Strings

Strings are treated as arrays of type char.

A character array is a string if it ends with a null character('\0') .

String Constant/String Literals

A string constant is a sequence of characters enclosed in double quotes.

The double quotes are not a part of the string.

Some examples are:

"M"

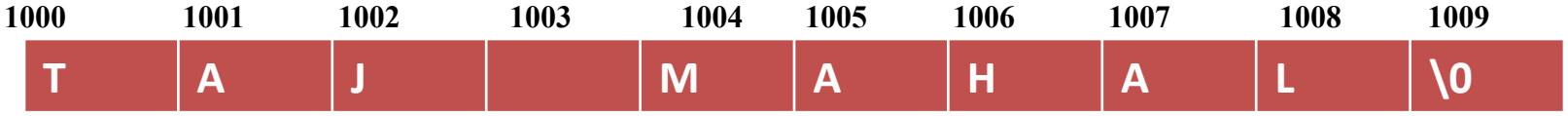
"Taj Mahal"

"2345"

"Subhas Chandra Bose was a great leader"

Whenever a string constant is written anywhere in the program, it is stored somewhere in memory as an array of characters terminated by a null character('\0').

The string constant itself becomes a pointer to the first character in the array.



such character occupies one byte and compiler automatically inserts the null character at the end.

The header file used is <string.h>

String Library Function

strlen(s1)	returns the length of the string s1	
strcmp(s1,s2)	returns a value	<0 when s1<s2
		=0 when s1=s2
		>0 when s1>s2
strcpy(s1,s2)	copies s2 to s1	
strcat(s1,s2)	concatenates s2 at the end of s1	

Program to test whether the word is palindrome or not.

```
main()
{
   int i,j;
  char word[];
     printf("Enter a word to test whether it is palindrome or not");
     scanf("%s",word);
     i=strlen(word);
     i--;
     j=0;
     while(j<=i)
     {
       if(word[i]!=word[j])
       {
                printf("The entered word is not palindrome");
       }
     i--;
     j++;
     }
 }
```

WAP to filter only the alphabets from a string

```
main()
{
    char str1[20],str2[20];
    int i,k,j=0;
    printf("Enter a word");
    scanf("%s",str1);
    k=strlen(str1);
    printf("The original string is %s\n", str1);
    for(i=0;i<k;i++)
    {
            if (str1[i]>='a'&& str1[i]<='z')||str1[i]>='A'&& str1[i]<='Z')
            {
            str2[j]=str1[i];
            j++;
            }
     }
str2[j]='\0' ;
    printf("After filteration the new string is %s\n",str2);
}
```

Output

Enter a word: a12b34c56d67e

The original string is: a12b34c56d67e

After filteration the new string is: abcde

WAP to convert the lowercase letters into uppercase

```
main()
{
char str[20],l;
int  i,k;
printf(“Enter a word:”);
scanf(“%s”,str);
k=strlen(str);
printf(“before conversion the string is :%s\n”,str”);
for(i=0;i<k;i++)
        {
if(str[i]>=‘a’&& str[i]<=‘z’)
str[i]=str[i]-32;
        }
printf(“After Conversion the string is: %s\n”,str);
}
```

Output

Enter a word: new york

Before conversion the string is : new york

After conversion the string is :NEW YORK

- WAP to concatenate two strings and create a new string.

```
main()
{
        char s1[20],s2[20],s3[40];
        int i,j,k;
        printf("Enter the first string");
        scanf("%s",s1);
         printf("Enter the second string");
        scanf("%s",s2);
        i=strlen(s1);
         j=strlen(s2);
        for(k=0;k<i ;k++)
                s3[k]=s1[k];
        for(k=0;k<j;k++)
                 s3[i+k]=s1[k];
s3[i+j]=s'\0';
    printf("Two strings are %s and %s", s1,s2,s3);
}
```

Output

Enter the first string : good

Enter the second string : morning

Two strings are good and morning

New string is goodmorning

WAP to unscramble a four character string

```
main()
    {
        char word[4];
        int i,j,k;
        begin:
                scanf("%s",word);
                if(strlen(word)!=4)
        goto begin;
                for(i=0;i<4;i++)
                        {
                                for(j=0;j<4;j++)
                                if(i !=j)
                                for(k=0;k<4;k++)
                                if(i !=k)
                                 if(j!=k)
                printf("%c%c%c\n", word[i],wor[j],word[k],word[6-(i+j+k)]);
                printf("\n");
                        }
    }
```

