MODULE 4:

Strings

What is string?

• String is an array of characters terminated by NULL character which is denoted by the escape sequence '\0'.
• A String literal is a sequence of characters enclosed within two double quotes.
•Example "ACHARYA".
•Let us see how the string is stored:

The common operations performed on character strings

•Reading and writing strings.
•Combining strings together.
•Copying one string to another.
•Comparing strings for equality.
•Extracting a portion of a string.

Declaring and Initializing String Variables

Declaration: Let us see how to declare a string variable.

The general form :

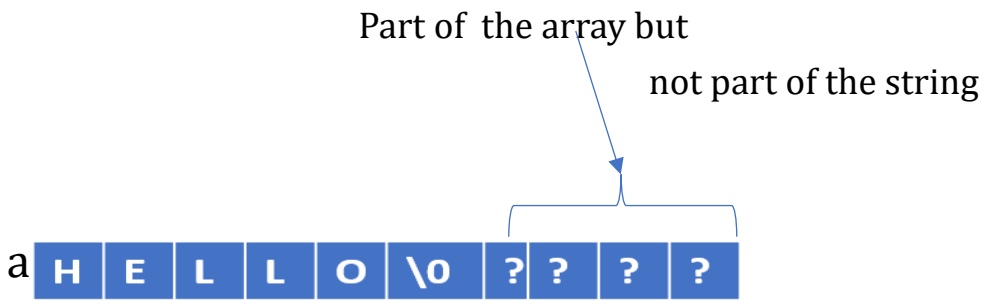
char string_name[size];

Example 1: char a[50];

0 1 2 3…… ... 49

Example 2 : char a[10]; /*10 bytes are allocated*/

strcpy(a, ”Hello”);



junk character

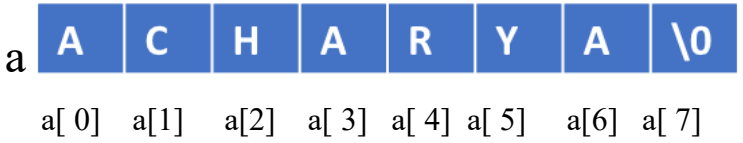
Initializing:

Initialization is the process of assigning values to a variable before doing manipulation.

The initialization can be done in various ways:

i. Initializing locations character by character.

char a[8]={'A','C','H','A','R','Y','A'};

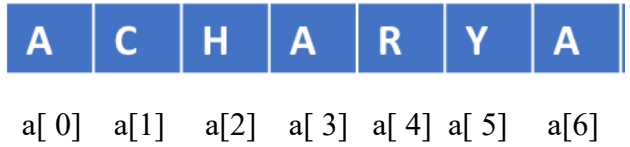


ii. Partial array initialization.

char a[8]={'I','N','S','T','I','T'};

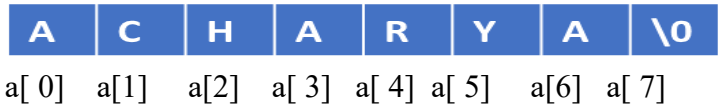
iii. Initialization without specifying the size.

char a[]={'A','C','H',A','R','Y','A'};



iv. Array initialization with a string.

char a[]="ACHARYA";



String input/output functions

The various I/O functions associated with strings can be classified as below.

i. Formatted input function -Ex: scanf()
ii. Formatted output function -Ex: printf()
iii. Unformatted input function -Ex: gets()
iv. Unformatted output function- Ex: puts()

i. Formatted input function -scanf()

- String conversion code (%s)

