PRACTICAL-1

Aim-WAP in function with usage of an argument and no return value for print a simple message.

Input:-

```
#include<stdio.h>

void checkmessage();
int main()
{
   checkmessage(); //argument is not passed
   return 0;
}
//return type is void meanig doesn't return any value
void checkmessage()
{
   printf("\nWhere the world builds software\n");
}
```

Output:-

```
Where the world builds software
```

PRACTICAL-2

Aim-WAP to find out the average of 5 number by use of User Defined Functions.

Input:-

```
#include<stdio.h>

float average(int,int,int,int,int); //to get average
int main()
{
   int num_1,num_2,num_3,num_4,num_5;
   float avg;
   printf("enter the first integer number :");
   scanf("%d",&num_1);
   printf("enter the second integer number:");
   scanf("%d",&num_2);
   printf("enter the third integer number:");
   scanf("%d",&num_3);
   printf("enter the forth integer number:");
   scanf("%d",&num_4);
   printf("enter the fifth integer number:");
   scanf("%d",&num_5);
   avg=average(num_1,num_2,num_3,num_4,num_5);
   printf("average:%f\n",avg);
   return 0;
}
float average(int a,int b,int c,int d ,int e)
{
   float average;
   return  ((float)(a)+(float)(b)+(float)(c)+(float)(d)+(float)(e))/5;
}
```

Output:-

```
enter the first integer number :21
enter the second integer number:32
enter the third integer number:45
enter the forth integer number:34
enter the fifth integer number:12
average:28.799999
```

PRACTICAL-3

Aim-WAP to using UDF of type with Argument and No Return Values to calculate modulo of values entered by user.

Input:-

```
#include <stdio.h>

int display(int x,int y);
int main()
{
   int x,y;
   printf("enter the value of x:");
   scanf("%d",&x);
   printf("enter the value of y:");
   scanf("%d",&y);
   display(x,y);
   printf("modulo is:%d",display(x,y));
   return 0;
}
int display(int x,int y)
{
   double m;
   m=x%y;
   return m;
}
```

Output:-

```
enter the value of x:100
enter the value of y:12
modulo is:4
```

PRACTICAL-4

Aim-WAP for User Define Function type No Arguments Passed and No Return Value to find out the weather number enter by user is Prime or not

Input:-

```
#include <stdio.h>

void checkPrimeNumber();
//no arguments and no return value
int main()
{
   checkPrimeNumber();    // argument is not passed
   return 0;
}

// return type is void meaning doesn't return any value
void checkPrimeNumber()
{
   int n, i, a = 0;

   printf("Enter a  integer: ");
   scanf("%d",&n);

   for(i=2; i <= n/2; ++i)
   {
      if(n%i == 0)
      {
         a = 1;
      }
   }
   if (a == 1)
      printf("%d is not a prime number.", n);
   else
      printf("%d is a prime number.", n);
}
```

Output:-

```
Enter a  integer: 5
5 is a prime number.
```

PRACTICAL-4a

Aim-WAP for User Define Function type No arguments passed but a return Value to find out the weather number enter by user is Prime or not

Input:-

```
#include <stdio.h>
int Integer();

int main()
{
   int n, i, a = 0;
  // no argument is passed
   n = Integer();
   for(i=2; i<=n/2; ++i)
   {
      if(n%i==0){
         a = 1;
         break;
      }
   }
   if (a == 1)
      printf("%d is not a prime number.", n);
   else
      printf("%d is a prime number.", n);
   return 0;
}
// returns integer entered by the user
int Integer()
{
   int n;
   printf("Enter a  integer: ");
   scanf("%d",&n);
   return n;
}
```

Output:-

```
Enter a  integer: 5
5 is a prime number.
```

PRACTICAL-4b

Aim-WAP for User Define Function Type Argument passed but no return Value to find out the weather number enter by user is Prime or not

Input:-

```
#include <stdio.h>
void checkPrime(int n);

int main()
{
   int n;
   printf("Enter a positive integer: ");
   scanf("%d",&n);
   // n is passed to the function
   checkPrime(n);
   return 0;
}
// return type is void meaning doesn't return any value
void checkPrime(int n)
{
   int i, a = 0;
   for(i=2; i <= n/2; ++i)
   {
      if(n%i == 0){
         a = 1;
         break;
      }
   }
   if(a == 1)
      printf("%d is not a prime number.",n);
   else
      printf("%d is a prime number.", n);
}
```

Output:-

```
Enter a  integer: 5
5 is a prime number.
```

PRACTICAL-4c

Aim-WAP for User Define Function Type Argument passed and a return value to find out the weather number enter by user is Prime or not

