BITS Pilani
Hyderabad Campus

CS F111: Computer Programming

(Second Semester 2021-22)

Lect 18:Strings & Arrays

Nikumani Choudhury
Asst. Professor, Dept. of Computer Sc. & Information System

String Input/Output continued…

- <u>Using putchar ( ) and puts ( ):</u>

- char ch = ‘A’;
- putchar (ch);

$\Rightarrow$

char ch = ‘A’;
printf (“%c”, ch);

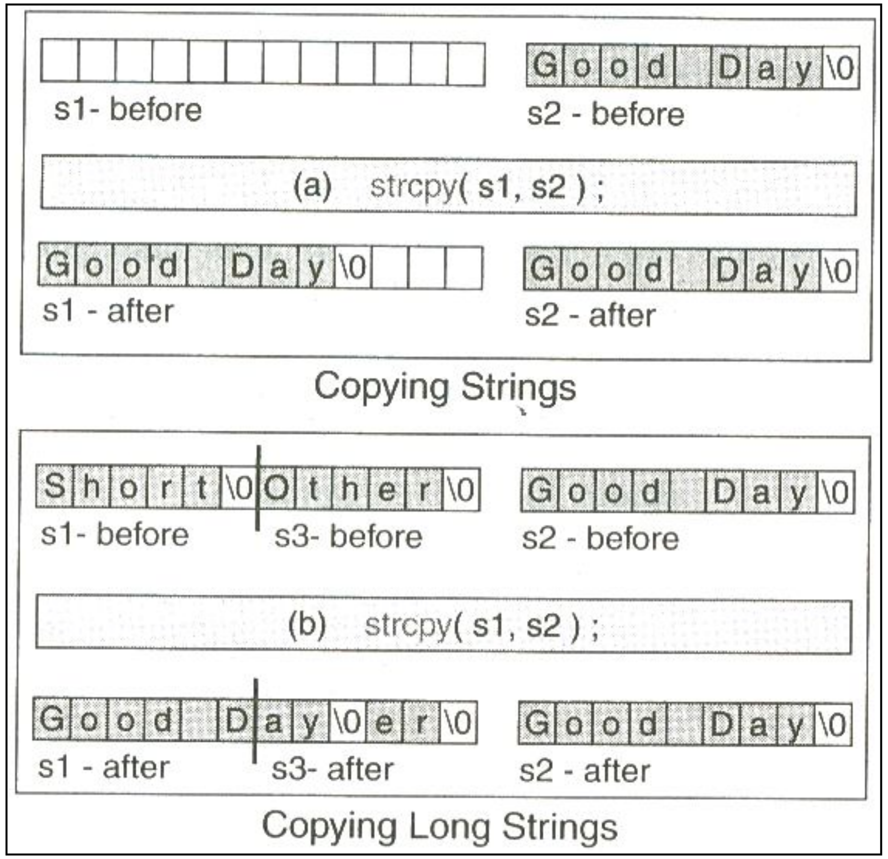
- char name[5] = “BITS”;
- for (i = 0; i < 4; i++)
-   putchar (name[i]);
- putchar (‘\n’);

