FUNCTION OF C LANGUAGE.

Function is a self-statement of block which is used to divide the large programme in to

Sub module or small part for simplicity of application or programme.

What is a benefit of function.

The main benefit of function is define once and use multiple times.means if we want to reuse

The function code simply call it.

Types of function.

There are two types of function.

1] library function:-library function means those function provided by c language use can not edit in library function.

e.g. printf(),scanf()etc.

can not write print() or scanf() etc.

2] user defined function:- those function defined by use for its own use is called as user

Define function.user defined function name depend on user choice.

Remember if we want to use the any user defined function for that we need to use the three important things.

1]function declaration /prototype of function

Declaration specify which function we will use in programme its name as well as its return fun type as well as how many values function can transper and its datatype.

2]function calling:- function call means execute or run the body of function or definition function function calling generally use for reuse the function definition.

3] function definition/called function:- function definition means write the actual bodynof function or write the code of function.

Types of user define function

1]function with returning value with argument or parameter

2] function without returning value with argument

3]function without returning value without argument.

4]function with returning value without argument.

How to declare the function in programme

Syntax:- returntype function name (datatype);

e.g. int add(int,int);

above declaration says we will use the user define function in programme namely add and this function return the integer type value and this function can pass the two values of type integer .

note:- if we give the return type of function as void then function cannot return the any value.

How to call the function in c language.

1] if function return values

How to call function in function has return type.

Return value=function name(variable name as per declaration);

e.g. c=add(a,b);

here in above example a,b,c must be integer type because declaration says.

2] if function not return the value

How to call the function if function has no return type.

Function name(variable);

e.g.add(a,b);

here add cannot return the value if add() function return type is void.

How to define the function in c programme.

e.g. return type functionname (data type variable name,datatype variable name)

{

Body of function

}

e.g. int add(int x,int y)

{

Return(x+y);

}

Note:-- cannot define function in main function this is the rule of c language

Function define must be before man of after after main.

//1] programme using function with return type with argument

```
#include<stdio.h>
#include<conio.h>
void main(){
int no,sq;
int squre (int);
clrscr();
printf("enter the number for squre \n");
scanf("%d",&no);
sq=squre(no);
printf("squre  of number  is %d",sq);
getch();
}
Int squre(int x)
{
```

Return(x*x);

}

Output:-

```
enter the number for squre
8
squre  of number  is 64_
```

//2.program using without return value without argument.

#include<stdio.h>

#include<conio.h>

void main(){

void squre();

clrscr();

squre();

getch();

}

void squre ()

{
int no;

printf(“enter the value for squre\n”);

scanf(“%d”,&no);

printf(“squre of number is %d”,no*no);

}

Output:-

```
enter the value for squre
25
squre of number is 625_
```

//3.program with returning value without argument

```
#include<stdio.h>

#include<conio.h>

void main(){

int squre();

clrscr();

printf("squre of number is %d",squre());

getch();

}

void squre ()

{

int no;

printf("enter the value for squre\n");

scanf("%d",&no);

return(no*no);

}
```

Output:-

CALL BY VALUE AND CALL BY REFERENCE

Call by value:-

Call by value means actual argument pass the photo copy to formal argument or calling function pass the duplication value to called or definition of function.means if we made any change on formal argument not affect on actual argument.

Actual argument means calling function argument and formal argument means called function argument.

//Following program demonstrate the call by value.

```
#include<stdio.h>

#include<conio.h>

void main(){

int a,b,c;

clrscr();

void swap(int ,int);

printf(“enter two values”);

scanf(“%d %d”,&a,&b);

swap(a,b);

printf(“A=%d\t B=%d\n”,a,b);

getch();

}

void swap(int x,int y){

temp;

x=y;

y=temp;

printf(“X=%d\t Y=%d\n”,x,y);

}
```

Output:-

This program output

X=20 y=10

X=10 y=20

If we swap the values in definition not effect on calling function argument.

So the drawback of call by value is cannot access the more than one return value from function

Because function return the single value from function at a time.

So the we need to call by value and call by reference we send the address of calling function

Argument to called function argument or actual argument address to the formal address.