Input:-

```
#include <stdio.h>
int checkPrime(int n);

int main()
{
   int n, a;
   printf("Enter a positive integer: ");
   scanf("%d",&n);
   // n is passed to the checkPrimeNumber() function
   // the returned value is assigned to the flag variable
   a = checkPrime(n);
   if(a == 1)
      printf("%d is not a prime number",n);
   else
      printf("%d is a prime number",n);
   return 0;
}
// int is returned from the function
int checkPrime(int n)
{
   int i;
   for(i=2; i <= n/2; ++i)
   {
      if(n%i == 0)
         return 1;
   }
   return 0;
}
```

Output:-

```
Enter a  integer: 5
5 is a prime number.
```

PRACTICAL-5

Aim-WAP to find square of given number using function with the use of function with an argument and a return value.

Input:-

```
#include<stdio.h>

float square(float n);

int main()
{
   float number, s;
   printf("Enter any integer Value : ");
   scanf("%f", &number);
   s = square(number);
   printf("square of  number %.2f is  =  %.2f", number, s);
   return 0;
}
float square(float n)
{
   return (n*n);
}
```

Output:-

```
Enter any integer Value : 4.5
square of  number 4.50 is  =  20.25
```

PRACTICAL-6

Aim-WAP to print the Fibonacci Series up to n terms using function. The value of n should be given by user.

Input:-

```
#include<stdio.h>

void fibonacci(int);

int main()
{
   int n;
   printf("Enter n terms: ");
   scanf("%d", &n);
   printf("The fibonacci series is: \n");
   fibonacci(n);
   return 0;
}
void fibonacci(int n)
{
   int a=0, b=1, c=0,i;
   for (i=0;i<n;i++)
   {
     printf("%d\t", c);
     a = b;
     b = c;
     c = a+b;
   }
}
```

Output:-

```
Enter n terms: 10
The fibonacci series is:
0	1	1	2	3	5	8	13	21	34
```

PRACTICAL-7

Aim-WAP to find out the factorial of a given number by using a recursive function

Input:-

```
#include<stdio.h>

long int  number(int n);

int main ()
{
   int n;
   printf("enter the integer:");
   scanf("%d",&n);
   printf("Factorial of %d = %ld",n,number(n));
   return 0;
}
long int  number(int n)
{
   if (n>=1)
   {
      return n*number(n-1);
   }
   else
   {
      return 1;
   }

}
```

Output:-

```
enter the integer:7
Factorial of 7 = 5040
```

PRACTICAL-7a

Aim-WAP to Print Natural Numbers using Recursion

Input:-

```
#include<stdio.h>

void natural(int);

int main ()
{
    int l;
    printf("enter the number of terms:");
    scanf("%d",&l);
    printf("\natural number from 1 to %d are:",l);
    natural(l);
    return 0;
}
void natural(int num)
{
    if (num)
    {
        natural(num-1);
    }
    else
    {
        return;
    }
    printf("\n%d\n",num);
}
```

Output:-

```
enter the number of terms:14
natural number from 1 to 14 are:
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

PRACTICAL-7b

Aim-Write a program in C to calculate the sum of numbers from 1 to n using recursion.

Input:-

```
#include<stdio.h>

int number(int n);

int main ()
{
   int num ;
   printf("enter the integer:");
   scanf("%d",&num);
   printf("sum=%d", number(num));
   return 0;
}
int number(int n)
{
   if (n!=0)
   {
      return n+number(n-1);
   }
   else
   {
      return n;
   }
}
```

Output:-

```
enter the integer:5
sum=15
```

PRACTICAL-8

Aim-WAP to print the Fibonacci Series up to n terms using recursion. The value of n should be given by user.

Input:-

```
#include<stdio.h>

int f(int n);

int main()
{
    int n,sum=0;
    printf("enter the value of n:");
    scanf("%d",&n);
    printf("fibonacci series:");
    while (f(sum) <= n)
    {
        printf("%d ",f(sum));
        sum++;
    }
    return 0;
}
int f(int n)
{
    if (n<=1)
    {
        return n;
    }
    return f(n-1)+f(n-2);
}
```

Output:-

```
enter the value of n:100
fibonacci series : 0 1 1 2 3 5 8 13 21 34 55 89
```

PRACTICAL-9

Aim-WAP to find the GCD of two given integers by using the recursive function

Input:-

