PPS Unit 4

Function
A function is a self-contained program segment that **carries out some specific, well-defined task**. Every C program consists of one or more functions. One of these functions must be called main.
Execution of the program always begins by carrying out the instructions contained in main.

Advantages of functions:

1)Code Reuse 2)Easy to find errors
3)Large program divided in parts

Funtion Types :

Inbuilt Function: They are **predefined** in the library of the C. These are used to perform the most common operations like calculations, updation, etc. Some of the library functions are **printf, scanf, sqrt, etc**

User Defined Function: To perform the according to the **requirement of user** the user have to develop some functions by itself, these functions are called **user-defined functions**.

Parts of Function:

a)Prototype b) function calling c) Function definition

1)Function Declaration **or Function Prototype**: A function prototype gives information to the compiler that the function may later be used in the program.. It doesn't contain function body.

2)**Calling a Function**: When a program calls a function, the program control is transferred to the called function.

3)**Define a Function:** The function body contains a collection of statements that define what the function does.

Parameters in C

The variables that are defined when the function is declared are known as parameters. These are used in function call statements to send value from the calling function to the receiving function.

Formal parameter: .

The variables declared in the function prototype or definition are known as Formal arguments .

Example: **void add(int a ,int b);**

Actual arguments.

the values that are passed to the called function from the main function are known as Actual arguments.

Example: add(p,q);

Categories of Functions:

Categories of Function:

1)Functions without arguments and without return values.

2)Functions without arguments and with return values.

3)Functions with parameters and without return values.

4)Functions with parameters and return values.

Category 1

Functions without arguments and without return values

Program to add two numbers

```
#include<stdio.h>
void add(); //function prototype
void main()
{
add();
}
void add()
{
int c;
printf("enter two values\n");
scanf("%d%d",&p,&q);
c=a+b;
printf(" add is %d",c)
}
```

Category 2:

Functions without arguments and with return values.

Program to substract two numbers

```
#include<stdio.h>
int sub(); //function prototype
void main()
{
int p;
p=sub();
printf("substraction of numbers is %d",p);
}
int sub()
{
int c;
printf("enter two values\n");
scanf("%d%d",&p,&q);
c=a-b;
return c;
}
```

Category 3:

Functions with parameters and without return values.

```
#include<stdio.h>
void add(int a,int b);    //formal parameter and function prototype
void main()
{
int p,q;
printf("enter two values\n");
scanf("%d%d",&p,&q);
add(p,q);  // actual argument
}
void add(int a,int b)
{
Int c;
c=a+b;
printf(" add is %d",c)
}
```

Output: Enter two numbers 7 8

add is 15

Category 4

Functions with parameters and with return values.

Program to substract two numbers .

```
#include<stdio.h>
int add(int a,int b);
void main()
{
int p,q,z;
printf("enter two values\n");
scanf("%d%d",&p,&q);
z=add(p,q);  // actual argument
printf("substract is %d",z);
}
int add(int a,int b)
{
Int c;
c=a-b;
return c;
}
```

Output :

enter two values

12 14

substract is -2

Call by value and call by reference in C

Type of Passing	A copy of the value is passed into the function in call by value	An address of value is passed into the function in call by reference
Changes	Changes made inside the function is limited to the function only in call by value.	Changes made inside the function validate outside of the function also in call by reference.
Memory Location (Referred)	The memory location referred to by the actual arguments and passed parameters of a function are different. In call by value.	The memory location referred to by the actual arguments and passed parameters of a function are the same

Pointer: Pointer is a variable that holds address of another variable. To declare a pointer, we use the (*) operator before its name.