Output

Enter four letter word :pots

post
ptos
ptso
psot
psto
opts
opst
otps
otsp
ospt
ostp
tpos
tpso
tops
tosp
tspo
tsop
spot
spto
sopt
sotp
stpo
stop

String Variables

```
main()
{
      char str[]="INDIA";
      int i ;
      for(i=0;str[i]!='\0';i++)
      {
      printf("character=%c\t",str[i]);
       printf("address=%u\t",&str[i]);
      }
}
```

arr[0]	arr[1]	arr[2]	arr[3]	arr[4]	arr[5]
I	N	D	I	A	\0
1000	1001	1002	1003	1004	1005

Output

Character=I	Address=1000
Character=N	Address=1001
Character=D	Address=1002
Character=I	Address=1003
Character=A	Address=1004

```
main()
{
    char str[]="INDIA";
    char *p ;
    p=str;
    while(*p!='\0')
    {
    printf("character=%c\t",*p);
     printf("address=%u\n",p);
    p++;
   }
}
```

```
main()
{
    char  name[20];
     printf(“Enter name:”);
    gets(name);
     printf(“Entered  name is:”);
    puts(name);

    }
```

Output:

Enter name: Leo Tolstoy

Entered name is: Leo Tolstoy

Note: For entering strings with whitespaces we can use the function gets().

For getting the output of the string .

strlen(): It is a string library function.

```
main()
{
      char str[20];
      int length;
       printf("Enter the string:");
      scanf("%s",str);
      length=strlen(str);
       printf("Length of the string is:%d\n",length);
      }
```

Output
Enter a string: Dronacharya
Length of the string is : 8

Creation of this function

Array Version

```
int  astrlen(char str[ ])
{
     int i=0;
 while(str[i]!='\0')
      i++;
      return i ;
}
```

Pointer Version

```
int  pstrlen(char str[ ])
{
            char *start=str;
while(*str[i]!='\0')
            str++;
            return(str-start);
}
```

Strcmp(): It is a string library function for comparing two strings.

```
main()
{
    char str1[10], str2[10];
     printf("Enter the first string:");
    scanf("%s",str1);
     printf("Enter the second string:");
    scanf("%s",str2);
    if((strcmp(str1,str2))= =0)
    printf("Strings are same\n");
else
     printf("Strings are not same\n");
    }
```

Output

Enter first string: Banglore

Enter second string: Manglore

Strings are not same

Creation of this function

Array Version

```
int  astrmp(char str1[ ] ,char str2[ ])
{
     int i=0;
 while(str1[i]!='\0'&& str2[i]!='\0'&& str1[i]== str2[i])
      i++;
if(str1[i]==str2[i])
          return  0 ;
else
return(str1[i]-str2[i]);
}
```

Pointer Version

```
int  pstrmp(char *str1 ,char *str2)
{
  while(*str1[i]!='\0'&& *str2[i]!='\0'&& *str1== *str2)
    str1++;
    str2++;
if(*str1==*str2)
          return  0 ;
else
return(*str1-*str2);
}
```

strcpy(): It is a string library function for copying one string to another

strcpy(str1,str2) copies str2 to str1 .str2 is the source string and str1 is destination string.

```
main()
{
    char str1[10], str2[10];
     printf("Enter the first string:");
    scanf("%s",str1);
     printf("Enter the second string:");
    scanf("%s",str2);
    strcpy(str1,str2);
    printf("First strings :%s \t\t Second string: %s\n",str1,str2);
    strcpy(str1,"Delhi");
     strcpy(str2,"Calcutta");
     printf("First strings :%s \t\t Second string: %s\n",str1,str2);
    }
```

Output

```
Enter first string: Bombay
Enter second string: Mumbai
First String: Mumbai          Second string: Mumbai
First String: Delhi           Second string: Calcutta
```

Creation of this function

Array Version

```
char * astrcpy(char str1[ ] ,char str2[ ])
{
     int i=0;
 while(str2[i] !='\0')
   {
          str1[i]=str2[i] ; /*copy character by character*/
            i++;
}
str1[i]='\0';
return str1;
}
```

