STRINGS:

String Concepts

String is an *array of characters* that is terminated by \0 (null character). This null character indicates the end of the string. Strings are always enclosed by double quotes (" "). Whereas, character is enclosed by single quotes.

Or

In „C" language the group of characters, digits, and symbols enclosed within double quotation (" ") marks are called as string otherwise a string is an array of characters and terminated by NULL character which is denoted by the escape sequence „\0".

C Strings

Declaration of String: C does not support string as a data type. However, it allows us to represent strings as character arrays. In C, a string variable is any valid C variable name and it is always declared as an array of characters.

Syntax: char string_name[size];

The size determines the number of characters in the string name.

Note: In declaration of string size must be required to mention otherwise it gives an error.

Ex: char str[]; // Invalid

char str[0]; // Invalid

char str[-1]; // Invalid

char str[12.63];//Invalid

char str[20]; // Valid

char a[9]; //Valid

Memory allocation

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]

Hear a Variable memory allocated 9 bytes.

Here, the string variable a can hold maximum of 9 characters including NULL (\0) character.

Prasant Dash
7008191907

Declaring and Initializing a string variables

Syntax : char string_name[size]={"string" };

char ch[6]={"USBM"};

U	S	B	M	\0	\0

Note: In Initialization of the string if the specific number of character is not initialized it then rest of all character will be initialized with NULL.

char ch[10]={'U' , 'S' , 'B' , 'M' , ' ' ,'B' ,'B' ,'S' ,'R'};

U	S	B	M		B	B	S	R	\0

Char A[2]={"USBM"} //invalid

Different ways of initialization can be done in various ways:

1 : Initilizing locations character by character.
2 : Partial array initialization.
3 : Intilization without size.
4 : Array initialization with a string .

1. Initilizing locations character by character

Consider the following declaration with initialization,

Char b[9]={'C','O','M','P','U','T','E','R'};

The compiler allocates 9 memory locations ranging from 0 to 8 and these locations are initialized with the characters in the order specified. The remaining locations are automatically initialized to null characters.

Prasant Dash
7008191907

C	O	M	P	U	T	E	R	\0	\0

2 : Partial Array Initilization :

If the characters to be initialized are less than the size of the array, then the characters are stored sequentially from left to right. The remaining locations will be initialized to NULL characters automatically.

Ex : Consider the partial initialization

int a[10]={'R','A','M','A' };

The compiler allocates 10 bytes for the variable a ranging from 0 to 9 and initializes first four locations with the ASCII characters of 'R', 'A', 'M','A'.The remaining locations are automatically filled with NULL characters (i.e,\0).

R	A	M	A	\0	\0	\0	\0	\0	\0

3 : Initilization without size :

Cconsider the declaration along with the initialization

char b[]={'C','O','M','P','U','T','E','R'};

In this declaration, The compiler will set the array size to the total number of initial values i.e 8. The character will be stored in these memory locations in the order specified.

4) Array Initilization with a String :

Consider the declaration with string initialization.

Prasant Dash
7008191907

char b[ ] = “COMPUTER”;

Here, the string length is 8 bytes. But , string size is 9 bytes. So the compiler reserves 8+1 memory locations and these locations are initialized with the characters in the order specified. The string is terminated by \0 by the compiler.

Reading and Writing Strings : The „%s“ control string can be used in scanf() statement to read a string from the terminal and the same may be used to write string to the terminal in printf() statement.

Input function scanf() can be used with %s format specifies to read a string input from the terminal. But there is one problem with scanf() function, it terminates its input on first white space it encounters. Therefore if you try to read an input string "Hello World" using scanf() function, it will only read Hello and terminate after encountering white spaces.

However, C supports a format specification known as the edit set conversion code %[..] that can be used to read a line containing a variety of characters, including white spaces.

