```
{
        int num;
        printf("enter a number");
        scanf("%d",&num);
        f=fact(num);
        printf("factorial is =%d\n"f);
}
fact (int num)
{
        If  (num==0||num==1)
return 1;
else
return(num*fact(num-1));
}
```

Lecture Note: 18

Monolithic Programming

The program which contains a single function for the large program is called monolithic program. In monolithic program not divided the program, it is huge long pieces of code that jump back and forth doing all the tasks like single thread of execution, the program requires. Problem arise in monolithic program is that, when the program **size** increases it leads inconvenience and difficult to maintain

such as testing, debugging etc. Many disadvantages of monolithic programming are:

1. Difficult to check error on large programs size.
2. Difficult to maintain because of huge size.
3. Code can be specific to a particular problem. i.e. it cannot be reused.

Many early languages (FORTRAN, COBOL, BASIC, C) required one huge workspace with labelled areas that may does specific tasks but are not isolated.

Modular Programming

The process of subdividing a computer program into separate sub-programs such as functions and subroutines is called Modular programming. **Modular programming sometimes also called as structured programming.** It enables multiple programmers to divide up the large program and debug pieces of program independently and tested.

. Then the linker will link all these modules to form the complete program. This principle dividing software up into parts, or modules, where a module can be changed, replaced, or removed, with minimal effect on the other software it works with. Segmenting the program into modules clearly defined functions, it can determine the source of program errors more easily. Breaking down program functions into modules, where each of which accomplishes one function and contains all the source code and variables needed to accomplish that function. Modular program is the solution to the problem of very large program that are difficult to debug, test and maintain. A program module may be rewritten while its inputs and outputs remain the same. The person making a change may only understand a small portion of the original program.

Object-oriented programming (OOP) is compatible with the modular programming concept to a large extent.

. , Less code has to be written that makes shorter.

- A single procedure can be developed for reuse, eliminating the need to retype the code many times.
- Programs can be designed more easily because a small team deals with only a small part of the entire code.
- Modular programming allows many programmers to collaborate on the same application.
- The code is stored across multiple files.
- Code is short, simple and easy to understand and modify, make simple to figure out how the program is operate and reduce likely hood of bugs.
- Errors can easily be identified, as they are localized to a subroutine or function or isolated to specific module.
- The same code can be reused in many applications.
- The scoping of variables and functions can easily be controlled.

Disadvantages
However it may takes longer to develop the program using this technique.

Storage Classes

Storage class in c language is a specifier which tells the compiler where and how to store variables, its initial value and scope of the variables in a program. Or attributes of variable is known as storage class or in compiler point of view a variable identify some physical location within a computer where its string of bits value can be stored is known as storage class.

The kind of location in the computer, where value can be stored is either in the memory or in the register. There are various storage class which determined, in which of the two location value would be stored.

Syntax of declaring storage classes is:-

storageclass datatype variable name;

There are four types of storage classes and all are keywords:-

1) Automatic (auto)

2) Register (register)

3) Static (static)

4) External (extern)

Examples:-

auto float x; or float x;

extern int x;

register char c;

static int y;

Compiler assume different storage class based on:-

1) Storage class:- tells us about storage place(where variable would be stored).

2) Intial value :-what would be the initial value of the variable.

If initial value not assigned, then what value taken by uninitialized variable.

3) Scope of the variable:-what would be the value of the variable of the program.

4) **Life time :-** It is the time between the creation and distribution of a variable or how long would variable exists.

1. Automatic storage class

The keyword used to declare automatic storage class is auto.

Its features:-

Storage-memory location

Default initial value:-unpredictable value or garbage value.

Scope:-local to the block or function in which variable is defined.

Life time:-Till the control remains within function or block in which it is defined. It terminates when function is released.

The variable without any storage class specifier is called automatic variable.

Example:-

```
main( )
{
auto int i;
printf("i=",i);
}
```

Lecture Note: 19

2. Register storage class

The keyword used to declare this storage class is register.

The features are:-

Storage:-CPU register.

Default initial value :-garbage value

Scope :-local to the function or block in which it is defined.

Life time :-till controls remains within function or blocks in which it is defined.

Register variable don't have memory address so we can't apply address operator on it. CPU register generally of 16 bits or 2 bytes. So we can apply storage classes only for integers, characters, pointer type.