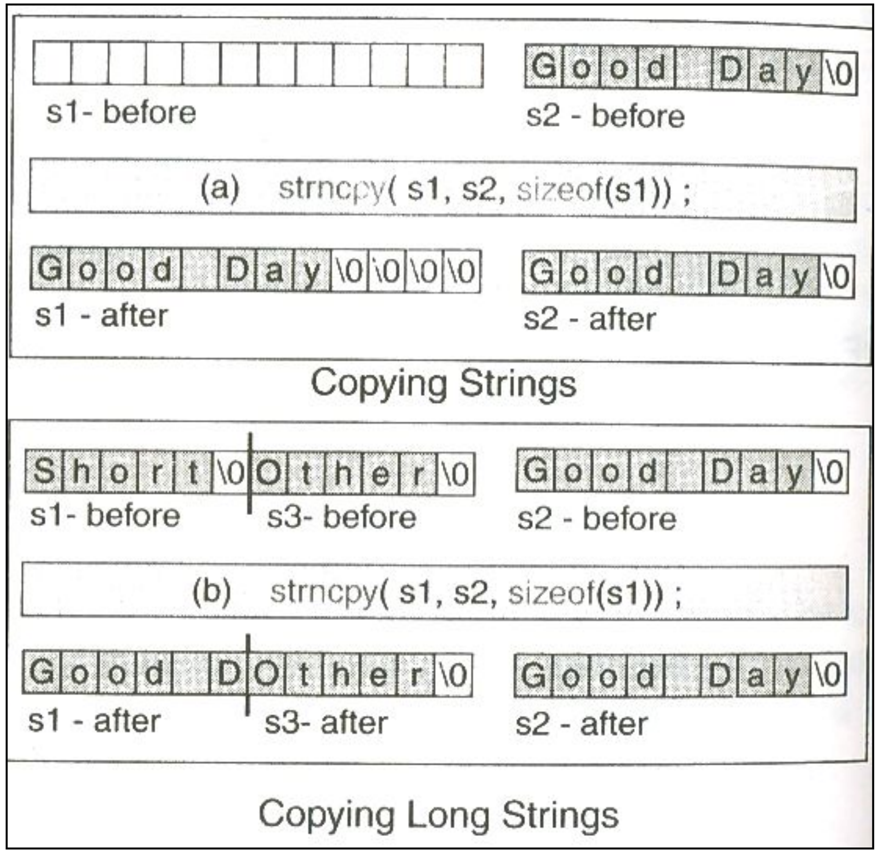
char line [30];
gets (line);
puts (line);

Arithmetic on Characters

- Whenever a character constant or variable is used in an expression, it is automatically converted to an integer value.
- x = ‘a’;
- printf (“%d\n”, x);
- arithmetic
  - x = ‘a’ – 1;
    printf (“%d\n”,x);
- A character can be converted to its equivalent integer by
  - x = character – ‘0’;
  - x = Ascii of ‘7’ – Ascii of ‘0’ = 55 – 48 = 7
- String of digits into their integer value (x = atoi (string);)

String Manipulation functions





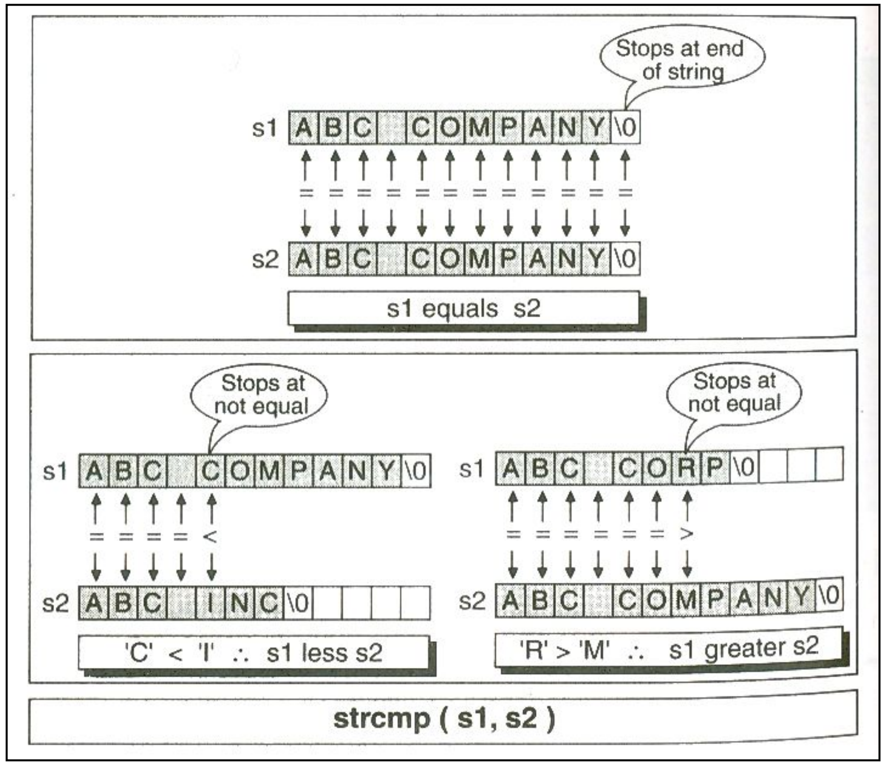
```
char   *strcpy  (  char   *s1,
const char *s2 );
```

