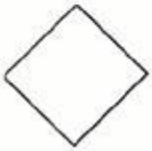
Flowcharts are written with program flow from the top of a page to the bottom. Each command is placed in a box of the appropriate shape, and arrows are used to direct program flow. The following shapes are often used in flowcharts:

An oval indicates beginning or end of a program.

A parallelogram is a point where there is input to or ouput from the program.

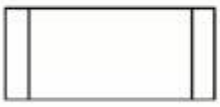
A rectangle indicates the assignment of a value to a variable, constant, or parameter. the assigned value can be the reult of a computation. The computation would also be included in the rectangle.



A diamond indicates a point where a decision is made.

An open-ended rectangle contains comment statements. The comment is connected to the program flow via a dashed line.

A hexagon indicates the beginning of a repitition.



The double-lined rectangle indicates the use of an algorithm specified outside the program, such as a subroutine.

Circles can be used to combine flow lines.

Arrows indicate the direction and order of program execution.

Pseudocode is a method of describing computer algorithms using a combination of natural language and programming language. It is essentially an intermittent step towards the development of the actual code. It allows the programmer to formulate their thoughts on the organization and sequence of a computer algorithm without the need for actually following the exact coding syntax. Although pseudocode is frequently used there are no set of rules for its exact implementation. In general, here are some rules that are frequently followed when writing pseudocode:

- The usual Fortran symobols are used for arithmetic operations (+, -, *, / , **).
- Symbolic names are used to indicate the quantities being processed.
- Certain Fortran keywords can be used, such as PRINT, WRITE, READ, etc.
- Indentation should be used to indicate branches and loops of instruction.

Here is an example problem, including a flowchart, pseudocode, and the final Fortran 90 program. This problem and solution are from Nyhoff, pg 206:

For a given value, *Limit*, what is the smallest positive integer *Number* for which the sum

Sum = 1 + 2 + ... + *Number*

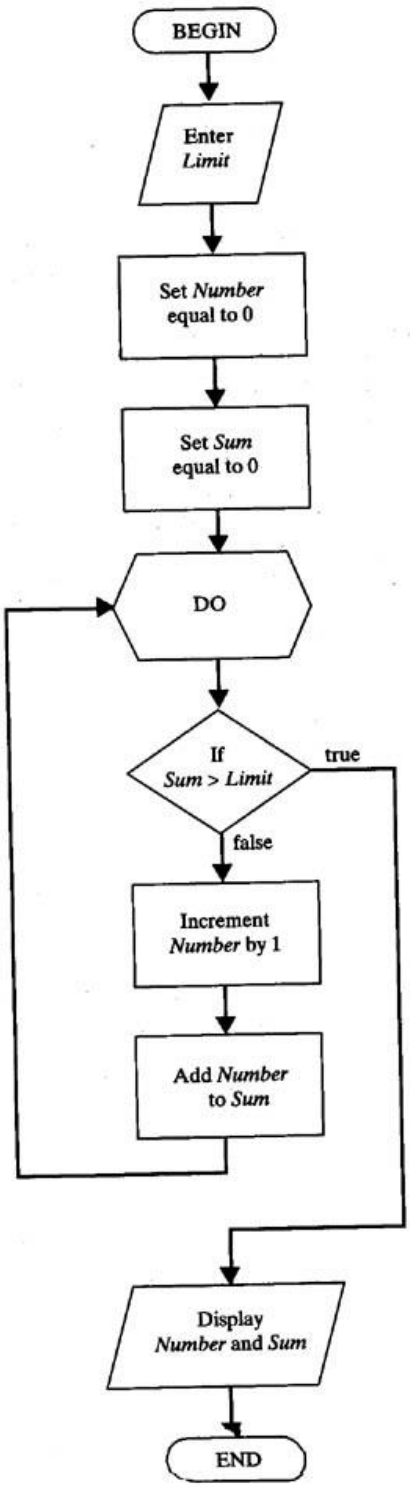
is greater than *Limit*. What is the value for this *Sum*?

Pseudocode:

Input: An integer *Limit*
Ouput: Two integers: *Number* and *Sum*

1. Enter *Limit*
2. Set *Number* = 0.
3. Set *Sum* = 0.
4. Repeat the following:
   a. If *Sum* > *Limit*, terminate the repitition, otherwise.
   b. Increment *Number* by one.
   c. Add *Number* to *Sum* and set equal to *Sum*.
5. Print *Number* and *Sum*.

Flowchart:


BEGIN
Enter
Limit
Set Number
equal to 0
Set Sum
equal to 0
DO
If
Sum > Limit
true
false
Increment
Number by 1
Add Number
to Sum
Display
Number and Sum
END

Benefit:

Flowcharts are especially beneficial for smaller concepts and problems, while pseudocode is more efficient for larger programming problems. Flowcharts provide an easy method of communication about the logic and offer a good starting point for the project because they are easier to create than pseudocode in the beginning stages.

Pseudocode provides a beneficial bridge to the project code because it closely follows the logic that the code will. Pseudocode also helps programmers share ideas without spending too much time creating code, and it provides a structure that is not dependent on any one programming language.

REFERENCES:

Jackson, G.S., et al. “Differences Between Psuedocode and Flowcharts.” *Techwalla*, https://www.techwalla.com/articles/differences-between-psuedocode-and-flowcharts.

FOR3_1.Html, http://www.owlnet.rice.edu/~ceng303/manuals/fortran/FOR3_3.html.