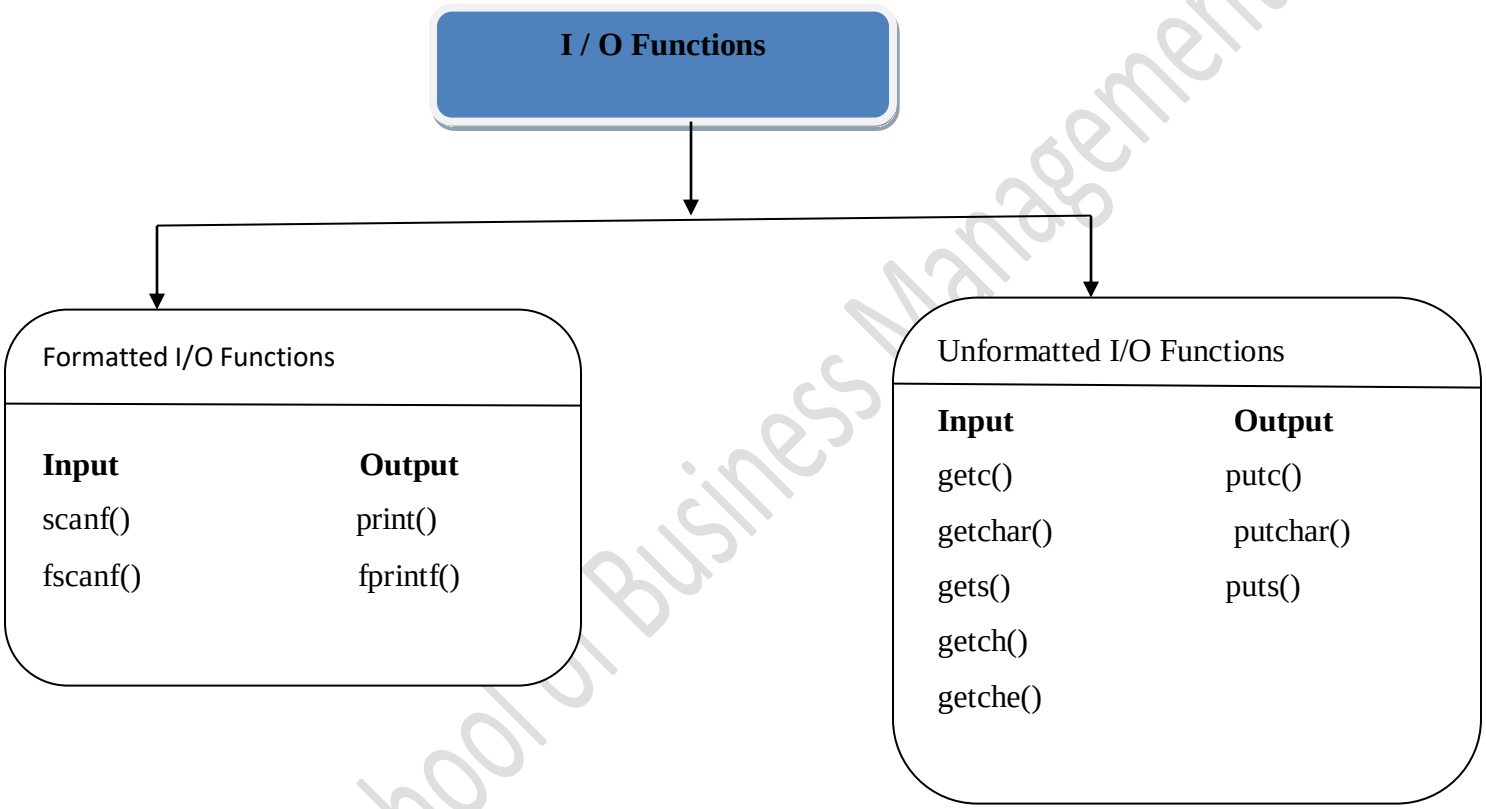
```
#include<stdio.h>
#include<string.h>
void main()
{
 char str[20];
printf("Enter a string");
 scanf("%[^\n]",&str);
 printf("%s",str);
 getch();
}
```


Prasant Dash
7008191907

String Input/output Functions

The strings can be read from the keyboard and can be displayed onto the monitor using various functions.

The various input and output functions that are associated with can be classified as



Unformated I/O Functions

1 : getchar() function : A single character can be given to the computer using „C" input library function getchar().

Syntax : char variable=getchar();

The getchar() function is written in standared I/O library. It reads a single character from a standared input device. This function do not require any arguments, through a pair of parantheses, must follow the statements getchar().

Prasant Dash
7008191907

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{
char ch;
printf("Enter any character/digit:");
ch=getchar();
if(isalpha(ch)>0)
printf("it is a alphabet:%c\n",ch);
else if(isdigit(ch)>0)
printf("it is a digit:%c\n",ch);
else
printf("it is a alphanumeric:%c\n",ch);
        getch();

        }
```

OUTPUT : Enter any character/Digit : abc
it is a alphabet: a

2 : putchar() function : The putchar() function is used to display one character at a time on the standared output device. This function does the reverse operation of the single character input function.

Syntax : putchar(character varaiable);

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{
char ch;
printf("Enter any alphabet either in lower or uppercase:");
ch=getchar();
```


Prasant Dash
7008191907

```
if(islower(ch))
putchar(toupper(ch));
else
putchar(tolower(ch));
getch();
}
```

OUTPUT : Enter any alphabet either in lower or uppercase :a

A

3 : gets() : The gets() function is used to read the string (String is a group of characters) from the standard input device (keyboard).