Pointer Example

```
#include<stdio.h>
void  main()
{
int  a=50;
int *p;
p=&a;//stores the address of number variable
printf("Address of p variable is %d\n",p);
printf("Value of p variable is %d \n",*p);
 printf("Address of a variable is %d \n",&a);
}
```

Output:

```
Address of p variable is 2018028084
Value of p variable is 50
Address of a variable is 2018028084
```

Call by Value Example: Swapping the values of the two variables

```
1. #include <stdio.h>
2. void swap(int  x, int y); //prototype of the function
3. void main()
4. {
1.     printf("Enter 2 numbers\n");
2. scanf("%d%d",&a,&b);
5.     printf("Before swapping the values in main a = %d, b = %d\n",a,b);
6.   swap(a,b);
7.     printf("After swapping values in main a = %d, b = %d\n",a,b);
8.   }
9. void swap (int a, int b)
10.        {
11.            int temp;
12.            temp = x;
13.            x=y;
14.            y=temp;
15.            printf("After swapping values in function x = %d, y = %d\n",x,y); //
   Formal parameters, a = 20, b = 10
16.        }
```

Output

```
Before swapping the values in main a = 10, b = 20
After swapping values in function x= 20, y = 10
After swapping values in main a = 10, b = 20
```

Call by reference program to swap 2 numbers

```
3.
4.  #include <stdio.h>
5.  void swap(int *x, int *y); //prototype and formal parameter
6.  void main()
7.  {
8.      int a,b;
9.  printf("Enter 2 numbers\n");
10.     scanf("%d%d",&a,&b);
11.         printf("Before swapping the values in main a = %d, b = %d\n",a,b);
12.         swap(&a,&b);  //calling and actual parameter
13.         printf("After swapping values in main a = %d, b = %d\n",a,b);
        }
14.     void swap (int *x, int *y)  // Formal parameters, a = 20, b = 10
15.     {
16.         int temp;
17.         temp = *x;
18.         *x=*y;
19.         *y=temp;
20.         printf("After swapping values in function x = %d, y = %d\n",*x,*y);
21.     }
```

Output

```
Before swapping the values in main a = 10, b = 20
After swapping values in function x = 20, y = 10
After swapping values in main a = 20, b = 10
```

Recursion in C

A function that calls itself is known as a recursive function. And, this technique is known as recursion. Recursion must have a terminating condition also known as base case.

The base case in a recursive algorithm is *the case that is no longer recursive*.

factorial program in c using recursion.

```
1. #include<stdio.h>
2. int fact(int n)  ;
3.   void main()
4. {
5.   int num,r;
6.     printf("Enter a number:\n ");
7.   scanf("%d", &num);
8.      r = fact(num);
9.   printf("Factorial of %d ",  r);
10.        }
11.        int fact(int n)
12.        {
13.          if (n == 0)
14.            return 1;
15.          else
16.            return(n * fact(n-1));
17.        }
```

Output:

```
Enter a number: 6
Factorial of 5 is: 720
```

Fibnocci series using recursion:

0 1 1 2 3 5 8 13

```
#include <stdio.h>
int fib(int num);   //prototype
void main()
{
   int num;
   printf("Enter the number of elements to be in the series : ");
   scanf("%d", &num);
   for (int i = 0; i < num; i++)
   {
      printf("%d, ", fib(i));
   }
  }

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

Output:

Enter the number of elements to be in the series : 5

0 ,1 ,1 ,2 ,3,

Searching in Arrays:

Searching is used to find position of an element in an array of elements.For example:Six intergers 23 45 67 89 34 56

Search 89

89 is at position 4

There are two methods of searching:

1)Linear Search:Search an element one by one in an array.Its search time complexity is

Program for linear search

#include<stdio.h>
void main()
{
int pos,i,a[10],n,num,flag=0;
printf("enter how many integers\n");
scanf("%d",&n);
printf("enter elements\n");
for(i=0;i<n;i++)
scanf("%d",&a[i]);
printf("enter number to search\n");
scanf("%d",&num);
for(i=0;i<n;i++)
{
if(a[i]==num)
{
pos=i;
flag=1;
break;
} }
if(flag==1)
printf("number is at %d position",pos+1);
else
printf("number not found");
}
	Output:
	enter how many integers
	5
	enter elements
	12 3 56 78 90
	enter number to search
	78
	number is at 4 position
	
	Its time complexity is
	Best case:
	O(1)
	Average case:
	O(N)
	Worst case: O(N)

Program for binary search in C