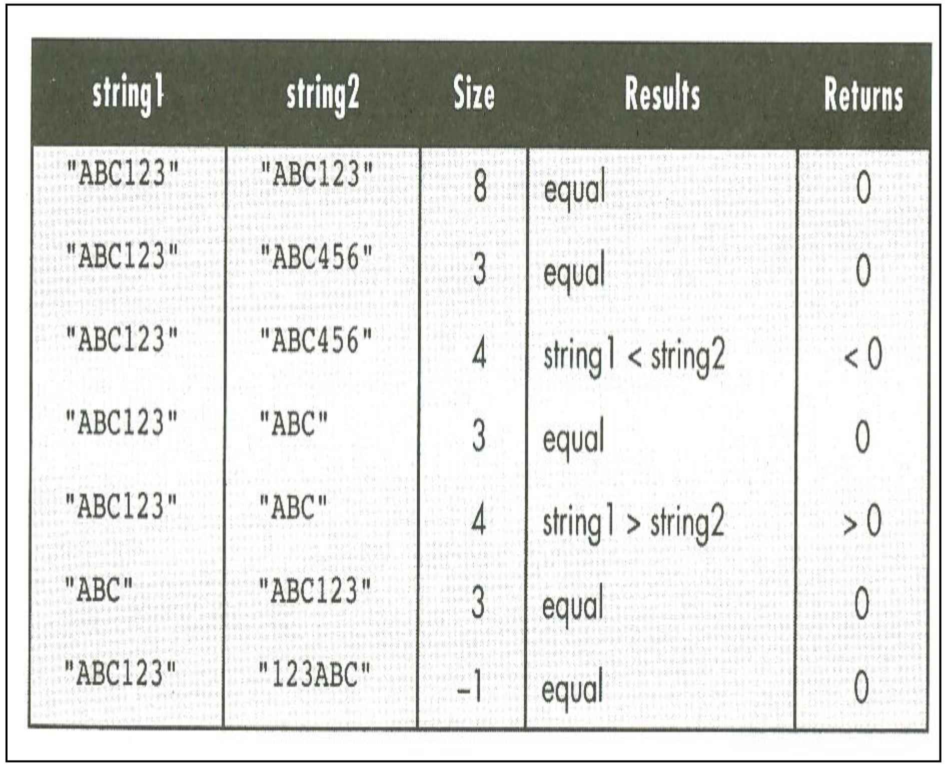
```
char  *strncpy  ( char  *s1,
const char *s2, size_t n );
```

- In the call strcpy(str1, str2), strcpy has no way to check that the s2 string will fit in the array pointed to by s1.
- If it doesn't, undefined behavior results.

safer

Continued…





string1	string2	Size	Results	Returns
"ABC123"	"ABC123"	8	equal	0
"ABC123"	"ABC456"	3	equal	0
"ABC123"	"ABC456"	4	string1 < string2	< 0
"ABC123"	"ABC"	3	equal	0
"ABC123"	"ABC"	4	string1 > string2	> 0
"ABC"	"ABC123"	3	equal	0
"ABC123"	"123ABC"	-1	equal	0

```
int strcmp ( const char *s1, const char *s2 );
```

```
int strncmp( const char *s1, const char *s2, size_t n );
```