Syntax : gets(char type of array variable);

Ex :

```
#include<stdio.h>
#include<conio.h>
void main()
{
char str[40];
clrscr();
printf("Enter String name:");
gets(str);
printf("Print the string name%s:",str);
getch();
}
```

OUTPUT : Enter the string : reddy

Print the string: reddy

4 : puts() : The puts() function is used to display the string to the standared output device (Monitor).

Syntax : puts(char type of array variable);

Program using gets() function and puts() function.

Prasant Dash
7008191907

```
#include<stdio.h>
#include<conio.h>
void main()
{
char str[40];

puts("Enter String name:");
gets(str);
puts("Print the string name:");
puts(str);
getch();
}
```

OUTPUT : Enter string name :

subbareddy

Print the string name

subbareddy

getch() function :The getch function reads a single character directly from the keyboard, without echoing to the screen.

Syntax : int getch();

```
Ex : #include<stdio.h>
void main()
{
char c;
c=getch();
}
```

getche() function : The getche() function reads a single character from the keyboard and echoes it to the current text window.

Syntax : int getche();

```
Ex : #include<stdio.h>
```


Prasant Dash
7008191907

```
void main()
        {
char c;
c=getche();
}
```

getc() function : This function is used to accept a single character from the standared input to a character variable.

Syntax : character variable=getc();

putc() function : This function is used to display a single character in a character variable to standared output device.

Syntax : putc(character variable);

String Manipulation Functions/ String Handling Functions

The various string handling functions that are supported in C languages are as shown

String Function	Description
strlen(str)	Returns the length of the string str.
strcpy(str1,str2)	Copies the string str2 to string str1
strcat(str1,str2)	Append string str2 to string str1.
strlwr(str)	Converts the string str to lowercase
strupr(str)	Converts the string str to uppercase.
strrev(str)	Reverse the string str.
strcmp(str1,str2)	Compare two strings str1 and str2.

Prasant Dash
7008191907

All these functions are defined in **string.h header file.**

1 : strlen(string) – String Length : This function is used to count and return the number of characters present in a string.

Syntax : var=strlen(string);

Ex : Progrm using strlen() function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char name[]="JBREC";
int len1,len2;

len1=strlen(name);
len2=strlen("JBRECECE");
printf("The string length of %s is: %d\n",name,len1);
printf("The string length of %s is: %d","JBRECECE",len2);
getch();
}
```

OUTPUT:
The string length of JBREC is : 5
The string length of JBRECECE is :8

2 : strcpy(string1,string2) – String Copy : This function is used to copy the contents of one string to another string.

Syntax : strcpy(string1,string2);

Where

string1 : is the destination string.

Prasant Dash
7008191907

string 2: is the source string.

i.e the contents of string2 is assigned to the contents of string1.

Ex : Progrm using strcpy() function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
        {
char str1[]="REDDY";
char str2[10];
strcpy(str2,str1);
printf("The string1 is :%s\n",str1);
printf("The string2 is :%s\n",str2);
strcpy(str2,str1+1);
printf("The string1 is :%s\n",str1);
printf("The string2 is :%s",str2);
}
```

OUTPUT :

The string1 is : REDDY

The string2 is : REDDY

The string1 is : REDDY

The string2 is : EDDY

//Write a program to copy contents of one string to another string.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char str1[10],str2[20];
int index;
```


Prasant Dash
7008191907

```
printf("Enter the string\n");
scanf("%s",str1);
        for(index=0;str1[index]!='\0';index++)
str2[index]=str1[index];
str2[index]='\0';
printf("String1 is :%s\n",str1);
printf("String2 is :%s\n",str2);
getch();
}
```

OUTPUT :

Enter the string : cprogramming

String1 is : cprogramming

String2 is : cprogramming

3 : strlwr(string) – String LowerCase : This function is used to converts upper case letters of the string in to lower case letters.

Syntax : strlwr(string);

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[]="JBREC";
clrscr();
strlwr(str);
printf("The lowercase is :%s\n",str);
getch();
}
```


Prasant Dash
7008191907

OUTPUT : The lowercase is : jbrec

Write a program to which converts given string in to lowercase.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char str[10];
int index;
printf("Enter the string:");
scanf("%s",str);
for(index=0;str[index]!='\0';index++)
{
if(str[index]>='A' && str[index]<='Z')
str[index]=str[index]+32;
}
printf("After conversionis :%s",str);
getch();
}
```

OUTPUT : Enter the string : SUBBAREDDY

After conversion string is :subbareddy

4 : strupr(string) – String UpperCase : This function is used to converts lower case letters of the string in to upper case letters.

Syntax : strupr(string);

Program using strupr() function.

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


Prasant Dash
7008191907