```
#include<stdio.h>

int hcf(int n_1,int n_2);
int main()
{
    int n_1,n_2;
    printf("enter the value of n_1:");
    scanf("%d",&n_1);
    printf("enter the value of n_2:");
    scanf("%d",&n_2);
    printf("G.C.D %d and %d is %d \n",n_1,n_2,hcf(n_1,n_2));
    return 0;
}
int hcf(int n_1,int n_2){
    if (n_2 != 0)
    {
        return hcf(n_2,n_1%n_2);
    }
    else{
        return n_1;
    }
}
```

Output:-

```
enter the value of n_1:6664
enter the value of n_2:40
G.C.D 6664 and 40 is 8
```

PRACTICAL-10

Aim-WAP to check a given number is even or odd using the function

Input:-

```
#include<stdio.h>

double oddeven(int n_1);

int main ()
{
   int n_1;
   printf("\n enter the value :");
   scanf("%d",&n_1);
   if (oddeven(n_1))
   {
      printf("number is odd\n");
   }
   else{
      printf("number is even\n");
   }
   return 0;
}
double oddeven(int n_1)
{
   return (n_1 & 1);
}
```

Output:-

```
 enter the value
:93895721347957417068374896786784375683479851764398583147681347684607813
4687438674876583476598746845671345634765983465986495734985734875
number is odd
```

PRACTICAL-11

Aim-Write a C programming to find out maximum and minimum of some values using function which will return an array.

Input:-

```
#include<stdio.h>

int array(int a[],int n);

int main()
{
    int i,n,sum;
    printf("enter the value of array:");
    scanf("%d",&n);
    int a[n];
    printf("enter the value :");
    for ( i = 0; i < n; i++)
    {
        scanf("%d",&a[i]);
    }
    array(a,n);
    return 0;
}

int array(int a[],int n)
{
    int max,min,i;
    min=max=a[0];
    for ( i = 1; i < n; i++)
    {
        if (min>a[i])
        {
            min=a[i];
        }
        if (max<a[i])
        {
            max=a[i];
        }
    }
    printf("minimum of array is :%d",min);
    printf("\nmaximun of array is :%d\n",max);
    return 0;
}
```

Output:-

```
enter the value of array:5
enter the value :1
54
-1
-5
55
minimum of array is :-5
maximun of array is :55
```

PRACTICAL-12

Aim-Write a program in C to check Armstrong and perfect numbers using the function.

Input:-

```
#include<stdio.h>
#include<stdlib.h>
int armstrong(int n);
int perfect(int n);
int main()
{
   int n;
   printf("input any number:");
   scanf("%d",&n);
   if (armstrong(n))
   {
      printf("the %d is an armstrong number \n",n);
   }
   else{
      printf("the %d is not an armstrong number\n",n);
   }
   if (perfect(n))
   {
      printf("the %d is a perfect number\n ",n);
   }
   else{
      printf("the %d is not a perfect number \n",n);
   }
   return 0;
}
int armstrong(int n)
{
   int l,sum,num;
   sum=0;
   num=n;
   while (num!=0)
   {
      l=num%10;
      sum+=l*l*l;
      num=num/10;
   }
   return (n==sum);
}
int perfect(int n)
{
   int i,sum,num;
   sum=0;
   num=n;
   for ( i = 1; i < num; i++)
   {
      if (num%i==0)
      {
            sum+=i;
        }
    }
    return (n==sum);
}
```

Output:-

```
input any number:2421
the 2421 is not an armstrong number
the 2421 is not a perfect number
```

PRACTICAL-13

Aim-WAP to find sum of n elements entered by user. For this program use calloc() function to allocate memory dynamically.

Input:-

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

int main()
{
   int n, i, *ptr, sum = 0;
   printf("Enter number of elements: ");
   scanf("%d", &n);

   ptr = (int*) calloc(n, sizeof(int));
   if(ptr == NULL)
   {
      printf("Error! memory not allocated.");
      exit(0);
   }

   printf("Enter elements: ");
   for(i = 0; i < n; ++i)
   {
      scanf("%d", ptr + i);
      sum += *(ptr + i);
   }

   printf("Sum = %d", sum);
   free(ptr);
   return 0;
}
```

Output:-

```
Enter number of elements: 5
Enter elements: 1
2
3
4
5
Sum = 15
```

PRACTICAL-14

Aim-Write a C program to use function to insert a sub-string in to given main string from a given position.