Pointer Version

```
char *pstrcpy(char *str1 ,char *str2)
{
   while(*str2[i]!='\0')
    * str1= *str2;
str1++  ;
str2++  ;
}
*str1= '\0'
return str1;
}
```

strcat(): It is a string library function for concatenating two strings

strcpy(str1,str2) copies str2 to str1 .str2 is the source string and str1 is destination string.

```
main()
{
    char str1[10], str2[10];
     printf("Enter the first string:");
    scanf("%s",str1);
     printf("Enter the second string:");
    scanf("%s",str2);
    strcat(str1,str2);
    printf("First strings :%s \t\t Second string: %s\n",str1,str2);
    strcat(str1,"_one");
     printf("Now first strings  is:%s \n,str1);
    }
```

Output

Enter first string: data

Enter second string: base

First String: database Second string: base

Now first String is : database_one

Creation of this function

Array Version

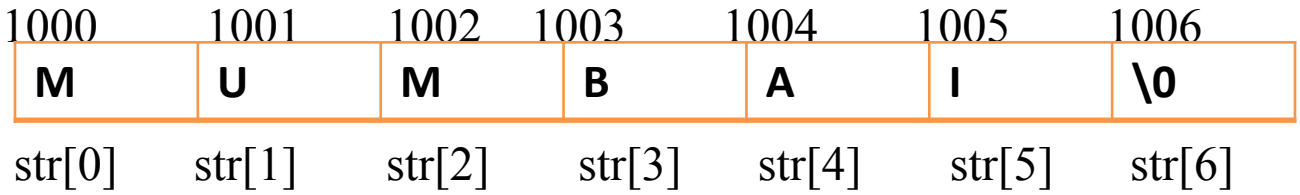
```
char * astrcat(char str1[ ] ,char str2[ ])
{
     int i=0, j=0 ;
 while(str1[i] !='\0')  /* check for the end of first string*/
          i++;
 while(str2[j] !='\0')
   {
          str1[i]=str2[j] ; /* Add second string at the end of first*/
            i++;
            j++;
}
str1[i]='\0';
return str1;
}
```

Pointer Version

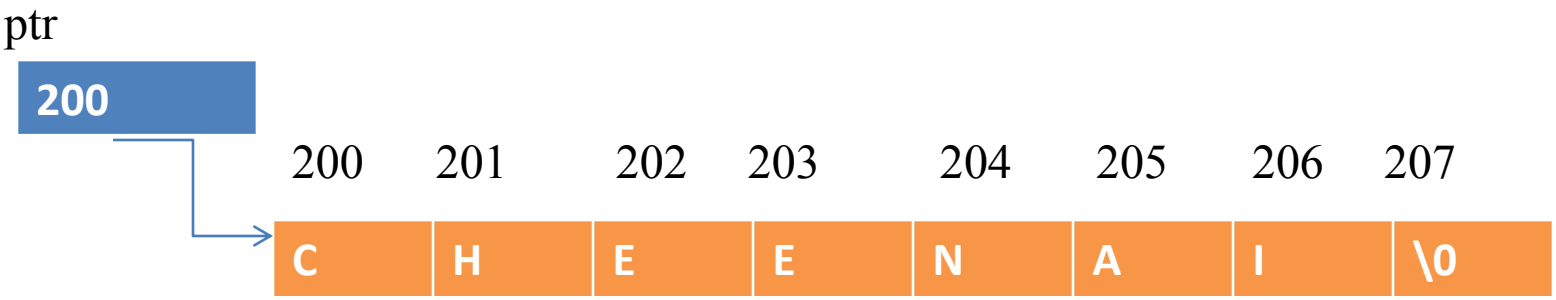
```
char *pstrcat(char *str1 ,char *str2)
{
   while(*str1!='\0')
     str1++;
  while(*str2!='\0')
{
          *str1=*str2 ;
          str1++;
          str2++  ;
}
*str1= '\0'
return str1;
}
```

String Pointers

char str[]="MUMBAI";



char *ptr ="chennai";



str is a constant pointer and will always contain address 1000 while ptr is a pointer variable and may contain any other address.