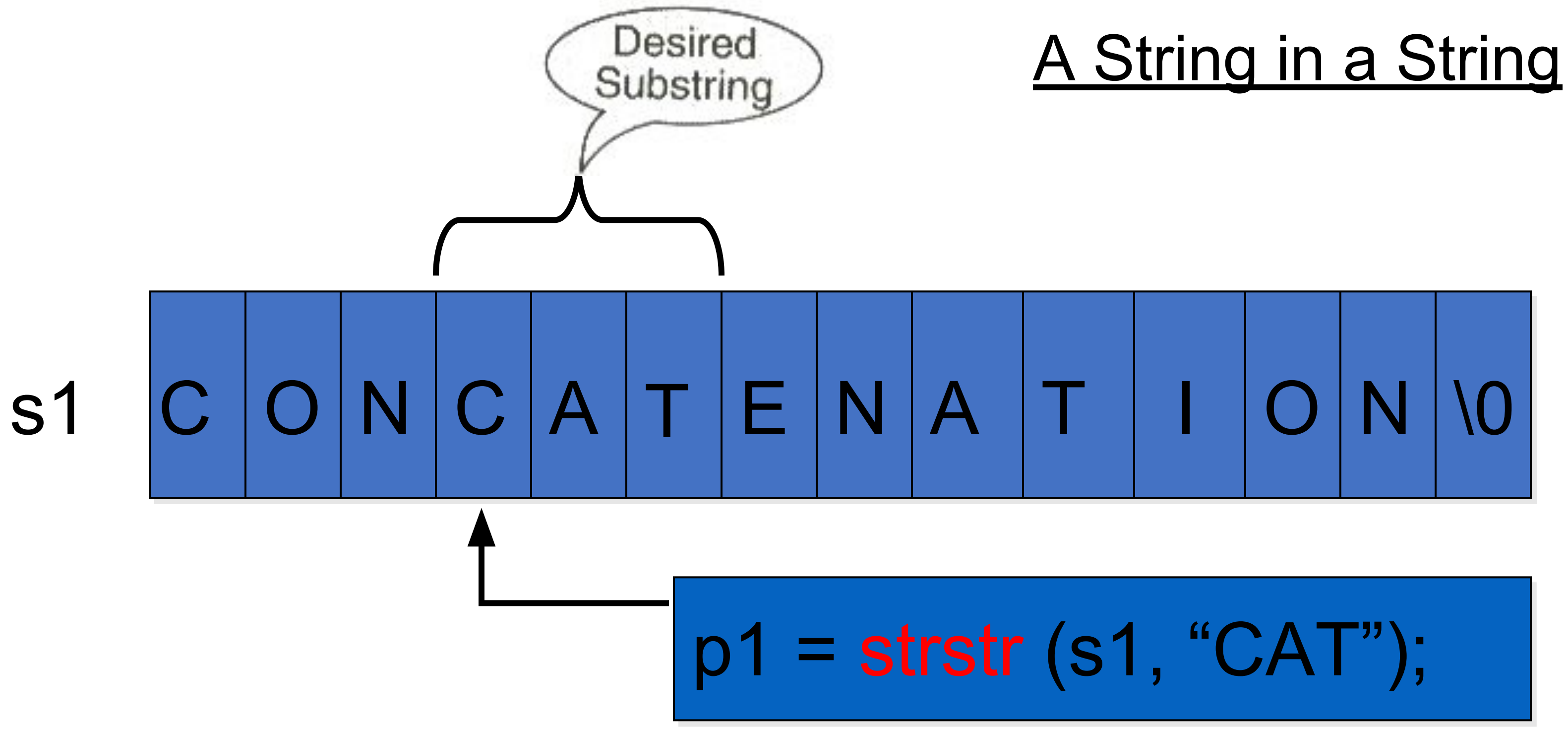
Continued…

String Length:

n = strlen (str);

Returns number of characters before '\0'