Input:-

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

int string(char a[1000], char b[1000], int i );

int main ()
{
   char a[1000],b[1000];
   int i;
   printf("\n enter the main string:");
   gets(a);
   printf("\n enter the sub string:");
   gets(b);
   printf("\n enter the string to insert sub string:");
   scanf("%d",&i);
   string(a,b,i);
   return 0;
}
int string(char a[1000],char b[1000],int i)
{
   char temp[1000];
   int m,n,j,k;
   m=strlen(a);
   n=strlen(b);
   for ( j = 0; j < i; j++)
   {
      temp[j]=a[j];
   }
   for ( j = j,k=0; j < n+i ,k<n; j++,k++)
   {
      temp[j]=b[k];
   }
   for ( j = n+i,k=i; j < m ,k<m; j++,k++)
   {
      temp[j]=a[k];
   }
   puts(temp);
   return 0;
}
```

Output:-

```
 enter the main string:hello

 enter the sub string : hi

 enter the string to insert sub string:3
helhilo
```

PRACTICAL-14a

Aim-Write a C program that uses functions to delete n Characters from a given position in a given string.

Input:-

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

int delchar(char *x,int a, int b);

int main()
{
   char string[10];
   int n,pos,p;
   puts("Enter the string");
   gets(string);
   printf("Enter the position from where to delete");
   scanf("%d",&pos);
   printf("Enter the number of characters to be deleted");
   scanf("%d",&n);
   delchar(string, n,pos);
}
int delchar(char *x,int a, int b)
{
   if ((a+b-1) <= strlen(x))
   {
      strcpy(&x[b-1],&x[a+b-1]);
      puts(x);
   }
   return 0;
}
```

Output:-

```
Enter the string
warning: this program uses gets(), which is unsafe.
programming9
Enter the position from where to delete6
Enter the number of characters to be deleted5
progrg9
Abort trap: 6
```

PRACTICAL-15

Aim-WAP to perform the Swapping of two numbers using Function Call by Value and Call by Reference.

Input:-

```
#include <stdio.h>

void swap(int*, int*);

int main()
{
  int x, y;

  printf("Enter the value of x and y\n");
  scanf("%d%d",&x,&y);

  printf("Before Swapping\nx = %d\ny = %d\n", x, y);

  swap(&x, &y);

  printf("After Swapping\nx = %d\ny = %d\n", x, y);

  return 0;
}

void swap(int *a, int *b)
{
  int temp;

  temp = *b;
  *b = *a;
  *a = temp;
}
```

Output:-

```
Enter the value of x and y
21
35
Before Swapping
x = 21
y = 35
After Swapping
x = 35
y = 21
```

PRACTICAL-16

Aim-WAP to demonstrate use of library function for finding the result of power of the number.

Input:-

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

int main()
{
   double b,e,power;
   printf("enter the base :");
   scanf("%lf",&b);
   printf("enter the exponent :");
   scanf("%lf",&e);
   power=pow(b,e);
   printf("%.2lf^%.2lf=%.2lf",b,e,power);
   return 0;
}
```

Output:-

```
enter the base :6
enter the exponent :2
6.00^2.00=36.00
```

PRACTICAL-16a

Aim-WAP to demonstrate use of library function for generating random numbers

Input:-

```
#include <stdio.h>
#include <stdlib.h>
int main() {
  int c, n,max,num;
   printf("Enter the number of random numbers you want\n");
   scanf("%d", &n);
   printf("Enter the maximum value of random number\n");
   scanf("%d", &max);
  printf("random numbers  \n");

  for (c = 1; c <= n; c++) {
    n = rand() % 100 + 1;
    printf("%d\n", n);
  }

  return 0;
}
```

Output:-

```
Enter the number of random numbers you want
100
Enter the maximum value of random number
10
random numbers
8
50
74
59
31
73
45
79
24
10
```

PRACTICAL-16b

Aim-WAP to demonstrate use of library function for square root of a number.

Input:-

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

double findroot(double n);

int main()
{
    int n;
    printf("enter the value of n:");
    scanf("%d",&n);
    printf("%f ", findroot(n));
    return 0;
}

double findroot(double n)
{
    return sqrt(n);
}
```

Output:-

```
enter the value of n:6
2.449490
```

PRACTICAL-17

Aim-WAP to demonstrate local variable, global variable, register and static variable using related keywords.

Input:-

```
#include<stdio.h>

int stc()
{
   static int i=0;
   i++;
   return i;
}
int gba=10;
int main()
{
   printf("static value : %d",stc());
   printf("\t static value : %d",stc());
   printf("\n global value : %d",gba);
   int loc=15;
   printf("\n local value : %d",loc);
   register int reg=20;
   int a=5;
   register int *rgstr=&a;
   printf("%d",*rgstr);
   return 0;
}
```

Output:-

```
static value : 1        static value : 2
 global value : 10
```

PRACTICAL-18

Aim-WAP using function to read values of sides of triangle from user and display its perimeter or area based on user's choice.

