Structured Programming

Key Terms

control structures
: Mechanisms that allow us to control the flow of execution within a program.

sequence
: A control structure where the program executes the items in the order listed.

selection
: A control structure where the program chooses between two or more options.

iteration
: A control structure that allows some lines of code to be executed many times.

branching
: An uncontrolled structure that allows the flow of execution to jump to a different part of the program.

spaghetti code
: A phrase that is used for unstructured and difficult-to-maintain source code.

structured programming
: A method of planning programs that avoids the branching category of control structures.

- In early programming languages, like Fortran (first invented in 1954) and various low level machine languages, the goto statement allowed the computer to deviate from the sequential execution of the program instructions.
- The goto statement was recognized to be a very powerful construction, and soon, programs of increasing complexity and power were developed.

- However, the increasingly complex code that resulted from goto statements became harder and harder to maintain.

- The concept of **structured programming** started in the late 1960's with an article by Edsger Dijkstra. He proposed a "go to less" method of planning programming logic that eliminated the need for the branching category of control structures. He showed that any program construction that could be created with goto statements could be created more simply with the sequence, repetition and decision control structures. The topic was debated for about 20 years. But ultimately – "By the end of the 20th century nearly all computer scientists were convinced that it is useful to learn and apply the concepts of structured programming."

Structured programming is a programming paradigm aimed at improving the clarity, quality, and development time of a computer program by making extensive use of the structured control flow constructs of selection (if/then/else) and repetition (while and for), block structures, and subroutines in contrast to using simple tests and jumps such as the go to statement, which can lead to "**spaghetti code**" that is potentially difficult to follow and maintain.

One of the most important concepts of programming is the ability to control a program so that different lines of code are executed or that some lines of code are executed many times. The mechanisms that allow us to control the flow of execution are called **control structures**. There are three main categories of control structures:

- **Sequence** – Execute a list of statements in order.

Selection – Choose at most one action from several alternative conditions. This is where you select or choose between two or more

flows. The choice is decided by asking some sort of question. The answer determines the path (or which lines of code) will be executed.

- **Iteration** – Also known as repetition, it allows some code (one/many lines) to be executed (or repeated) several times. The code might not be executed at all (repeat it zero times), executed a fixed number of times or executed indefinitely until some condition has been met. It is also known as looping.

  A fourth category describes unstructured code.

- **Branching** – An uncontrolled structure that allows the flow of execution to jump to a different part of the program by the use of goto statment. This category is rarely used in modular structured programming.

  All high-level programming languages have control structures. All languages have the first three categories of control structures (sequence, selection, and iteration). Most have if-then-else structure (which belongs to the selection category) and the while structure (which belongs to the iteration category). After these two basic structures, there are usually language variations.

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 like the execution of the program starts with main() function. Therefore, a structured program is well maintained, neat and clean program. This is the reason why the Structured Programming Approach is well accepted in the programming world.

Advantages of Structured Programming Approach:

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

Disadvantages of Structured Programming Approach:

1. Since it is Machine-Independent, So it takes time to convert into machine code.
2. The program depends upon changeable factors like data-types. Therefore it needs to be updated with the need on the go.
3. Usually the development in this approach takes longer time as it is language-dependent. Whereas in the case of assembly language, the development takes lesser time as it is fixed for the machine.