Array of Strings or Two Dimensional Array of Characters

```
char  arr[5][10]=          {
                                "WHITE",
                                "RED",
                                "GREEN",
                                "YELLOW",
                                "BLUE"
                           };
```

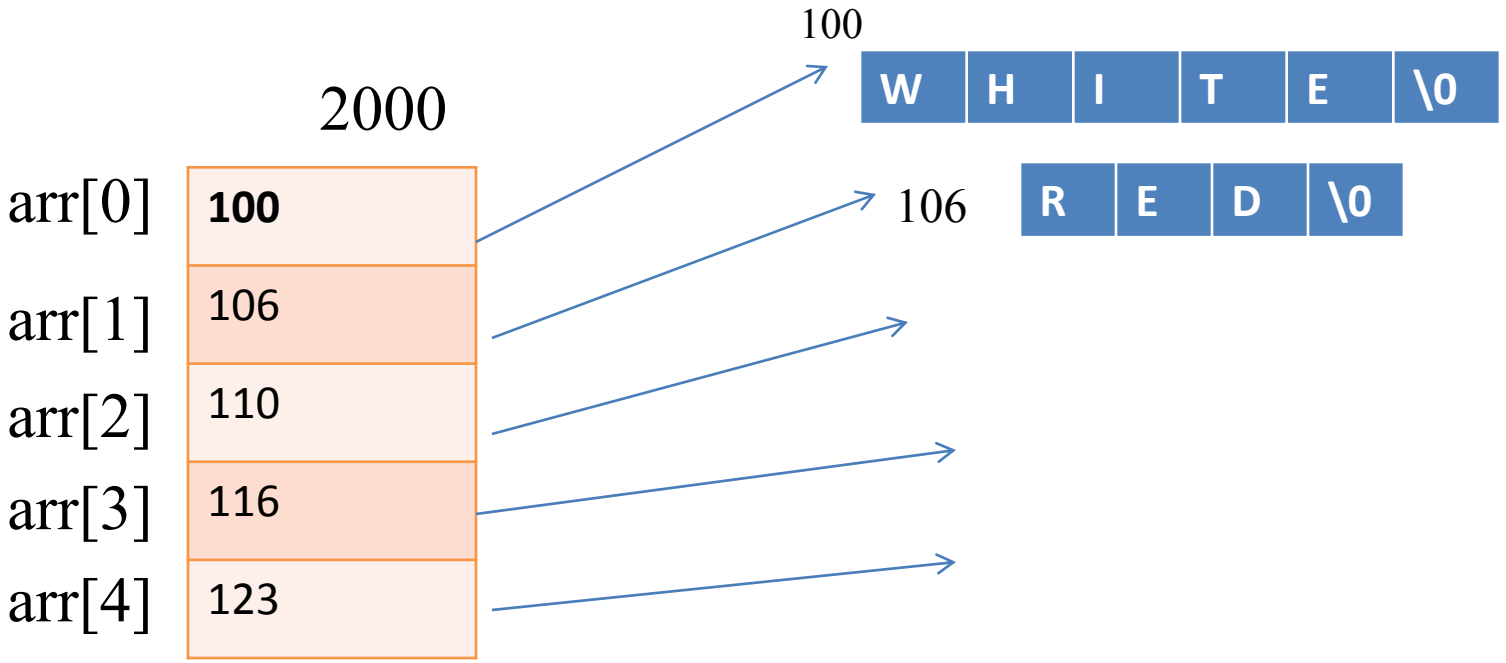
2000	W	H	I	T	E	\0				2009
2010	R	E	D	\0						2019
	G	R	E	E	N	\0				
	Y	E	L	L	O	W	\0			
	B	L	U	E	\0					

```
#include<stdio.h>
# define N 5
# define LEN 10
      main()
{
      char arr[N][LEN]=              {
                                                  "WHITE",
                                                  "RED",
                                                  "GREEN",
                                                  "YELLOW",
                                                  "BLUE"
                                    };
  int  i ;
      for(i=0;i<N; i++)
      {
            printf("String=%s\t",arr[i]);
            printf("Address of String=%u\n", arr[i]);
      }
```

Output:

String=white Address of String =2000
String=red Address of String =2010
String=green Address of String =2020
String=yellow Address of String =2030
String=blue Address of String =2040

Array of pointers to strings



Array of pointers to strings

```
#include<stdio.h>
main()
{
      int  i ;
      char *arrp[ ] ={
                                                "WHITE",
                                                "RED",
                                                "GREEN",
                                                "YELLOW",
                                                "BLUE"
                                    };
      for(i=0; i<5;  i++)
      {
            pintf("String=%s\t",arrp[i]);
            printf("Address of String=%u\n", arrp[i]);
             printf("Address of String is strored at =%u\n", arrp+i);
}
```

Output:

String=white	Address of String :100	Address of String stored at=2000
String=red	Address of String :106	Address of String stored at=2010
String=green	Address of String:110	Address of String stored at=2020
String=yellow	Address of String :116	Address of String stored at=2030
String=blue	Address of String :123	Address of String stored at=2040