1 Explain about call by value and call by reference with suitable examples.

Sol : The 2 types of parameters passing mechanisms are :

[1] Call by value

[2] Call by reference.

Example :- Swapping of 2 numbers in 2 different techniques.

```
// Call by value
#include<stdio.h>
void swap(int,int)
{
  int a,b;
  scanf("%d %d",&a,&b);
  printf("Before swapping in main a=%d b=%d",a,b);
  swap(a,b);
  printf("After swapping in main a=%d b=%d",a,b);
}
```

```
void swap(int a, int b)
{
  int temp;
  temp=a;
  a=b;
  b=temp;
  printf("After swapping in function a=%d b=%d",a,b);
}
```

* In main the value of arguments are not changed.

output

Before swapping in main a=10 b=20;

After swapping in function a=20 b=10;

After swapping in main a=10 b=20;

Sol : The 2 types of parameters passing mechanisms are :

[1] Call by value

[2] Call by reference.

Example :- Swapping of 2 numbers in 2 different techniques.

```
// Call by value
#include<stdio.h>
void swap(int,int)
{
  int a,b;
  scanf("%d %d",&a,&b);
  printf("Before swapping in main a=%d b=%d",a,b);
  swap(a,b);
  printf("After swapping in main a=%d b=%d",a,b);
}
```

```
void swap(int a, int b)
{
  int temp;
  temp=a;
  a=b;
  b=temp;
  printf("After swapping in function a=%d b=%d",a,b);
}
```

* In main the value of arguments are not changed.

output

Before swapping in main a=10 b=20;

After swapping in function a=20 b=10;

After swapping in main a=10 b=20;

2 Write a C programme for Multiplication of two matrixes.

[9] C program for multiplication of 2 matrices in C.

Sol: Code

```
#include<stdio.h>
int main
{
int a[10][10], b[10][10], mul[10][10], r, c, i, j, K;
printf ("enter number of row = "),
scanf ("%d", &r);
printf ("enter number of column = ");
scanf ("%d", & c);
printf ("enter 1st matrix element = \n");
for(i=0; i<r; i++)
{
for(j=0; j<c; j++)
{
scanf ("%d", & a[i][j])
}
}
printf ("enter 2nd matrix elements = \n");
for(i=0; i<r; i++)
{
for(j=0; j<c; j++)
{
scanf ("%d", & b[i][j]);
}
}
printf ("multiply of matrix = \n);
for(i=0; i<r; i++)
{
mul [i][j] =0
for(j=0; j<c; j++)
{
for (K=0; K<c; K++)
{
mul [i][j] += a[i][K] * b[K][j];
for (i=0; i<r; i++)
{for (j=0; j<c; j++))
{
printf ("%d\t", mul[i][j]);
}printf ("\n");
}
return 0;
}
```

3 Write a C programme to implement Fibonacci series using recursion.

ans) //fibonacci series using recursions

```
#include <stdio.h>
int fib(int n);
int main()
{
    int n,i;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("%d",fib(i));
    }
    return 0;
}
int fib(int n)
{
    int r;
    if(n==0)
    return 0;
    else if(n==1)
    return 1;
    else
    return r=fib(n-1)+fib(n-2);
}
```

4 Explain about String handling functions?

Ans)

Ans: (1) strlen() function:

This function is used to count the number of character in a string.

ex: strlen(program) = 7

(2) strcmp() function:

This function compares two strings character by character.

Return Value from strcmp()

Return Value	Remarks
0	if strings are equal.
>0	if first non-matching character in str1 is greater (in ASCII) than that of str2.
<0	if first non-matching character in str1 is lesser (in ASCII) than that of str2.

ex: str1[] = "abcd"

str2[] = "abCd"

strcmp(str1, str2) = 1

(3) strcat() function:

This function joins two strings. The strcat() function joins the destination string and source string and the result is stored in destination string.

ex: char str1[100] = "This is", str2[] = "a program";

strcat(str1, str2);

printf("%s", str1);

printf("%s", str2);

Output:

This is a program

a program.

[4] strcpy() function

- The strcpy() function copies the string pointed by source (including null character) to the destination.
- The strcpy() function also returns copied string.

example: char str1[20] = "C programming", str2[20];

strcpy(str2, str1);

printf("%s", str2);

Output: C programming.

[5] strlwr() function

- This function converts all upper case alphabets to lower case alphabets

[6] strupr() function

- This function converts all lower case alphabets to upper case alphabets

example: str1[20] = "LOWER"

strlwr(str1) = lower

str2[20] = "upper"

strupr(str2) = UPPER

5 Write a C programme to sort the given set of strings
.ans)

```
g in alphabetical order.
ans: #include <stdio.h>
#include <string.h>
int main()
{
  char string1[20];
  int i,j,n;
  char temp;
  scanf("%s", string1);
  n= strlen(string1);
  for(i=0; i<n-1 ; i++)
  {
    for(j=i+1; j<n; j++)
    {
      if (string1[i] >string1[j])
      {
        temp= string1[i];
        string1[i] = string1[j];
        string1[j] = temp;
      }
    }
  }
  printf("after sorting string is %s",string1);

  return 0;
```

6 What do you mean by a function? Give the structure of the user defined function and explain about the arguments and return values.
ans)
A function is a named, independent section of c code that performs a specific task and optionally returns a value to the calling program.
Structure of user defined functions-
It consists of 3 parts
1)Function prototype
<return type><function type>(<datatype1…..<datatype n>);
2)Function calling
function name(variable1…variable n)
3)Function definition
<returntype><function name>(<datatype1>....<datatype n>) [no semicolon]

Arguments-
Arguments are the variables with its datatype which are written within ‘< >’.
Arguments may or may-not be included within the functions.

