Function :- A function in C programming is a block of code that performs a specific task. It can be called or used multiple times in a C-program which allows the user to reduce the code size and increase redability.

Advantages of function

- Code Reusability :- function avid repetation by using the same block.
- Reduces Complexity :- It reduces the the complexity as it is divided in smaller tasks.
- Reduces Compile time :- As the code is divided into ~~same~~ smaller task and is called used at many places of same program it reduces the lenght of code.:-
- Reduces code size :- As the ~~code~~ Repeating tasks are done through the function once a function is made it can be called or used with a single line of code. which reduces the size of code.

→ Difference betwn user-defined function and library function.

User-defined function	Library function.
1) It is created by programmer to preform a specific task.	1) Pretented and stored in standard libraries.
2) Must be decleared and difined before use	2) No need to define. Just use include the proper header file
3) Used only in the programs where difined	3) Can be used in any programs by including library.
4) It is customizeble and can be modified.	5) It's functionality is fixed and can't be modified

4) → A user defined function (U.D.f) can be created in three steps.

1) <u>function Prototype</u> (function decleration)

→ function should be decleared before they are used.

Syntax:

Data type function_name (Parameter list);

Ex,

int sum(int a, int b);

2) Function Defination:- function defination is the actual body of function wher the main blocue code is written and preform a specific tasles.

Syntax:-

```
data_type  function_name (function parameters)
{
    // code
}
```

Example:-

```
int sum (int a , int b)
{
    int sum;
    sum=a+b;
    return sum;
}
```

The major difference in function declaration and function defination is that the function delearation need a ";" after the line ends. and function defination is to write code inside the "{ }" to preform a tasle.

3) Function call:- Function call can be either with return value or argument called. ~~by calling User defined function are mainly~~.

user defined function are maily divided into four types.

1. With argument and with return value
2. With argument but without return value
3. Without argument but with return value
4. With out argument and without return value

1) ~~By wi~~

Wap to find sum of 2 numbers.

① Without Argument without return

```
#include<stdio.h>
void sum();
int main()
{
    int a,b,c
    sum();
    return 0;
}
void sum()
{
  int a,b,c;
  printf("Enter two numbers");
  scanf("%d %d",&a,&b);
  c=a+b;
  printf("Addition = %d", c);
}
```

2) Without Argument With return value:

```
#include<stdio.h>
int sum();
int main()
{
  int c;
  printf("sum = %d
  c= sum();
  printf("sum = %d", c);
  return;
}
int sum()
{
  int a,b;
```

```
    printf("Enter Two numbers");
    scanf("%d %d", &a, &b);
    return a+b;
}
```

③ With argument without return value

```
#include<stdio.h>
void sum(int, int);
int main()
{
    int a,b;
    printf("Enter Two numbers");
    scanf("%d %d", &a, &b);
    sum(a,b);
    return 0;
}
void sum(int a, int b)
{
    int c;
    c=a+b;
    printf("Sum=%d", c);
}
```

④ With argument with return value:

```
#include<stdio.h>
int sum(int, int);
int main()
{
    int a,b,c;
    printf("Enter Two numbers");
    scanf("%d %d", &a, &b);
    c=sum(a,b);
    printf("Sum = %d", c);
}
```

```
int sum(int a, int b)
{
    int c;
    c = a+b;
    return c;
}
```

Recursive function:- Recursion is a programing technique where a function calls itself untill the base case is found, when the base condition found it terminates it self.

Ex;

```
int fact(int n)
    if (n==1){
        return 0;
    }
    else
    {
        return n * fact(n-1);
    }
```

Importance of Recursion in C:

- Helps solve problems like factorial, fibonacci and tree traversal
- Reduces complex code for repetitive tasks.
- Makes code simpler and elegant for divid-and-conquer problem.
- Essential for algorithms in data structure.

