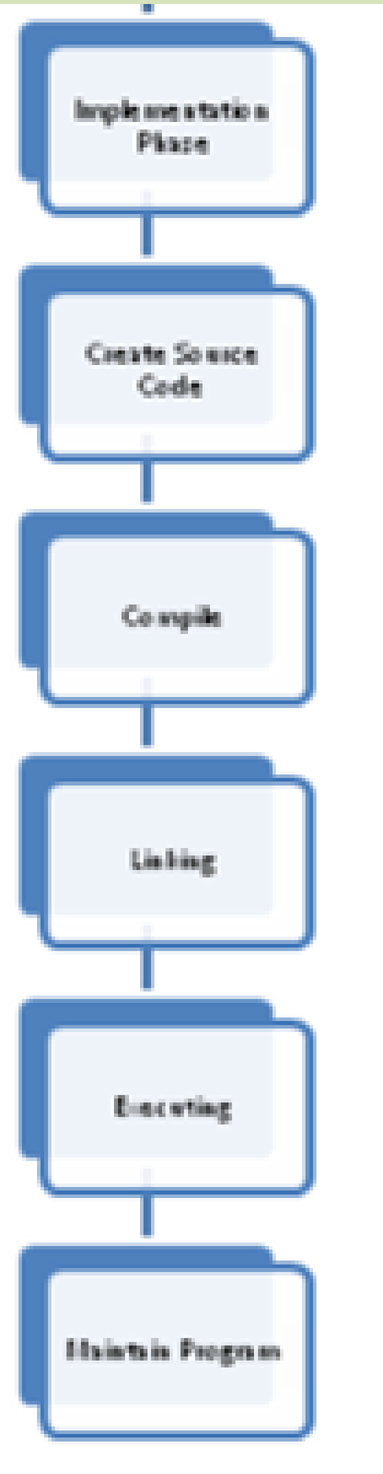
PROGRAM IMPLEMENTATION

The Implementation Phase Steps:



1. Translate the Algorithm into a Specific Programming Language

This involves translating the algorithm into a programming language; initially manually on paper and then on the computer. A text editor is used to enter the programming code into the computer. The source code is stored in a file; in the case of Pascal the file extension is .pas.

2. Compile the Source Code

This is the process when the compiler translates the source code to object code. Object code is the machine language equivalent of the source code. During compilation the syntax of the source code is checked to ensure conformity with the rules of the programming language. Syntax errors result in an incomplete compilation. The errors must be corrected and the program re-compiled**.**

Types of Errors

Syntax Error: An error which breaks one of the rules of a given programming language. For example, in Pascal the omission of the semicolon at the end of a statement: sum:= x + y The compiler usually issues messages which identify the location of these errors.

Logic Error: An error which has the correct syntax but does not give the correct result. There is a fault in the logic of the program. For example: sum := A - B;

Run-time Error: An error that occurs when the program is executed. For example, division by zero or referencing an array location that is outside of the legitimate bounds of the array.

```
Number :=1;
Number:= Number-1;
Answer:= Total/Number; {Division By Zero Example}
```

NB.

Debugging: The process of locating and correcting errors that occur in a program.

3. Linking to Modules

The compiled program must be combined with other system modules to form an executable image that can be loaded into memory. Linking is the final stage before a program can be executed where the special operations' code is added.

NB.

Loading is process of reading a program from disk storage into memory.

4. Program Execution

The program is sent to the CPU. The control unit interprets each instruction and passes it to the appropriate unit for execution. During execution, if any run-time errors are detected the program will terminate prematurely.

NB.

Executing is when the instructions in the program are carried out.

5. Maintaining the Program

This involves making periodic modifications to the program when the requirements change. For example, VAT may change from 15% to 17.5%.

CHARACTERISTICS OF GOOD PROGRAMS

- The program should produce the correct results at all times.
- The program is free of errors.
- The program is well documented (Internal documentation :mnemonic identifier names, comments, indentation, white space. External documentation: user manuals).
- The program is maintainable. A program can be maintain by: (a) dividing it into modules; (b) using constants; (c) properly documenting the program.

PROGRAM DOCUMENTATION

Reasons for Documentation:

- Makes it easy for users to understand the program
- If there is a problem with a section of the program users can refer to the documentation to solve it
- Makes it easy for other programmers to modify the code

Types of Documentation:

Internal (appears inside the program)

- Use of meaningful variable names.
- Descriptive comments to explain code.
- Indentation of code to improve readability.
- Effective use of white spaces to improve readability.

External (Created separately and supplied with the program)

Types of External Documentation

Technical

- How to install the program
- Software and Hardware specifications, e.g. operating system, memory and hard disk requirements
- Program problem
- Algorithm
- Program code
- Test Data
- Data and file structures

User

- How to operate the program, e.g. user manual
- How to solve simple problems