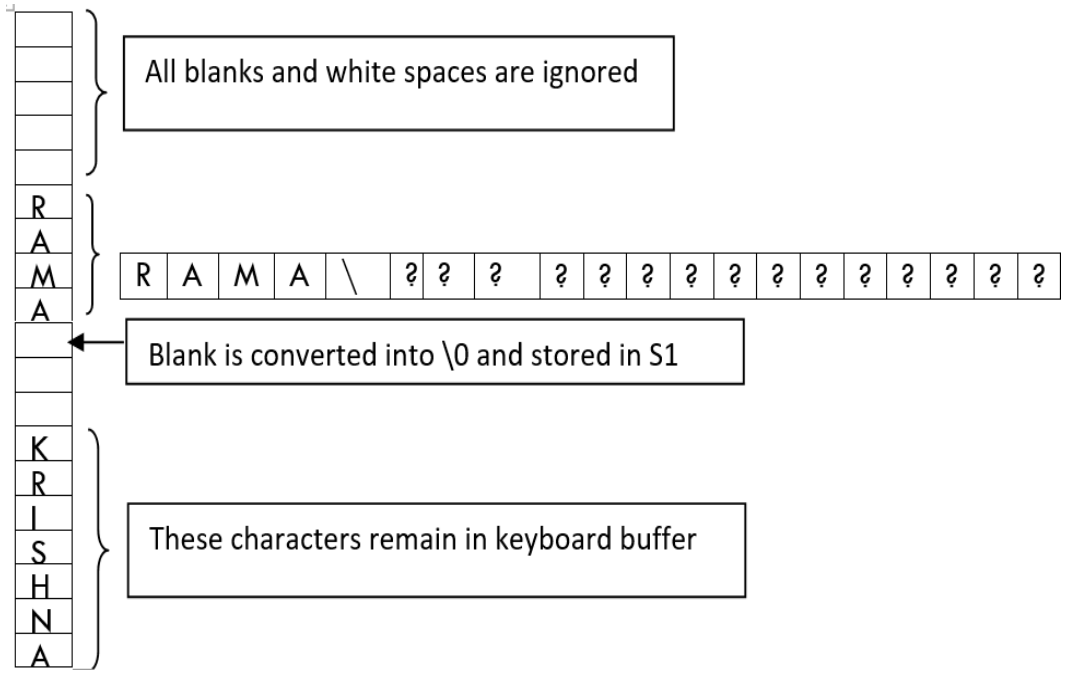
- The edit set conversion code(%[...] or %[^\n])

Example:

```
char s1[15];
scanf("%s",s1);
```

If the input through the keyboard is:

RAMA KRISHNA



Edit set conversion code[%[...])

Advantage using edit set conversion code, it is possible to enter a line of text with white spaces.

The syntax :

scanf("%[....]",str);

scanf("%[^\n]",str); /* read all character except \n*/

ii. Formatted Output function-printf()

•The printf function with %s can be used to display an array of characters that is terminated by the null character.

Example:

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

•Can be used to display the entire contents of the array name.

•We can also specify the precision with which the array is displayed.

For example, the specification

```
printf("%12.4s",text);
```

indicates that the first four characters are to printed in a field width of 12 columns.

```
printf("%-10.4s",text);
```

If we include the minus sign in the specification the string will be printed left-justified.

/*Program to read a series of words using scanf function*/

```
main()
{
char text1[50],text2[50],text3[50],text4[50];
printf(“Enter text:\n”);
scanf(“%s %s”, text1,text2);
scanf(“%s”, text3);
scanf(“%s”, text4);
printf(“\n”);
printf(“text1= %s\n text2=%s\n”, text1,text2);
printf(“text3= %s\n text4= %s\n”, text3,text4);
}
```

OUTPUT:

```
Enter text:
Acharya Institute of Technology
text1= Acharya
text2= Institute
text3= of
Text4= Technology
```

/*Program to illustrate writing strings using %s format */

```
main()
 {
char state[15]= “MADHYA PRADESH”;
printf(“\n \n”);
printf(“--------------------------------\n”);
printf(“%15s\n”, state);
printf(“%5s\n”, state); //When field width is less than the length of the string, the entire string is printed
printf(“%15.6s \n”, state); //first six characters are to be printed in a field width of 10 columns.
printf(“%-15.6s \n”, state); //String will br printed left justified
printf(“%15.0s\n”, state); //nothing is printed
printf(“%.3s\n”, state); //Specifies the number of characters to be printed
printf(“%s\n”, state);
}
```

OUTPUT:

```
MADHYA PRADESH
MADHYA PRADESH
          MADHYA
MADHYA

MAD
MADHYA PRADESH
```

iii. Unformatted input function -Ex: getchar(), gets()

- getchar function can use to read successive single character from the input and place them into the char array.