Call by reference:- in call by reference we need to pass the actual argument address to formal argument so we need to declare argument type is pointer declaration.

```
#include<stdio.h>

#include<conio.h>

void main(){

int a,b;

clrscr();

void swap(int *,int *);

printf(“enter the two values “);

scanf(“%d%d”,&a,&b);

swap(&a,&b);

printf(“A=%d\t B=%d\n”,a,b);

getch();

}
```

```
void swap(int *x,int *y){

temp=*x;

*x=*y;

*y=temp;

printf("X=%d\t Y=%d\n",x,y);

        }
```

This program generate the output

X=10 y=20

A=20 B=10.

Note:- if we want to learn the detail pointer with function for that refer the campus oriented c notes of girisir or mca –cet notes of girisir).

FUNCTION RECURSION

Function recursion means when function call itself again and again is called as function recursion. Function recursion is a same like as loop.

Or

Call same function within its definition is called as function recursion.

//Following program shows the use of function recursion.

```
#include<stdio.h>
#include<conio.h>
void main(){
void recursion(int);
int value;
clrscr();
printf("enter the value");
scanf("%d",&value);
recursion(value);
getch();
}
void recursion(int x){
If(x==0)
printf("stop function execution \n");
break;
else
```

```
printf("\n good morning");
recursion(x-1);// recuesive function call made here .
}
```

Output:-

above program shows the five times good morning message.
because first time function call and jumb of function definition then consider the value of
X is 5 then compare x value with 0 in function definition then condition is false and execute the else part of function and print good morning message then execute the statement recursion (x-1)then x value is 4 and function again jump on function definition and compare x value with 0 then again condition is false print good morning message this type of iterative work done whenever x value become a zero. When x value is 0 then if condition true and give the message stop function execution and stop the execution of program.

Solved program of function.

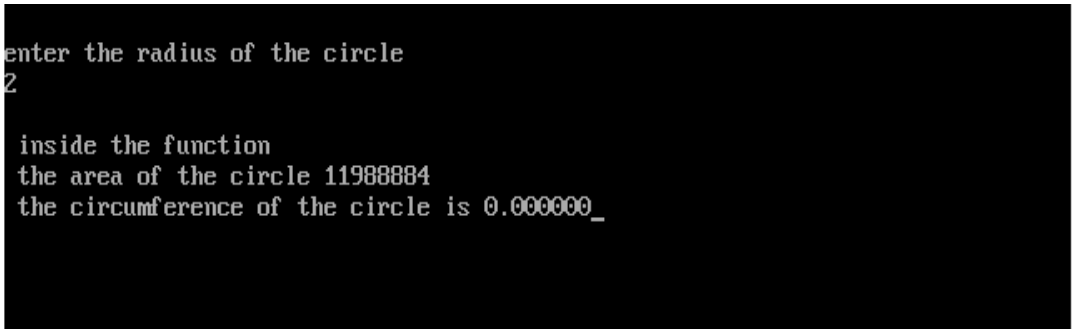
//Finding the area and circumference of circle using the function.

```
#include<stdio.h>
#include<conio.h>
void find(int);

void main(){
```

```
int radius;
clrscr();
printf(“enter the radius of the circle”);
scanf(“%d”,&radius);
find(radius);
getch();
}
voidfind(int){
float area;
printf(“\n inside the function”);
area=3.14*r*r;
circum=2*3.14*r;
printf(“\n the area of the circle %2.f”,area);
printf(“\n the circumference of the circle is %2f”,circum);
}
```

Output:

