Assignment-4 (Functions): -

- Write a program with a function declaration, definition and call (eg:- sum)
- Without prototype for a function give different type of, different no.of arguments to a function and test the behavior
- Create a local & global variable of same name and test the value
- Write a program to find how many times a function is being called (use local static variable as count)
- Try register storage class for local, global variables. Can we get address of register variable
- Try some nested calls

  sqrt(pow(2,abs(x))), putchar(toupper(ch)) etc
- Test linking of a extern variable & global variable within single program
- Write a function to swap two variables using Pass by value, Pass by reference
- Write a single function to return sum, product of two no.s
- Recursion programs
  - factorial, sum of n no.s,
  - gcd, fibonacci series,
  - No. format conversions(decimal,binary and octal)
  - count no.of 1s in a binary code
- Whats wrong in this code, any fixes to the problem?

```
int* test(int x)
 {
    int y=x*x;
    return &y;
 }
```
- Try conversions between int*, const int* while passing parameters to functions

```
int *p;      const int *q;
test(p);     void test(const int* );
test(q);     void test2(int *);
```
- Passing 1D, 2D arrays to a function

  - sum,min,max of array elements
  - Matrix operations
- Can you retun arrays from a function
- Create multifile program

  main.c – calling sum, square function

  sum.c - sum definition sqr.c – square definition

  compile each file separately and link them (preferably use Makefile)

  Try extern, static linkage specifiers for global variables, functions
- Function Pointers
  - Write a simple program to test function pointer
  - Menu driven programs without if,else,switch(array of function pointers)
  - Passing function names as parameters

```
void test(int x, int y, int (*fp) (int,int))
{
    int z = fp(x,y);
    ----
}
test(10,20,sum);
```

- typedef for function pointer

```
typedef int (*pftype)();
pftype pf1;
pf1=sum; pf1(10,20);
```

Some more on arrays & pointers:-

- Array of pointers

  int *p[5];   p[0]=&x;   p[1]=&y; etc.

  sizeof(p), sizeof(*p),sizeof(**p);

- Pointer to whole 1D array

  int a[5];

  int (*pa)[5]; p=&a;   sizeof(p), sizeof(*p) etc.

  int b[3][5];

  pa=b; Significance of pa+1, pa+2, *pa+j, *(*pa+j)

- Pointer to 2D array

  int arr[2][3];

  int (*pb)[2][3];

  pb=&arr;

  sizeof(pb), sizeof(*pb), sizeof(**pb)

  Values of pb, pb+1, *pb, *pb+1