A String in a String



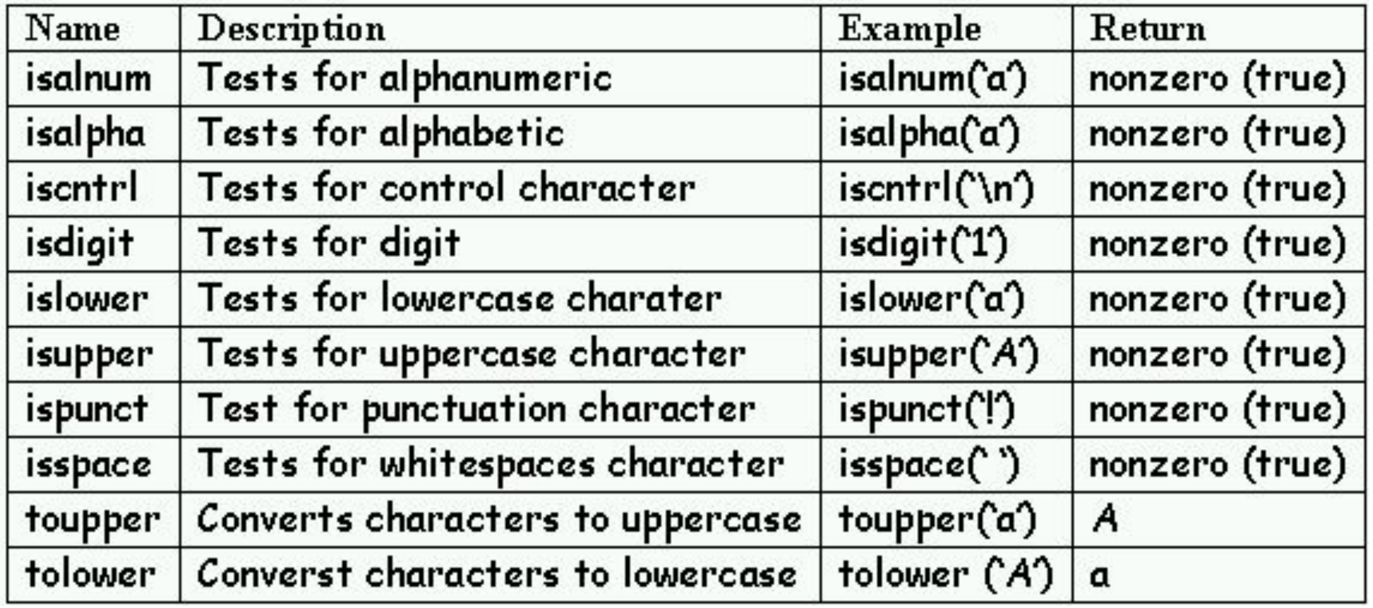
String handling Example

```
/*If two strings are not equal,
 concatenate or join them*/
scanf ("%s %s", s1, s2);
x = strcmp (s1, s2);
if ( x != 0 )
        strcat (s1, s2);
strcpy (s3, s1);
l1 = strlen (s1);
l2 = strlen (s2);
l3 = strlen (s3);
printf("%s %s %s", s1, s2, s3);
printf ("%d %d %d", l1, l2, l3);
```

```
char s1[10], s2[10], s3[10];
 int x, l1, l2, l3;
```

Character related functions

- In addition to the string processing functions, C also provides a family of character-related functions that facilitate character manipulations.
- To use these functions you need to #include <ctype.h> header file.
- List of some character related functions:



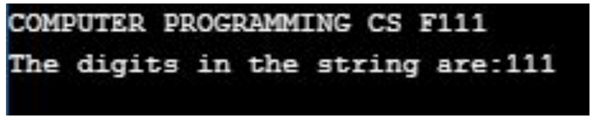
Name	Description	Example	Return
isalnum	Tests for alphanumeric	isalnum('a')	nonzero (true)
isalpha	Tests for alphabetic	isalpha('a')	nonzero (true)
iscntrl	Tests for control character	iscntrl('\n')	nonzero (true)
isdigit	Tests for digit	isdigit('1')	nonzero (true)
islower	Tests for lowercase charater	islower('a')	nonzero (true)
isupper	Tests for uppercase character	isupper('A')	nonzero (true)
ispunct	Test for punctuation character	ispunct('!')	nonzero (true)
isspace	Tests for whitespaces character	isspace(' ')	nonzero (true)
toupper	Converts characters to uppercase	toupper('a')	A
tolower	Converst characters to lowercase	tolower ('A')	a

An Example

```
#include<stdio.h>
#include<string.h>
#include<ctype.h>
int main(void)
{
   char s[]= "Computer programming CS F111";
   int len=strlen(s), i;

   for ( i=0 ; i< len ; i++)
         s[i]= toupper(s[i]);
   puts(s);

   printf("The digits in the string are:");
   for ( i=0 ; i< len ; i++){
     if(isdigit(s[i]))
         printf("%c", s[i]);
   }
return 0;
}
```



```
COMPUTER PROGRAMMING CS F111
The digits in the string are:111
```

