Structured Programming Concepts

- **Structured Programming Approach**, can be defined as a programming approach in which the program is made as a single structure. It means that the code will execute the instruction by instruction one after the other.
- It doesn't support the possibility of jumping from one instruction to some other with the help of any statement like GOTO, etc. Therefore, the instructions in this approach will be executed in a serial and structured manner.

Structured Programming Concepts

- The structured program consists of well structured and separated modules. But the entry and exit in a Structured program is a single-time event. It means that the program uses single-entry and single-exit elements. Therefore a structured program is well maintained, neat and clean program.

Structured Programming Concepts

- The structured program mainly consists of three types of elements:
  - Sequence Statements
  - Selection Statements
  - Iteration Statements
- The first is sequencing, which has to do with the logical sequence provided by the statements in the program. As they are executed, each step in the sequence must logically progress to the next without producing any undesirable effects.

Structured Programming Concepts

- The second element is selection. This step allows the selection of any number of statements to execute in the program. These statements will contain certain keywords that can identify the sequence as a logically ordered executable. These terms are “if," “then," “endif," or “switch.“
- A third element is repetition. As a program proceeds, a statement continues to be active until the program gets to the point where some other action needs to take place. The keywords include "repeat," “for," or “do…until.”

Advantages of Structured Programming

- Easier to read and understand
- User Friendly
- Easier to Maintain
- Mainly problem based instead of being machine based
- Development is easier as it requires less effort and time
- Easier to Debug
- Machine-Independent, mostly.

Disadvantages of Structured Programming

- Since it is Machine-Independent, So it takes time to convert into machine code.
- The program depends upon changeable factors like data-types. Therefore it needs to be updated with the need on the go.

Programming Methodologies
Top-Down and Bottom-up Programming

- Program Design: **Program design** is the process of converting a set of requirements into a collection of commands or a program that can be executed on a computer system. A **program** is a series of instructions that the computer executes in order to perform some meaningful work.

Top-Down Programming

- **Top down** program design is an approach to program design that starts with the general concept and repeatedly breaks it down into its component parts. In other words, it starts with the abstract and continually subdivides it until it reaches the specific.
- The basic idea in top-down approach is to break a complex algorithm or a problem into smaller segments called modules, this process is also called as *modularization.* The modules are further decomposed until there is no space left for breaking the modules without hampering the originality.

Top-Down Programming

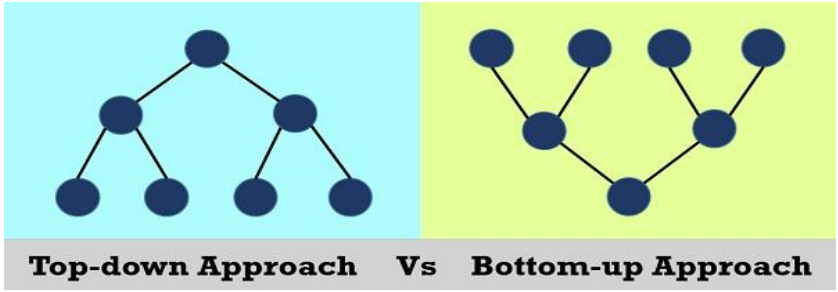
- Example: Consider creating the prime factorization of a number like 1540. The steps involved might look like:
- 1540
- 2 x 770
- 2 x 2 x 385
- 2 x 2 x 5 x 77
- 2 x 2 x 5 x 7 x 11

Bottom-up Programming

- **Bottom up** program design works in the exact opposite way. It starts with the component parts and repeatedly combines them to achieve the general concept. In other words, it starts with the specific and continually combines it until it reaches the abstract.
- In this approach we start working from the most basic level of problem solving and moving up in conjugation of several parts of the solution to achieve required results. The most fundamental units, modules and sub-modules are designed and solved individually, these units are then integrated together to get a more concrete base to problem solving.

Bottom-up Programming

- Example: Consider the factorization of 1540. For bottom up design the steps involved might look like:
- 2 x 2 x 5 x 7 x 11
- 2 x 2 x 5 x 77
- 2 x 2 x 385
- 2 x 770
- 1540



Top- Down	Bottom-up
Divides a problem into smaller units and then solve it.	Starts from solving small modules and adding them up together
This approach contains redundant information.	Redundancy can easily be eliminated.
A well-established communication is not required.	Communication among steps is mandatory.
The individual modules are thoroughly analyzed.	Works on the concept of data-hiding and encapsulation.
Structured programming languages such as C uses top down approach.	OOP languages like C++ and Java etc. uses bottom-up mechanism.
Relation among modules is not always required.	The modules must be related for better communication and work flow.