Example:

```
/* Reading character */
char ch,line[50];
ch=getchar();

/* Reading line of text can be read and stored in an array */
do
{
character = getchar();
line[c]=character;
c++;
}
```

- Reads a line of text from the keyboard and display it on the screen.

gets(str);

Example:

```
char line[80];
gets(line);
printf ("%s",line);
```

iv. Unformatted output function- Ex: putchar(),puts()

- C supports another character handling function putchar to output the value of character variable.

Example : char ch='A';
putchar(ch); or printf("%c",ch);

```
Char name[6]="PARIS";
for(i=0;i<5;i++)
 putchar(name[i]);
putchar("\n");
```

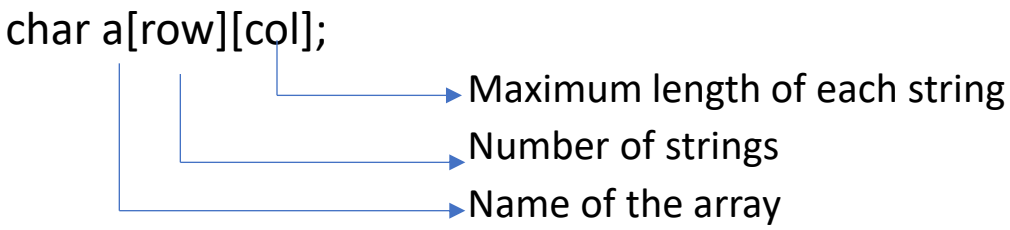
- Printing string values is to use the function **puts.**

puts(str);

Example: char line[80];
gets(line);
puts(line);

Arrays of strings

- To create an array of string, we should use two-dimensional array as shown below.
  Example:



- The memory representation for the following initialization

```
Char a[5][11]={
                "DHARMARAYA",
                "BHIMA",
                "ARJUNA",
                "NAKULA",
               "SAHADEVA"
              }
```

a

	0	1	2	3	4	5	6	7	8	9	10
0	D	H	A	R	M	A	R	A	Y	A	\0
1	B	H	I	M	A	\0					
2	A	R	J	U	N	A	\0				
3	N	A	K	U	L	A	\0				
4	S	A	H	A	D	E	V	A	\0		

The Programming statement to read 5 names can be written as shown below :

```
for(i=0;i<=4;i++)
{
        scanf("%s",a[i]);
}
```

The array elements can be printed as shown below:

```
for(i=0;i<=4;i++)
{
        printf("%s\n",a[i]);
}
```

Arithmetic Operations on Characters

- C allows us to manipulate characters the same way we do with numbers.
- Whenever a character constant or character variable is used in an expression, it is automatically convert into an integer value by the system.
- It is also possible to perform arithmetic operation on the character constants and variables.

  Example 1: x='z'-1=122-1=121;

- We may also use character constants in relational expressions.

  Example 2: ch>='A' && ch<='Z'

- We can convert a character digit to its equivalent integer value

  Example 3: x='7'-'0'=55-48=7

- The function **atoi** converts the string to its numeric equivalent.

  y=atoi(string)

String handling functions

- C library supports a large number of string functions. The list given below depicts the string functions.
- All string functions are defined in header file “string.h”