String Manipulations: Without using string.h

Ex1:Length of a string

```
#include <stdio.h>
int main(){
  int i = 0;
  char text[100];
  char c;
  while ((c = getchar() ) != '\n'
                      && i <= 99)
  {
    text[i] = c;
    i++;
  }
  text[i] = '\0';
printf("Length of text : %d",i);
return 0;
}
```

```
#include <stdio.h>
int main(){
int i, j;
char first[15] = {"Computer"};
char second[15] = {"Programming"};
char result[30];
for (i = 0; first[i] != '\0'; i++)
   result[i] = first[i];
result[i] = ' ';
for (j = 0; second[j] != '\0'; j++)
   result[i+j+1] = second[j];
result[i+j+1] = '\0';
printf ("%s\n", result);
return 0;
}
```

Ex2:Combing two strings

Continued …

Ex3: Copying a string into another

```
#include<stdio.h>
int main(void) {
char src[20], dst[20];
int i = 0;
scanf("%[^\n]", src);

while (src [i] != '\0'){
  dst [i] = src [i];
  i++;
}
dst [i] = '\0';
printf ("%s", dst);
return 0;
}
```

Ex4: Extracting a substring

```
#include <stdio.h>
void main()
{
   char str[100], sstr[100];
   int pos, l, c = 0;
   printf("Input the string: ");
   fgets(str, sizeof str, stdin);
   printf("Input the position to start
                              extraction:");
   scanf("%d", &pos);
   printf("Input the length of
                               substring:");
   scanf("%d", &l);
   while (c < l)
   {
      sstr[c] = str[pos+c-1];
      c++;
   }
   sstr[c] = '\0';
   printf("The substring: \"%s\"\n\n",
                                     sstr);
}
```

Ex5: Sorting a string

```c
#include <stdio.h>
#include <string.h>
void main()
{
   char str[100],ch;
   int i,j,l;
   printf("Input the string : ");
   fgets(str, sizeof(str), stdin);
   l=strlen(str);
   /* sorting process */
   for(i=1;i<l;i++)
      for(j=0;j<l-i;j++)
         if(str[j]>str[j+1])
         {
            ch=str[j];
            str[j] = str[j+1];
            str[j+1]=ch;
         }
   printf("After sorting the string appears like : \n");
   printf("%s\n\n",str);
}
```

Arrays of Strings

```
char
planets[][8]=
{"Mercury",
"Venus",
"Earth",
"Mars",
"Jupiter",
"Saturn",
”Uranus",
"Neptune",
"Pluto"};
```

	0	1	2	3	4	5	6	7
0	M	e	r	c	u	r	y	\0
1	V	e	n	u	s	\0	\0	\0
2	E	a	r	t	h	\0	\0	\0
3	M	a	r	s	\0	\0	\0	\0
4	J	u	p	i	t	e	r	\0
5	S	a	t	u	r	n	\0	\0
6	U	r	a	n	u	s	\0	\0
7	N	e	p	t	u	n	e	\0
8	P	l	u	t	o	\0	\0	\0