Return type-
It returns the datatype in which the called function should return the value.

7 Write a programme to read, calculate average and
print student marks using an array of structures.
Ans)

```
#include<stdio.h>
struct student
{
        char stname[10];
        int m[10],n,i,j,t;
}s[10];
int main()
{
   int average,total,i,j,n,t,m[10];
        printf("enter no of students");
        scanf("%d",&t);
        printf("enter no of subjects");
        scanf("%d",&n);
        for(i=0;i<t;i++)
        {
                scanf("%s",s[i].stname);
                for(j=0;j<n;j++)
                {
                        scanf("%d",&s[i].m[j]);
                }
        }
        for(i=0;i<t;i++)
        {
                total=0;
                printf("%s\n",s[i].stname);
                for(j=0;j<n;j++)
                {
                   total=total+s[i].m[j];
                        printf("%d\t",s[i].m[j]);
                }
                printf("%d\n",average=(total)/n);
        }
        return 0;
```

8 Differentiate between self-referential structure and nested structure with example.

Ans). In C-programming a self-referential structure is a structure that contains a pointer to an instance of same structure. It is used to create linked data structures such as linked lists and trees

Eg: struct node
{
int data;
struct node *next;
};

In this example, the 'node' structure contains an integer "data" and a pointer "next" to another instance of node. This allows us to create a linked list where each node points to next node in list.

- A nested structure is a structure that contains another structure as a member. It is used to group related data together and to create more complex data structures.

For example;

```
struct address
{
char street[20], city[20], state[20];
};
struct employee {
int id;
char name[20];
struct address addr;
};
```

9 Explain three dynamic memory allocation functions with suitable examples.

Ans: In C-programming dynamic memory allocation refers to process of allocating memory at runtime, as opposed to compiler-time.

(1) malloc(): This function is used to allocate a block of memory of specified size. It takes one argument which is the size of memory block in bytes. It returns a pointer to the first byte of allocated memory block. If the memory allocation is successful, the pointer returned malloc() points to the first byte of allocated memory bloc otherwise it returns a null pointer.

(2) calloc(): This function is used to allocate a block of memory for an array of specified number of elements, each of specified size. It takes 2 arguments the first argument is the number of elements in array and the second argument is the size of each element in bytes. It returns a pointer to the first byte of allocated memory block. If memory allocation is successful the pointer returned by calloc points to first byte of allocated memory block otherwise a nullpointer.

10) Explain about storage classes.

In C Programming a storage is very easy to specify the duration and visibility of variable (or) function.

There are 4 storage classes in C.

1) Automatic : These are local variables that are defined inside a function. They are also called local variables or automatic variables. They are automatically created and destroyed when function is called. They don't retain their value between function calls.

Example :

```
void func()
{
 int x=5;
 printf("%d", x);
}
```

2) Register : These are local variables that are stored in a register instead of memory. Using a register storage class can improve the performance of the program by reducing memory access time. However, the no of register is limited, so not all variables can be stored.

Example

```
void func()
{
  register int x;
  x=0;
  printf("%d", x);
}
```

3) Static : These are variables that retain their value b/w function calls. A variable defined as a static inside a function maintains its value between function c

```
void func()
{
static int x=0;
x++;
printf("%d", x);
}
```

4) Extern : These are variables that are defined in one file and can be accessed in another file.

An extern variable can be defined in one source file and used in another source file.

Example :

```
// file 1.c
int x;
x=5;
// file 2.c
extern int x;
printf("%d", x);
```

11) Develop a programme to create a library catalog with the following members: access number, authors name, title of the book, year of publication and book price using structures.

Ans)

```
#include<stdio.h>
struct library
{
        int accessno,yearofpub,bookprice;
        char bookname[10],author[20];
}lib[5];
int main()
{
        int i,n,a;
        printf("enter no of books");
        scanf("%d",&n);
        for(i=0;i<n;i++)
        {
                scanf("%d",&lib[i].accessno);
                scanf("%s",lib[i].author);
                scanf("%s",lib[i].bookname);
                scanf("%d",&lib[i].yearofpub);
                scanf("%d",&lib[i].bookprice);
        }
        printf("enter access number");
        scanf("%d",&a);
        for(i=0;i<n;i++)
        {
                if(i+1==a)
                {
              printf("%s\t",lib[i].author);
                printf("%s\t",lib[i].bookname);
                        printf("%d\t",lib[i].yearofpub);
                        printf("%d\t",lib[i].bookprice);
                }
        }
   return 0;
}
```

13 What is a Pointer? Explain pointer arithmetic
operations with suitable examples.
Ans)
A pointer is a variable that stores the memory address of another variable. Pointers are useful to DMA and function processing and many.
Pointer arithmetic is the manipulation of pointers to perform various operations like subtraction, addition, increment and many.
‘
In c, pointer arithmetic is performed in following ways-
Ptr++’ or Ptr-- -it increments or decrements the pointer to the element of the same type.
‘Ptr+n’ or Ptr-n -it adds or removes the pointer to the point moving it n elements of the same type.

15 Write a program to demonstrate read and write operations on a file.
Ans)

```
#include<stdio.h>
Int main()
{
FILE*fp;
fp=fopen(“example.txt”,:w”);
fprint(fp,”writing to a file in c”);
fclose(fp);
fp=fopen(“example.txt’,”r”);
Char ch;
while((ch=fgetc(fp))!=EOF)
{
      printf(“%c”,ch);
}
fclose(fp);
return 0;
}
```