String functions	Description of each function
strlen(str)	Returns length of the string str
strcpy(dest,src)	Copies the content of source string src to destination string dest.
strcmp(str1,str2)	Compares two strings str1 and str2
strcat(str1,str2)	Append string str2 to string str1
strncpy(dest,src,n)	Copies at most n characters of the source string src to destination string dest
strncmp(s1,s2,n)	This compares the left-most n characters of s1 to s2 and returns.
strncat(str1,str2,n)	Append first n characters of string str2 to string str1.
strstr(s1,s2)	Searchs the string s1 to see whether the string s2 is contained in s1
strrev(str)	Reverse the string
strlwr(str)	Converts the string str to lowercase
Strupr(str)	Converts the string str to uppercase

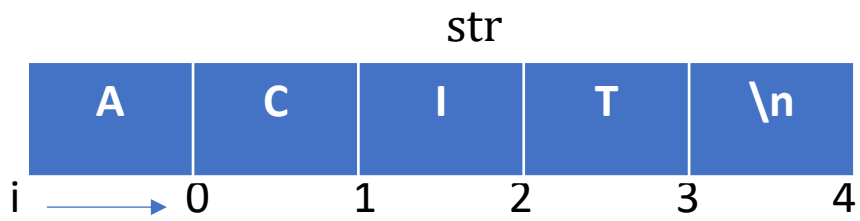
strlen(str):String Length

The syntax to use strlen() is

strlen(str)

•This function returns the length of the string str i.e. it counts all the characters up to '\0' except '\0'.
•So, an empty string has length zero.

Consider the string "ACIT"




```
i=0;
while(str[i]!='\0')
 i++
```

Built_in function	User-defined function

#include<stdio.h>
#include<string.h>
void main()
{
char str[]="RAMA";
printf("Length = %d\n",strlen(str));
}
 	
OUTPUT
Length=4
	#include<stdio.h>
	#include<string.h>
	Void main()
	{
	char str[20];
	int i = 0;
	printf("Enter the string\n");
	gets(str);
	i=0;
	while(str[i] != '\0')
	{
	i + +; \\0 1 2 3
	}
	printf("Length = %d\n",i); //3
	}
	
	INPUT
	AIT
	
	OUTPUT
	Length=3

strcat(s1,s2)-string concatenate

Syntax: It is defined in “string.h”.

strcat(char s1[], char s2[]);

Where:

s1 is the first string
s2 is the second string

- The function copies all the characters of string s2 to the end of string s1.

After strcat(s1, s2);

