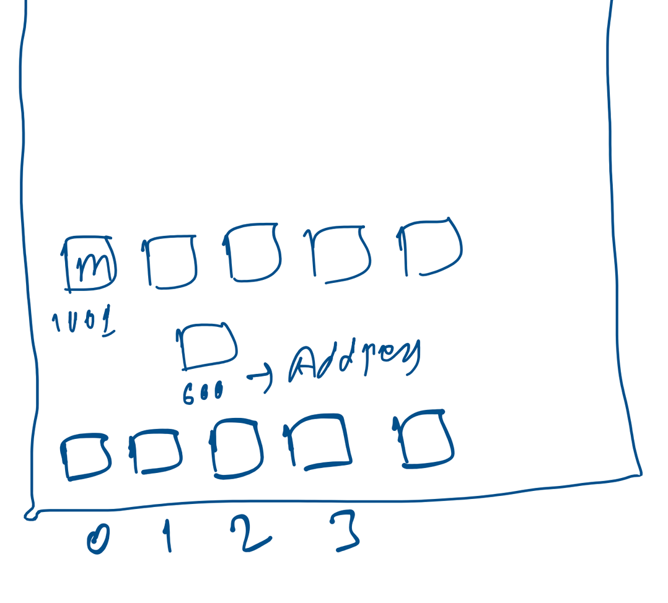
```
int i ;
char str [20] = " SAARBRÜCKEN" ;
```

option 1.

```
for ( i=0 ; str [i] ; i++ )
   print ( " %c ", str [i] );
```

option 2.

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



STRING FUNCTIONS

```
#include <string.h>
int strlen ( char * ) ;
char* strupr ( char * ) ;
char * strlwr ( char * ) ;
char * strrev ( char * ) ;
char * strcpy ( char * , char * ) ;
char * strcat ( char * , char * ) ;
int strcmp ( char * , char * ) ;
```

FUNCTION CALL BY PASSING STRING

```
f1 (" PATNA" )          void f1 ( char s [] )
f1 ( str )              {

                        }
```

HANDLING MULTIPLE STRINGS

```
char s[4][10] = { "PATNA", "MUMBAI", "BHOPAL", "DELHI" }
```

	0	1	2	3	4	5	6	7	8	9
s[0]	P	A	T	N	A					
s[1]	M	U	M	B	A	I				
s[2]	B	H	O	P	A	L				
s[3]	D	E	L	H	I					

write 5 city names ?

```
char s [5] [20] ;
int i ;
printf (" Enter 5 city name " ) ;
fgets ( s [i] , 20 , stdin ) ;
```