Ex6: Sorting strings using bubble sort

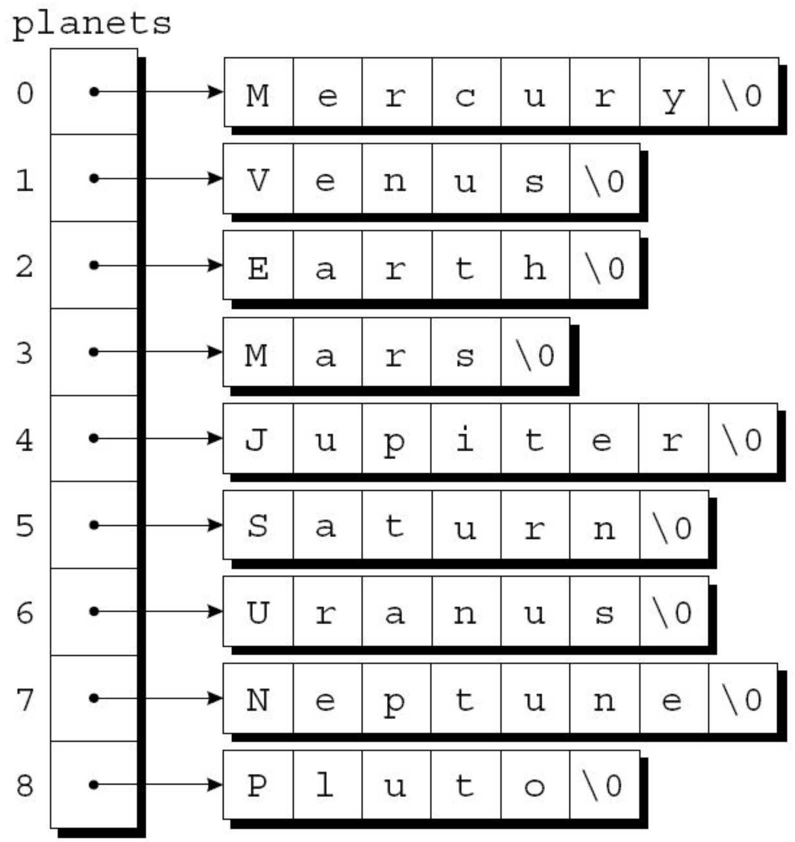
```c
#include <stdio.h>
#include <string.h>
void main(){
   char name[25][50],temp[25];
   int n,i,j;
   printf("Input number of strings :");
   scanf("%d",&n);
   printf("Input string %d :\n",n);
   for(i=0;i<=n;i++)
      fgets(name[i], sizeof name, stdin);
   for(i=1;i<=n;i++)
      for(j=0;j<=n-i;j++)
         if(strcmp(name[j],name[j+1])>0)
         {
            strcpy(temp,name[j]);
            strcpy(name[j],name[j+1]);
            strcpy(name[j+1],temp);
         }
   printf("The strings appears after sorting :\n");
   for(i=0;i<=n;i++)
      printf("%s\n",name[i]);
}
```

What we need is a **ragged/ jagged array,** whose rows can have different lengths.

Continued…

```
char
  *planets[]
  =
  {"Mercury"
  , "Venus",
  "Earth",
  "Mars",
  "Jupiter",
  "Saturn",
  "Uranus",
  "Neptune",
  "Pluto“

};
```



(Simulating a ragged array by creating an array whose elements are pointers to strings)

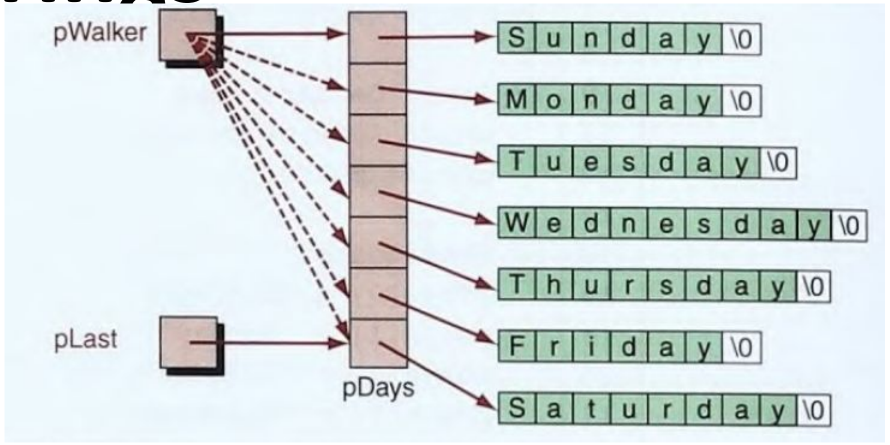
An Example of Array of Pointers to Strings

```
#include <stdio.h>
int main(){
  char* pDays[7];
  char** pLast;

  pDays[0] = "Sunday";
  pDays[1] = "Monday" ;
  pDays[2] = "Tuesday" ;
  pDays[3] = "Wednesday";
  pDays[4] = "Thursday";
  pDays[5] = "Friday" ;
  pDays[6] = "Saturday";
  printf( "The days of the week are:\n" );
  pLast = pDays + 6;

  for (char** pWalker=pDays; pWalker<=pLast; pWalker++)
     printf( "%s\n", *pWalker);

return 0;
}
```



```
int main(int argc, char *argv[])
```

```
The days of the
week are:
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
```