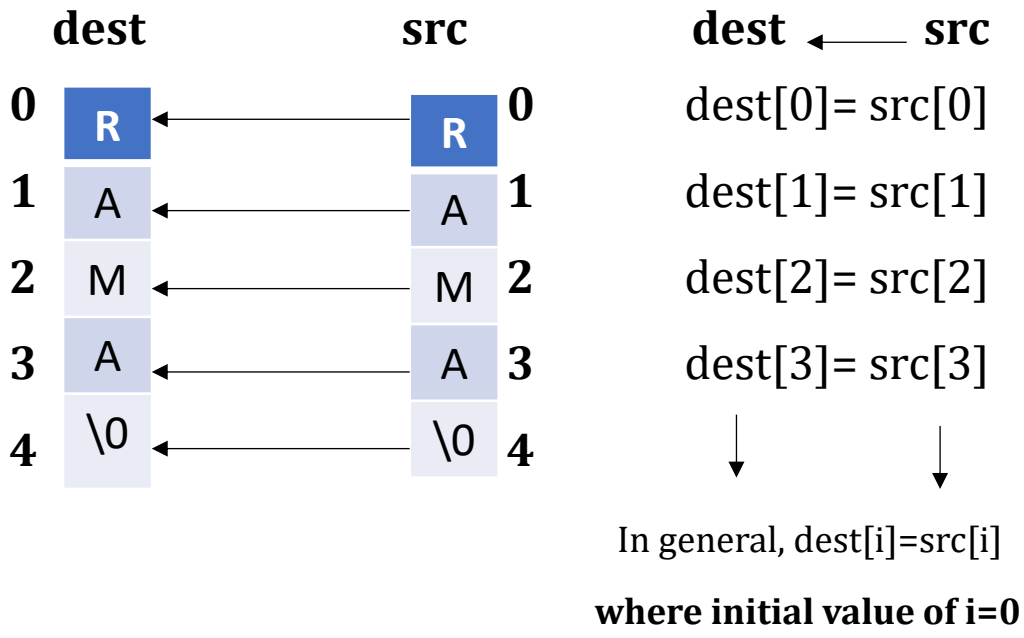
Built_in function	User defined function

#include <stdio.h>
 	
void main()
{
 	
char s1[15]=“RAMA”;
Char s2[]=“KRISHNA”;
 	
strcat(s1,s2);
 	
printf(“Con string =%s\n”,s1);
 	
}
 	
OUTPUT:
 	
Concatenated string=
RAMAKRISHNA
	#include<stdio.h>
	
	Void main()
	{
	
	int i, j;
	printf("Enter the string1\n");
	scanf("%[^\n]", str1);
	printf("Enter the string2\n");
	scanf("%[^\n]", str2);
	i = 0;
	while(str1[i] != '\0’)
	{
	i + +;
	}
	
	j=0;
	while(str2[j] != '\0’)
	{
	str1[i] = str2[j];
	i + +;
	j + +;
	}
	str1[i] = '\0’;
	}
	printf("Concat string = %s\n", str1);
	}

The syntax to use strcpy() is

strcpy(char dest[], char src[])

•The function strcpy copies the contents of source string src to destination dest including '\0'.
•So, the size of destination string **dest** should be greater or equal to the size of source string **src**.



Built_in function	User-defined function

```
#include<stdio.h>
#include<string.h>
void main()
{
   char src[]="RAMA";
   char dest[6];
   strcpy(dest,src);
   printf("Dest string = %s\n",dest);
}
```

INPUT

RAMA

OUTPUT

Dest string = RAMA

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

Void main()
{
  char src[20];
  char dest[20];
  int i = 0;
  printf("Enter the string\n");
  gets(src);

  while( src[i] != '\0')
  {
    dest[i]=src[i];
    i + +;
   }
    dest[i]='\0';

  printf("Dest String = %s\n",dest);
}
```

INPUT

Enter the string

RAMA

OUTPUT

Dest string = RAMA

strcmp(s1,s2)-string compare

Syntax: int strcmp(char s1[],char s2[]);

Where

s1 is the first string

s2 is the second string

•This function is used to compare two strings.
•The comparison starts with first character of each string.
•The comparision continues till the corresponding characters differ or until the end of the character reached.
The following values are returned after comparison:

if s1=s2→zero
if s1>s2→positive
if s1<s2→ negative

0	R	==	R	**0**	dest[0]== src[0]
1	A	==	A	**1**	dest[1]== src[1]
2	M	==	M	**2**	dest[2]= =src[2]
3	A	==	A	**3**	dest[3]= =src[3]
4	\0	==	\0	**4**	

In general, dest[i]==src[i]
where initial value of i=0

Built_in function	User-defined function

#include<stdio.h>
#include<string.h>
void main()
{
char s1[]=“RAMA”;
char s2[]=“KRISHNA”;
 	
Int difference;
difference=strcmp(s1,s2);
 	
If(difference==0)
Printf(%s = %s\n”,s1,s2);
else if (difference>0)
 	
Printf(“%s>%s\n”,s1,s2);
 	
else
 	
Printf(“%s<%s\n”,s1,s2);
}
	#include<stdio.h>
	#include<string.h>
	Void main()
	{
	Int difference,I;
	char str1[20], str2[20];
	
	printf("Enter the string1\n"); scanf("%[^\n]", str1); // gets(str1);
	
	printf("Enter the string2\n"); scanf("%[^\n]", str2); // gets(str2);
	
	i=0;
	While(s1[i]==s2[i])
	{
	If(s1[i]== ‘\0’)
	break;
	i++;
	}
	difference=s1[i]-s2[i];
	if(difference = = 0)
	printf("%s = %s\n", s1, s2);
	else if (difference >0)
	printf("%s > %s\n",s1, s2);
	else
	printf("%s < %s\n", s1, s2);
	}

Strrev(str)-String reverse

Syntax: strrev(str)

where
str is the given string

This function reverses all characters in the string str except the terminating NULL character '\0'.

Built_in function	User-defined function

#include<stdio.h>
#include<string.h>
void main()
{
char str[]="INDIA";
 	
strrev(str);
 	
printf("Rev String = %s\n",str);
}
 	
INPUT
INDIA
 	
OUTPUT
Rev String = AIDNI
	#include<stdio.h>
	#include<string.h>
	Void main()
	{
	char src[20],dest[20];
	int i,n;
	printf("Enter the string\n");
	gets(src);
	n=strlen(src)
	for(i=0;i<n;i++)
	{
	dest[n-1-i]=src[i];
	}
	dest[n]='\0';
	printf("Source Destination\n");
	printf("%-5s\t\t%5s\",src,dest);
	}
	INPUT
	INDIA
	
	OUTPUT
	Source Destination
	INDIA AIDNI

/*Program to convert lowercase characters in to upper case characters*/