```
#include<stdio.h>
void main()
{
   int arr[50],i,n,num,flag=0,first,last,mid;

   printf("Enter size of array:");
   scanf("%d",&n);
   printf("\nEnter array
element(ascending order)\n");
     for(i=0;i<n;++i)
      scanf("%d",&arr[i]);
     printf("\nEnter the element to
search:");
   scanf("%d",&num);
     first=0;
   last=n-1;
     while(first<=last)
   {
      mid=(first+last)/2;

      if(num==arr[mid]){
         flag=1;
         break;
      }
      else
         if(num>arr[mid])
            first=mid+1;
         else
            last=mid-1;    }

 if(flag==1)
      printf("\nElement found at position
%d",mid+1);
   else
      printf("\nElement not found");
}
```

2) Binary Search:**Binary search requires a sorted array**.In this array is divided in parts so this is also called divide and conquer method.It takes less time then linear search.Its time complexity is

Best case- O(1)

Average case – O(Log2N)

Worst case – O(Log2N)

Output:

size of array:5

Enter array element(ascending order)

12 34 56 78 90

Enter the element to search:78

Element found at position 4

Sorting: Sorting is a process of arranging data in a particular order either ascending or descending.

For example consider array elements 12 3 56 7 1

Its ascending order is 1 3 7 12 56

Three important searching are:

1.Bubble Sort:

Bubble sort program

```
#include<stdio.h>
  void main()
{
int a[50],n,i,j,temp;
printf("Enter the size of array: ");
scanf("%d",&n);
printf("Enter the array elements: ");
for(i=0;i<n;++i)
scanf("%d",&a[i]);
for(i=0;i<n-1;++i)
{
for(j=0;j<(n-i-1);++j)
{
if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}}}
printf("\nArray after sorting: ");
for(i=0;i<n;++i)
printf("%d ",a[i]);

}
```

Output:

Enter the size of array: 4

Enter the array elements: 3 5 1 2

Array after sorting: 1 2 3 5

Selection Sort:

```
#include<stdio.h>

void  main()
{
   int i,j,n,temp,a[30];
   printf("Enter the number of
elements:");
   scanf("%d",&n);
   printf("\nEnter the elements\n");

   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }

   for(i=0;i<n-1;i++)
   {
     for(j=i+1;j<n;j++)
       {
if(a[i]> a[j])
{

temp=a[i];
       a[i]=a[j];
       a[j]=temp;

          }
       }
   }
    printf("\nSorted list is as
follows\n");
   for(i=0;i<n;i++)
   {
      printf("%d ",a[i]);
   }
}
```

Output:

```
 Enter the number of
elements:4
Enter the elements
1 40 3 26
Sorted list is as follows
1 3 26 40
```

Insertion Sort:

```
#include<stdio.h>
  void main()
{
   int i,j,n,temp,a[30];
   printf("Enter the number of
elements:");
   scanf("%d",&n);
   printf("\nEnter the elements\n");
     for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }

 for(i=1;i<=n-1;i++)
   {
      temp=a[i];
      j=i-1;
        while((temp<a[j])&&(j>=0))
      {
         a[j+1]=a[j];
 j=j-1;
      }

      a[j+1]=temp;
   }

 printf("\nSorted list is as
follows\n");
   for(i=0;i<n;i++)
   {
      printf("%d ",a[i]);
   }
}
```

Output:

```
Enter the number of elements:4
Enter the elements
12 67 2 9
Sorted list is as follows
2 9 12 67
```

Functions with Array:

Functions and arrays can be used together in the same program.

syntax to pass an array to the function.

functionname(arrayname);//passing array

Program to add array elements using functions:

```
#include <stdio.h>
void sum_array(int a[],int n);
void main()
{
   int a[10],n,i;
   printf("enter how many numbers \n");
   scanf("%d",&n);
   printf("Enter the numbers\n");
   for(i=0;i<n;i++)
   scanf("%d",&a[i]);
   sum(a,n);
   }
      void sum(int a[],int n)
  {
     int sum=0,i;
      for(i=0;i<n;i++)
      {
       sum=sum+a[i];

      }
      printf("sum is %d",sum);
 }
```

Output:

```
enter how many numbers
4
Enter the numbers
1 4 6 8

sum is 19
```