Software:

Software is a combination of:

1) basically instructions (or set of instructions or computer logical programs) that when these instructions are executed then we can perform desired functions and maintain the performance,
2) data structures that enable the programs to adequately manipulate information for the required results, and,
3) documents that describe the operation and use of the programs for the better understandability of the user of the program.

Computer software, or just **software**, is a collection of computer programs and related data that provide the instructions for telling a computer what to do and how to do it. In other words, software is a conceptual entity which is:

1) a set of computer programs,
2) procedures, and
3) associated documentation concerned with the operation of a data processing system.

We can also say software refers to one or more computer programs and data held in the storage of the computer for some purposes.
In other words software is a set of **programs, procedures, algorithms** and its **documentation**.

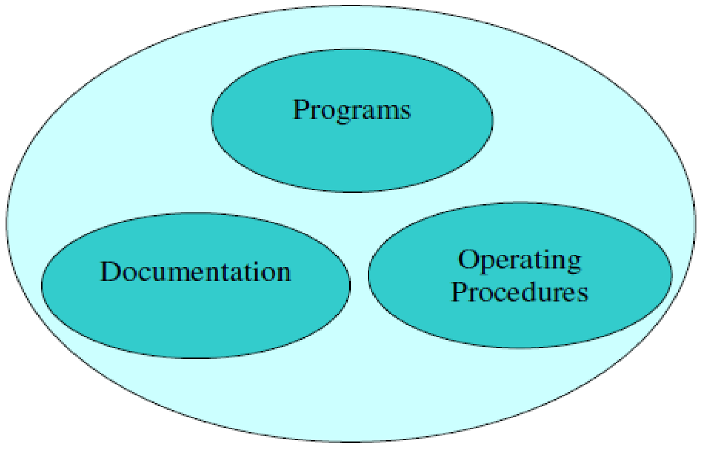

Figure 1.1: Software=Program+Documentation+Operating Procedures

Program Vs. Software

S. No	Program	Software
	A program is a set of instructions written in a programming language to perform a particular function.	Many programs combine together to form software. While **software** refers to a collection of several programs and other procedures and documentation.
	Programs are developed by individuals for their personal use, the programmer himself is the sole user.	Software are developed by a team, known as development team. Normally software users are not involved with the development.
	Programs are small in size and have limited functionality.	Software are extremely large.
	For a program, the user interface may not be	On the other hand, for a software product,

	very important, because the programmer is the sole user.	user interface must be carefully designed and implemented because developers of that product and users of that product are totally different.
	In case of a program, very little documentation is expected, in the form of comments.	Software product must be well documented
	A program can be developed according to the programmer's individual style of development.	a software product must be developed using the accepted software engineering principles.

Software Crisis:

- ➢ The era of electronic computers started in 1940s.
- ➢ The initial efforts of improvement were mainly focused on hardware.
- ➢ With the improvement in the field of electronics, hardware became very effective by 1960s.
- ➢ At this time, programming techniques available were not very effective and so, we were not exploiting hardware capabilities fully.
- ➢ The software development techniques were adhoc and programming-centered.
- ➢ These adhoc or programming-centered approach may work for small problems, but for the large and complex problems/projects, these techniques generally do not work.

As a result of this, computing world found itself in a crisis, which is known as ***"software crisis"***.

- ➢ If we have to control this software crisis, some scientific and systematic approach is needed for software development.
- ➢ To overcome this problem of software crisis, NATO science committee held two conferences in the 1960s in Europe. This is where the term '**software engineering**' was coined.
- ➢ The software crisis was a term used to describe the impact of rapid increases in computer power and the complexity of the problems which could be tackled.
- ➢ In essence, it refers to the difficulty of writing correct, understandable, and verifiable computer programs.
- ➢ The roots of the software crisis are complexity, expectations, and change.
- ➢ Various processes and methodologies have been developed over the last few decades to "tame" the software crisis, with varying degrees of success.

In general, software projects which are large, complicated, poorly-specified, and involve unfamiliar aspects, are still particularly vulnerable to large, unanticipated problems.

1968 NATO Conference on Software Engineering following points raised:

Problems with software:

- Often delivered too late
- Did not behave as user expects
- Rarely adaptable to changed circumstances
- Many errors detected after delivery
- Communication between stakeholders!

What are the reasons for the software crisis?

- Software complexity
- Failure to manage risk

Software Complexity

- Business software demands are increasing
- No one understands the entire system
- Legacy systems must be maintained, but the original developers are gone

Failure to Manage Risk

- The waterfall lifecycle can delay problem identification
- There is no proof that the system will run until late in the lifecycle
- The result is maximum risk

Managing the "Software Crisis"

Three dimensions to manage:

- Multiple programmers
- Application complexity
- Maintenance