Formula for perimeter => p=a+b+c, where a, b and c are lengths of side of a triangle. Formula for area => a = $\sqrt{s(s-a)(s-b)(s-c)}$ **here s = (a+b+c)/2**.

Input:-

```
#include <stdio.h>
#include <stdlib.h>
#include<math.h>

float findArea(float a, float b, float c) ;
float findPerimeter(float a,float b,float c);

float findArea(float a, float b, float c)
{
    // Length of sides must be positive and sum of any two sides
   // must be smaller than third side.
   if (a < 0 || b < 0 || c <0 || (a+b <= c) ||
      a+c <=b || b+c <=a)
   {
      printf("Not a valid trianglen");
      exit(0);
   }
   float s = (a+b+c)/2;
   return sqrt(s*(s-a)*(s-b)*(s-c));
}
float findPerimeter(float a,float b,float c)
{
   return (a+b+c);
}
int main()
{
   float a ,b,c;
   printf("enter the value of a ");
   scanf("%f",&a);
   printf("enter the value of b ");
   scanf("%f",&b);
   printf("enter the value of c ");
   scanf("%f",&c);
   printf("Area is %f", findArea(a, b, c));
   printf("perimeter %f",findPerimeter(a,b,c));
   return 0;
}
```

Output:-

```
enter the value of a3
enter the value of b4
enter the value of c5
Area is 6.000000perimeter 12.000000
```

PRACTICAL-19

Aim-WAP to demonstrate the basic operation of calculator using switch...case and functions.

Input:-

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

float add(float num1, float num2);
float sub(float num1, float num2);
float mult(float num1, float num2);
float div(float num1, float num2);
int main()
{
   char op;
   float num1, num2, result=0.0f;
   printf("Enter [value 1] [operator] [value 2]\n");
   /* Input two number and operator from user */
   scanf("%f %c %f", &num1, &op, &num2);
   switch(op)
   {
      case '+':
         result = add(num1, num2);
         break;
      case '-':
         result = sub(num1, num2);
         break;
      case '*':
         result = mult(num1, num2);
         break;
      case '/':
         result = div(num1, num2);
         break;
      default:
         printf("Invalid operator");
   } /* Print the result */
   printf("%.2f %c %.2f = %.2f", num1, op, num2, result);

   return 0;
}
/*Function to add two numbers*/
float add(float num1, float num2)
{
   return num1 + num2;
}
/*Function to subtract two numbers*/
float sub(float num1, float num2)
{
   return num1 - num2;
}
/*Function to multiply two numbers*/
float mult(float num1, float num2)
```

```
{
    return num1 * num2;
}
/*Function to divide two numbers*/
float div(float num1, float num2)
{
    return num1 / num2;
}
```

Output:-

```
Enter [value 1] [operator] [value 2]
22/2
22.00 / 2.00 = 11.00
```

PRACTICAL-20

Aim-WAP in C to read square matrix of order n, transpose it using user defined function and display transposed matrix from main() function.

Input:-

```
#include <stdio.h>
void transposel();
int main() {
   transposel();
   return 0;
}
void transposel()
{
   int a[100][100], transpose[100][100], r, c, i, j;
   printf("Enter rows and columns: ");
   scanf("%d %d", &r, &c);

   // Assigning elements to the matrix
   printf("\nEnter matrix elements:\n");
   for (i = 0; i < r; ++i)
      for (j = 0; j < c; ++j) {
         printf("Enter element a%d%d: ", i + 1, j + 1);
         scanf("%d", &a[i][j]);
      }

   // Displaying the matrix a[][]
   printf("\nEntered matrix: \n");
   for (i = 0; i < r; ++i)
      for (j = 0; j < c; ++j) {
         printf("%d  ", a[i][j]);
         if (j == c - 1)
            printf("\n");
      }
   for (i = 0; i < r; ++i)
   {
      for (j = 0; j < c; ++j)
      {
         transpose[j][i] = a[i][j];
      }
   }

   // Displaying the transpose of matrix a
   printf("\nTranspose of the matrix:\n");
   for (i = 0; i < c; ++i)
      for (j = 0; j < r; ++j) {
         printf("%d  ", transpose[i][j]);
         if (j == r - 1)
         {
            printf("\n");
         }
      }
}
```

Output:-

```
Enter rows and columns: 3
3

Enter matrix elements:
Enter element a11: 1
Enter element a12: 2
Enter element a13: 3
Enter element a21: 4
Enter element a22: 5
Enter element a23: 6
Enter element a31: 7
Enter element a32: 8
Enter element a33: 9

Entered matrix:
1  2  3
4  5  6
7  8  9

Transpose of the matrix:
1  4  7
2  5  8
3  6  9
```