6) Write programs

a) Wap to calculate Simple Interest by UDf named interest(). By return and argument.

```
#include<stdio.h>
int interest(int p, int t, int r)
{
  int i;
  i=(p*t*r)/100;
  return i;
}
int main()
{
    int p,t,r,i;
    printf("Enter P,T,R");
    scanf("%d %d %d", &p, &t, &r);
    i=interest(p,t,r);
    printf("Interest is %d",i);
    return 0;
}
```

b) Wap to calculate All the arthemetic operation by UDf.

```
#include<stdio.h>
int sum(int, int);
int difference(int, int);
int product(int, int);
int remainder(int, int);
float quotient(int, int);
int main()
{
    int a,b,s,d,p,r;
    float q;
    printf("Enter two numbers");
    scanf("%d", &b);
    scanf("%d %d", &a, &b);
```

```
    S = sum(a,b);
    d = difference(a,b);
    p = product(a,b);
    r = remaindr(a,b);
    q = quotient(a,b);
    printf("Sum = %d\n",S);
    printf("Difference = %d\n",d);
    printf("Product = %d\n",p);
    printf("Remainder = %d\n", r);
    printf("Quotient = %d\n",q);
}
int sum(int a, int b)
{
    return(a+b);
}
int difference(int a, int b)
{
    return(a-b);
}
int product(int a, int b)
{
    return(a*b);
}
int remaindr(int a, int b)
{
    return(a%b);
}
float quotient(int a, int b)
{
    return((float)a/b);
}
```

c) Wap to calculate the area, volume and circumfrenies.. of a sphere by UDf.

```
#include<stdio.h>
#define PI 3.1416
float area(int);
float vol(int);
float circumfrence(int);
int main()
{
    int r;
    float a,v,c;
    printf("Enter the Radius: ");
    scanf("%d",&r);
    a=area(r);
    bv= vol(r);
     c= circurfrence(r);
    printf("\nArea of Circle = %f",a);
    printf("\n circumfrence of Circle= %f",c);
    printf("\nVolume of sphere = %f",v);
    return 0;
}
float area(int r)
{
    float float area= 4*PI*r*r;
        return area;
}
float vol(int r)
{
    float volume;
    volume =(4.0/30)*PI*r*r*r;
    return volume;
}
float circumferenece(int r)
{ return 2*PI*r; }
```

Wap to find multiplication table by UDf named multi_table().

```
#include<stdio.h>
int table(int,int);
int main()
{
    int a,i,r;
    printf("Enter A Number");
    scanf("%d",&a);
    for(i=1;i<=10;i++)
    {
        r=table(a,i);
        printf("%d * %d = %d\n",a,i,r);
    }
}
int table(int a,int b)
{
    return a*b;
}
```

Wap to check either alphabet or not by UDf.

```
#include<stdio.h>
#include<string.h>
void alpha(char a);
int main()
{
    char a;
    printf("Enter a character:");
    scanf("%c",&a);
    alpha(a);
    return 0;
}
void alpha(char a){
    if((a>=65 && a<=90)||(a>=97 && a<=122)){
        printf("The Character %c Is An Alphabet",a);
    }
    else
```

```
{
    printf("The Character %c Is NOt An Alphate", a);
}
}
```

Wap to generate mirror value of a positive integers by UDf.

```
#include<stdio.h>
int mirror_value(int);
int main()
{
    int a,r;
    printf("Enter A Number:");
    scanf("%d",&a);
    r= mirror_value(a);
    printf("The Mirror Value Of %d IS %d", a, r);
    return 0;
}
int mirror_value(int a)
{
    int i, temp;
    int r=0;
    for(i=a ;i!=0 ; i/=10)
    {
        temp = i%10;
        r= r*10+temp
    }
    return r;
}
```

Wap to find HCF of two numbers By UDF.

```
#include<stdio.h>
int hcf(int, int);
int main()
{
    int a,b,hc;
    printf("Enter A Number: ");
    scanf("%d",&a);
    printf("Enter Another Number: ");
    scanf("%d",&b);
    hc=hcf(a,b);
    printf("The HCF of %d And %d Is %d",a,b,hc);
    return 0;
}
int hcf (int a, int b)
    {
      int m, i, hcf=1;
      min=(a<b)? a:b;
      for(i=1; i<=min; i++)
      {
          if(a%i==0 && b%i==0)
          {
              hcf=i;
          }
      }
      return hcf;
}
```

