UNIT-2:-

Functions

Every C-programme has at least one function which is main function. It is a mini-program (or) sub program.

In C-programming functions are divided into two types.

(i) Library functions (or) pre-defined functions.
(ii) User defined functions.

---

pre-defined function:-

We do not need to write a code for pre-defined function. It is already present inside the header file, we always includes at the begining of a programe.

ex:- printf(), scanf(), clrscr(), getch(), etc...

User defined function:-

User defined function in C is always written by the user. We have to write a body of a function and call the function whenever we require it.

Functions are divided into three activities,

(i) Function declaration. (befor the main)
(ii) Function calling (inside the main)
(iii) Function definition (outside the main).

Function declaration:-

It is declared before the main. Function declaration is also called as "Function prototype".

Syntax:-

return_data-type function-name (data-type parameters);

Function declaration must be end with ";" (semicolone).

Function definition:-
It means just write the body of a function. It consists of statements which are going to be perform a specific task. It is mondatory part of a function. It is declared outside the main.

Function calling:-
Whenever we call a function, it performs an operation. It is optional part of the function.

Function arguments or Parameters:-
Function arguments are used to recieve the neccessary values by the function calling.

Variable scope:-
Variable scope means the visibility of variables within a code of the programme.
In C-language, variables are declared inside the main those varivariables are called as local variables. Variables are declared before the main those variables are called as global variables.
Local variables are used within the code of program. But global variables are used entire program in any time, any place, any where.

Static variable:-
Static variables have a local scope. They are not destroyed when exiting the function. But local variables which will be destroyed when exiting the function. Static variable retains its value forever.

Inline functions:-
Inline functions are used for small computations. They are not suitable when large computation is involved.

syntax :-

```
inline function-name ()
{ ======= ; // function definition.
}
```

Functions Types :-

There are 4 different types of user defined function in C.

(i) Function with no argument and no return value.
(ii) Function with no argument and a return value.
(iii) Function with argument and no return value.
(iv) Function with argument and a return value.

Nesting of Functions :-

To use one function inside another function body. it is called nesting of functions.

What is Recursion ? :-

Recursion is a special way of nesting functions, where function calls itself inside it.

ex: Factorial of the given number, Towers of hanoi.

Pointers :-

A pointer is a variable whose value is the address of another variable. i.e, direct address of the memory location.

Syntax :-

```
datatype *var_name;
```

ex: `int *p;`
`int *ip;`

* is called as asterisk.

The asterisk used to declare a pointer is the same asterisk used for multiplication.

Advantages of pointers:-

(i) Pointers increases the execution speed.
(ii) Pointers enables us to access a variable that is defined outside the function.
(iii) Pointers reduces the length and complexity of a programme.

The address of the variable is done with the help of the operator & available in C.

Pointer declaration styles:-

```
int *p;
int *p;
int * p;
```

Note:- The operator & returns the memory address of variable on which it is operated, this is called referencing.

The * operator is called an indirection operator or dereferencing operator.

Note:- Many object oriented features in C++ are implemented using function pointers in C.

Storage classes in C:-

Storage classes in C are used to determine the, lifetime, visibility, memory allocation and initial value of a variable.

There are four types of storage classes in C,

(i) Automatic, (ii) External
(iii) Static, (iv) Register.

Automatic :-

(i) Automatic variables are allocated memory automatically.

(ii) The visibility and scope of the automatic variables is limited.

(iii) The automatic variables are initialized to garbage by default.

(iv) We can only initialize the automatic variables locally.

(v) auto keyword is used for defining the automatic variables.

External :-

(i) The external variables not allocated any memory.

(ii) The external variables are initialized to zero otherwise null.

(iii). We can only initialize the external variables globally.

(iv) External variables are declared any where in the program.

(v) The lifetime of the external variable is till the end of the main program.

(vi) External variables can be declared many times but can be initialized only one time.

(vii) extern keyword is used for defining the external variables.

Static :-

(i) The visibility and scope of the static variable is limited. Static variables are visible only to the function.