```
#include<stdio.h>
main()
 {
        char text[85];
        int i=0;
        printf(“Enter a line of text in lowercase:\t”);
        scanf(“%[^\n]”,text);
        printf(“%s”,text);
        printf(“\n Converted to uppercase text is :\t”);
        while(text[i]!=’0’)
        {
                printf(“%c”, toupper(text[i]));
                i++;
        }
        printf(“\n”);
}
```

/* Write a program to find the number of vowels and consonants in a text string */

```
#include <stdio.h>
#include<string.h>
void main()
{
char str[30];
int vow=0, cons=0,i=0;
printf(" Enter a string :");
gets(str);
while(str[i]!='\0')
{
if(str[i]=='a' || str[i]=='A' || str[i]=='e' || str[i]=='E' ||
   str[i]=='i' || str[i]== 'I' || str[i]=='o' ||  str[i]=='O' ||
   str[i]=='u' || str[i]== 'U')
     vow++;
else
     cons++;
i++;
}
printf("\n Number of vowels=%d",vow);
Printf("\n Number of consonants=%d",cons);
getch();
}
```


```
/*Program to sort strings in alphabetical order */
#define ITEMS 10
#define MAX 25
main()
{
        char str [ITEMS][MAX], dum[MAX];   int i=0;j=0;
        printf("Enter names of %d items \n", ITEMS);
        while(i<ITEMS)
        scanf("%s", str[i++]);
        for(i=1;i<ITEMS;i++)
        {
          for(j=1;j<=ITEMS-i;j++)
          {
            if(strcmp(string[j-1],string[j])>0)
                {
                 strcpy(dummy,string[j-1]);
                 strcpy(str[j-1],str[j]);
                 strcpy(str[j],dummy);
                 }
          }
        }
for(i=0;i<ITEMS;i++)
printf("%s", str[i]);
}
```


```
/*Program to show string handling functions */
#include<string.h>
 main()
 {
            char s1[20],s2[20],s3[20];
            int y,len1,len2,len3;
            printf("\n Enter two string constants\n");
            printf("?");   scanf("%s %s", s1,s2);
            x=strcmp(s1,s2);
            If(y!=0)
            {
            printf("\n\n Strings are not equal\n");
            strcat(s1,s2);
            }
            else
            printf("\n\n Strings are equal\n");
            strcpy(s3,s1);
            len1=strlen(s1);
            len2=strlen(s2);
            len3=strlen(s3);
              printf("\n s1= %s length= %d character \n",s1,len1);
              printf("\n s2= %s length= %d character \n",s2,len2);
              printf("\n s3= %s length= %d character \n",s3,len3);
}
```

ACHARYA