~~Wap to find out HCF of an~~
Wap to find calculate sum, average, max, ~~min~~ and min.

```
#include<stdio.h>
int sum(int []);
~~float average(~~
float average(int);
~~float~~ int minimum(int []);
int maximum(int []);

int main()
{
    int a[5], i, add, min, max;
    float av;
    for(i=0; i<5; i++)
    {
        printf("Enter The Integer %d : ", i+1);
        scanf("%d", &a[i]);
    }
    add = sum(a)
    av = average(add);
    min = minimum(a);
    max = maximum(a);
    printf("Sum = %d", add);
    ~~printf("Ave~~
    printf("\n Average = %f", av);
    printf("\n Maximum = %d ", max);
    printf("\n Minimum = %d", min);
    return 0;
}
int sum(int a[])
{
    int s=0, i;
    for(i=0; i<5; i++){ s=s+a[i];}
    return s;
}
```

```
float average(int add)
{
  float av=(float)add/15;
  return av;
}
int minimum(int a[])
{
      int min=a[0],i;
      for(i=1; i<15; i++)
      {
        if(a[i]<min) min=a[i];
      }
      return min;
}
int maximum(int a[])
{
      int max=a[0],i;
      for(i=1; i<15; i++)
      {
        if(a[i]>max) max=a[i];
      }
      return max;
}
```

Wap to sort the n. no. of intergers. by UDF

```
#include<stdio.h>
int sort(int b[], int a);
int main()
{
  int a,b[100],i,asc;
  printf("Enter The NO. Of Intergers: ");
  scanf("%d",&a);
  for(i=0;i<a;i++)
  {
    printf("Enter The Number %d: ",i+1);
    scanf("%d",&b[i]);
  }
```

```
asce = sort(b,a);
Printf("The sorted Array Is");
for(i=0; i<a ; i++)
 {
    printf("%d", b[i]);
 }
 return 0;
}
int sort(int b[] ,int a)
 {
    int i,j, temp;
    for(i=0; i<a ; i++)
     {
        for(j=i+1 ; j<a ;j++)
        {
           if(b[i] >b[j])
           {
              temp=b[j];
              b[j]=b[i];
              b[i] = temp;
           }
        }
     }
    return b[i];
}
```

Wap to count No. of words in a Enter in a sentence
By UDf.

```
#include<stdio.h>
#include<string.h>
int word_count(char a[])
{
    int i, word =1;
    for(i=0; a[i] != '\0' ; i++)
    {  if (a[i] == ' '){
        word++;
    } }
```

```
    return word;
}
int main()
{
    int i, word=1;
    char a[100];
    printf("Enter A sentence:");
    gets(a);
    printf("The Number of word Is %d", wordcount(a));
    return 0;
}
```

Prog to check the word is either palin or not palindrome or not.

```
#include<stdio.h>
#include<string.h>
void rspalin(char c[], char a[]);
int main()
{
    char a[100], c[100];
    printf("Enter a word");
    gets(a);
    strcpy(c,a);
    strrev(a);
    rspalin(c,a);
    return 0;
}
void rspalin(char c[], char a[])
{
    char b[100];
    strcpy(b,a);
    strrev(b);
    if(strcmp(a,b)==0)
    {
        printf("The word %s Is palindrome\n",c);
    }else{
        printf("The word %s Is Not palindrom\n",c);
    }
}
```

Wap to find factorial to tve interger by recursive function.

```
#include<stdio.h>
int fact(int);
int main()
{
    int n,f;
    printf("Enter Any positive Integer:");
    scanf("%d",&n);
    f=fact(n);
    printf("The factorial of %d is %d",n,f);
    return 0;
}
int fact(int n)
{
    if(n==1)
    {
        return 1;
    }else
        return n*fact(n-1);
    }
}
```

Wap to ~~find~~ print fibonacci up to 25th term. By recursive fun.

```
#include<stdio.h>
int fibonacci(int);
int main()
{
    int i;
    int n=25;
    for(i=0;i<n;i++)
    {
        printf("%d",fibonacci(i));
    }
    return 0;
}
```

```
int fibonacci(int n)
{
    if (n==0){
       return 0;
    }else if (n==1){
       return 1;
    }else{
       return fibonacci(n-1) + fibonacci(n-2);
    }
}
```

Wap to ~~find~~ print hailstone pattern up to 2nd term by recursive function.

```
#include<stdio.h>
void hailstone(int ,int);
int main(){
    int a=7, b=12;
    hailstone(a,n);
    return 0;
}
void hailstone(int a, int b)
{
   int i;
   printf("%d ", a);
   for(i=1;i<n;i++){
      if((a%2)==0){
          a=a/2;
      }else{
          a=(a*3)+1;
      }
       printf("%d", a);
   }
```