(ii) Default initial value of the static variable is zero. otherwise null.
(iii) static variables can be declared many times but can be assigned only one time.
(iv) static keyword is used for defining the static variables.

Register :-

(i) Register variables are allocated the memory into the CPU.
(ii) Default initial value of the register variable is 0.
(iii) The access time of the register variable is faster than automatic variables.
(iv) We can not use "&" operator for the register variables.
(v) register keyword is used for defining the register variables.

Storage classess	Storage Place	Default Value	Scope	Lifetime.
auto	RAM	Garbage	Local	within the function.
extern	RAM	zero	global	Till the end of the main program, may be declared any where in the program.
static	RAM	zero	Local	Till the end of the main program, retains value b/w multiple functions call.
register	register	garbage	local	within the function.

Structures :-

(i) Structure is a collection of non-homogeneous elements. It is declared before the main. It is a user defined datatype in C.

(ii) Struct keyword is used to creat a structure.

(iii) Structure can not be initialized with declaration.

ex:- struct student
{
int rno = 1;
char a[20] = "rsrcd"; "X" It is wrong.
};

(iv) Structure members are accessed using dot (.) operator.

(v) Structure members can be initialized using curly braces "{}".

(vi) Memory is allocated only when the variables are created.

(vii) We can not use operators like +, - etc. on structure variables.

(viii) Structure can access all data items at a time.

(ix) Structure provide a method for packing together data of different types.

(x) Size of structure is equal to the sizes of the different datatypes.

(i) C structure do not permit data hiding.

(ii) C structure do not permit functions inside structure.

(iii). C structures can not have a static members inside their body.

(iv). C programming language do not support access modifiers, so they can not be used in C structures.

(v) C structures can not have constructors inside structures.

Unions :-

(i) Union is a collection of non-homogeneous elements.
(ii) It is a user defined data type similar to the c structures.
(iii) Union keyword is used to defin the unions.
(iv) Memory is allocated ~~when~~ only when variables are created.
(v) union can access only one union member at a time.
(vi) size of union is equal to the size of the largest datatype or largest member.
(vii) We used (.) operator to access members of a union.

C Strings :-

String is an array of characters. In order to use string functions in our program, we must include <string.h> header file.

Declaration of String is

```
char str1[] = {'A','B','C','D','\0'};
char str1[] = "ABCD";
```

String I/O in C-programming :-

(i) printf and scanf
(ii) puts and gets.

```
printf("%s", str1);
puts(str1);
scanf("%s", &str1);
gets(str1);
```

String Functions in C :-

strlen, strlwr, strupr, strcat, strcpy, strcmp, strdup, strchr, strstr, strset, strrev.

strlen = String length.
strlwr = string to lowercase
strupr = string to uppercase
strcat = concatenation of strings.
strcpy = string copy
strcmp = string compare
strdup = string duplicate
strrev = string reverse.

Queue Operations:-

```
#include <stdio.h>
#include <conio.h>
#define max 5
void main()
{
 int queue[max], max=n, i, j, f=0, r=0, choice;
 printf ("Queue operations \n");
 printf ("1. Enqueue()\n2. Dequeue()\n3. Display\n4. Exit\n");
 while (choice!=4)
   {
    printf ("Enter the choice\n");
    scanf ("%d", &choice);
    switch (choice)
     {case 1: if (r==n)
                 printf ("Queue is full\n");
              else
                 {int val;
                  printf ("Enter val\n");
                  scanf ("%d", &val);
                   r++;
                   queue[r] =val;
                 }
              break;
```

```
case 2: if(f==-1)
            printf("Queue is empty\n");
        else
        {
            f++;
        }
        break;
case 3: printf("Queue elements are\n");
        if (f==-1)
        {
            printf("Queue is empty\n");
        }
        else
        {
            for(i=f; i<q; i++)
            {
                printf("%d\n", queue[i]);
            }
        }
        break;
case 4: printf("Exiting\n"); break;
default: printf("Invalid choice\n");
    }
  }
}
```