```
void main()
{
char str[]="jbrec";
strupr(str);
printf("UpperCase is :%s\n",str);
getch();
}
```

OUTPUT : UpperCase is : JBREC

Write a program to which converts given string in to uppercase.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char str[10];
int index;
printf("Enter the string:");
scanf("%s",str);
for(index=0;str[index]!='\0';index++)
{
if(str[index]>='a' && str[index]<='z')
str[index]=str[index]-32;
}
printf("After conversionis :%s",str);
getch();
}
```

OUTPUT : Enter the string : subbareddy

After conversion string is :SUBBAREDDY

Prasant Dash
7008191907

5 : strcmp(string1,string2) – String Comparisons : This function is used to compares two strings to find out whether they are same or different. If two strings are compared character by character until the end of one of the string is reached. If the two strings are same strcmp() returns a value zero. If they are not equal, it returns the numeric difference between the first non-matching characters.

Syntax : strcmp(string1,string2);

Program using strcmp() function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str1[]="reddy";
char str2[]="reddy";
int i,j,k;
i=strcmp(str1,str2);
j=strcmp(str1,"subba");
k=strcmp(str2,"Subba");
printf("%5d%5d%5d\n",i,j,k);
}
```

OUTPUT : 0 -1 32

6: strcat(string1,string2) – String Concatenation : This function is used to concatenate or combine, two strings together and forms a new concatenated string.

Syntax : strcat(sting1,string2);

Where

string1 : is the firdt string1.

Prasant Dash
7008191907

string2 : is the second string2

when the above function is executed, string2 is combined with string1 and it removes the null character (\0) of string1 and places string2 from there.

Program using strcat() function.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str1[10]="jbrec";
char str2[]="ece";
strcat(str1,str2);
printf("%s\n",str1);
printf("%s\n",str2);
getch();
}
```

OUTPUT : jbrecece

Ece

7 : strrev(string) - String Reverse :This function is used to reverse a string. This function takes only one argument and returns one argument.

Syntax : strrev(string);

Ex : Program using strrev() function

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


Prasant Dash
7008191907

```
void main()
        {
char str[20];
printf("Enter the string:");
scanf("%s",str);
printf("The string reversed is:%s",strrev(str));
getch();
}
OUTPUT : Enter the string :subbareddy
        The string reversed is : ydderabbus
```

String program

Q1. **.W.A.P Enter a string print lower case, upper case, length, concated, reverse using library function**

```
//Convert Given String into Uppercase Using Library Function
#include<stdio.h>
#include<string.h>
void main()
{
char string[100],string2[100];
int len;

printf("Enter String : ");
gets(string);
strupr(string);
printf("String after strupr : %s\n", string);

strlwr(string);
```


Prasant Dash
7008191907

```
printf("String after strlwr : %s\n", string);

strcpy(string2,string);

printf("\nCopied String : %s",string2);

len = strlen(string);

printf("\nLength of Given String : %d",len);
printf("\n String reverse : %s\n", strrev(string));

}
```

Q2.**Program to Concat Two Strings with using Library Function: Strcat**

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str1[100];
char str2[100];
char str3[100];
int len;

printf("\nEnter the String 1 : ");
gets(str1);

printf("\nEnter the String 2 : ");
gets(str2);

strcpy(str3,str1);
strcat(str3,str2);
```


Prasant Dash
7008191907

```
printf("\nConcated String : %s",str3);
getch();
}
```

Q3.Program to sort set of strings in alphabetical order

```
#include<stdio.h>
#include<string.h>
void main()
{
char s[5][20],t[20];
int i,j;

printf("Enter any five strings : \n");
for(i=0;i<5;i++)
   scanf("%s",s[i]);

for(i=1;i<5;i++)
   {
   for(j=1;j<5;j++)
      {
      if(strcmp(s[j-1],s[j])>0)
         {
         strcpy(t,s[j-1]);
         strcpy(s[j-1],s[j]);
         strcpy(s[j],t);
         }
      }
   }

printf("Strings in order are : ");
```


Prasant Dash
7008191907

```
for(i=0;i<5;i++)
    printf("\n%s",s[i]);


}
```

Q4. **Program to Find Length of String without using Library Function**

Q5. **Convert a String is Uppercase to lower and lower to Upper, Not without using Library function**

Q6. **W.A.P to find the frequency of a character in a string**

Q7.**W.A.P to print the biggest word in a string**

Q8. W.A.P to check a string is palindrome or not without using library functions

Prasant Dash
7008191907