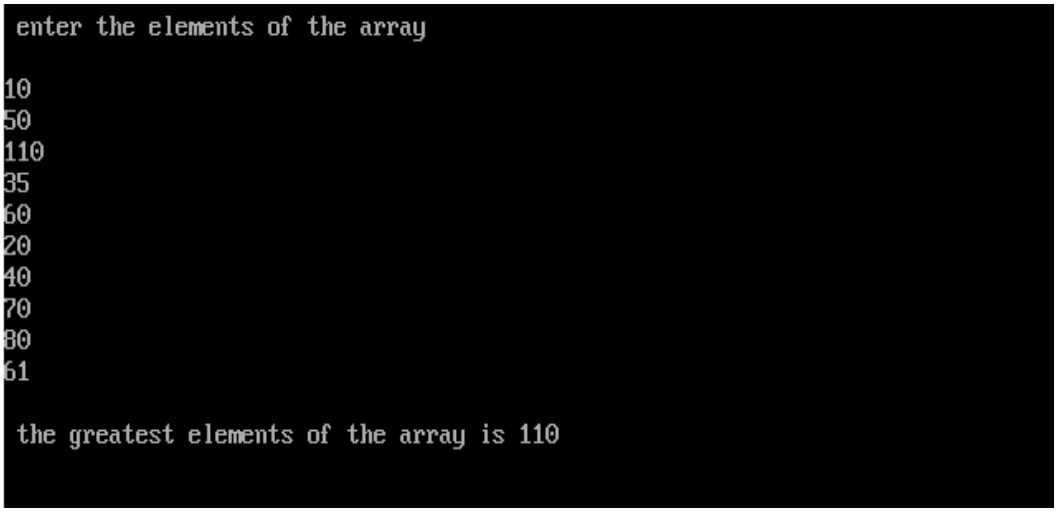


//Programme for return the greatest element from the array

```
#include<stdio.h>
#include<conio.h>
int findgreat(int *);
void main(){
int num[10],i,gr;
clrscr();
printf(“\n enter the elements of the array\n”);
for(i=0;i<10;i++)
scanf(“%d”,&num[i]);
gr=finfgreat(num);
printf(“\n the greatest elements of the array is %d”,gr);
getch();
}
int findgreat(int *ptr)
{
int i,great;
```

```
great=*ptr;
for(i=1;i<10;i++)
{
ptr++;
if(*ptr>great)
great=*ptr;
}
return great;
}
```

Output:



//Program finding the cube using the function.

```
#include<conio.h>
#include<stdio.h>
void cube(int x){
void main(){
int num,cube of num;
clrscr();
printf("enter the number");
scanf("%d",&num);
cubeofnum=cube(num);
printf("\n the cube of the number is %d",cubeofnum);
getch();
}
int cube (int x)
{
int y;
y=x*X*X;
printf("\n inside the function");
return y;
```

```
}
```

//Returning factorial from the function.

```
#include<stdio.h>
#include<conio.h>
long findfact(int);
void main(){
int num,i;
long fact;
clrscr();
printf("\n enter the number");
scanf("%d",&num);
fact=findfact(num);
printf("\n the factorial of %d is %ld",num,fact);
getch();
}
long findfact(int num)
{
int i;
long fact=1;
printf("\n inside the function");
for(i=1;i<=num;i++)
{
fact=fact*i;
}
return fact;
}
```

Output:

//Program finding the sum and average using the function

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

```
        void main(){
        int m1,m2,m3,m4,m5,total;
        float average;
        clrscr();
        printf("\n marks of 5 subjects");
        scanf("%d %d %d %d %d ",&m1,&m2,&m3,&m4,&m5);
        getsum_avg(m1,m2,m3,m4,m5,&total,&average);
        printf("\n the sum is %d",total);
        printf(\n the average.is %f",average);

        *s=m1+m2+m3+m4+m5;

*a= *s/5.0;

}
```

//Finding the raise to power of number

```
#include<stdio.h>

#include<conio.h>

getapower(int x);

void main()

int num1,num2,power;

clrscr();

printf("enter any two number");

scanf("%d%d",&num1,&num2);

power=getpower(num1,num2);

printf("\n num1 raise to power num2 is %d",power);

getch();

}
int getpower (int num,int pow){

int i,z=1;

for(i=0;i<pow;i++)

z=z*num;
```

```
return z;
}
```

//Finding the alternate number up to the given rangeusing the recursive function.

```
#include<stdio.h>
#include<conio.h>
int add(int n);
void main(){
int num,sum;
clrscr();
printf("\n enter the number");
scanf("%d",&num);
sum-add(num);
printf("\n the sum of the number is %d",sum);
getch();
}
int add(int n)
{
int sum;
if(n==0)
return 0;
else
if(n==1)
return 1;
sum=n+add(n-2);
```

return sum;

}

Output: