

Chapter  
1

**OVERVIEW**

*Get on the  
Fast Track!*



TM

**SYS-ED/  
COMPUTER  
EDUCATION  
TECHNIQUES, INC.**

**Objectives**

You will learn:

- C Structured programming versus spaghetti code.
- C Three basic logic structures.
- C Structured flowchart and structured code.
- C Benefits of structured programming.
- C Structured design.
- C Structured walk-through.
- C Structured flowcharts.
- C Three basic nodes used in structured flowcharts.
- C Three basic logic structures of structured programming.
- C Pseudocode.

---

## 1 Structured Programming vs. Spaghetti Code

Traditional "spaghetti-bowl" fashion of programming:

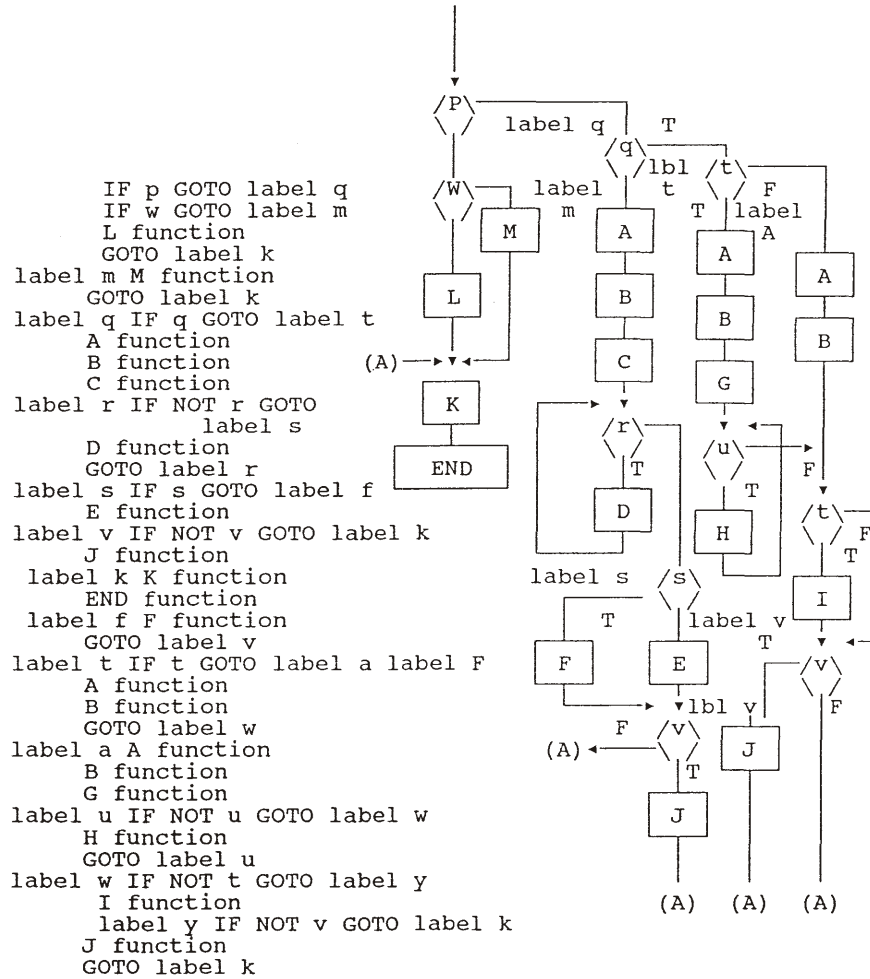
- C Excessive use of unconditional branches (**GO TOs**) and statement labels.

Definitions of structured programming:

- C The design, writing and testing of a program made up of interdependent parts in a definite pattern of organization.
- C A programming technique for developing source code logic that is understandably and verifiably correct; specifically, a technique that employs a top-down refinement strategy to produce code built from a small set of logical constructs (chiefly, the sequence construct, the decision construct, and the loop construct).

## 2 "Spaghetti" Code and Traditional (Unstructured) Flow-chart: Example

(from IBM Independent Study Program, Structured Programming)



---

### 3 Three Eras of COBOL Program Development

Three eras of COBOL program development.

C **GO TO** programming (c. 1965)

- Program designed using detailed program flowchart.
- Extensive use of **GO TO** statements.
- COBOL code difficult to read and modify.

C Modular programming (c. 1970)

- Sought to divide a program into a number of independent modules: one mainline and one or more subroutines.
- Increased use of **PERFORM** statements, but continued use of **GO TO** statements.

C Structured programming (c. 1975)

- Program designed using structure chart or other design document instead of a program flowchart.
- Use of three basic logic structures.
- Minimum use or total absence of **GO TO** statements.

---

## 4 Three Basic Logic Structures

Structured programming has three basic logic structures:

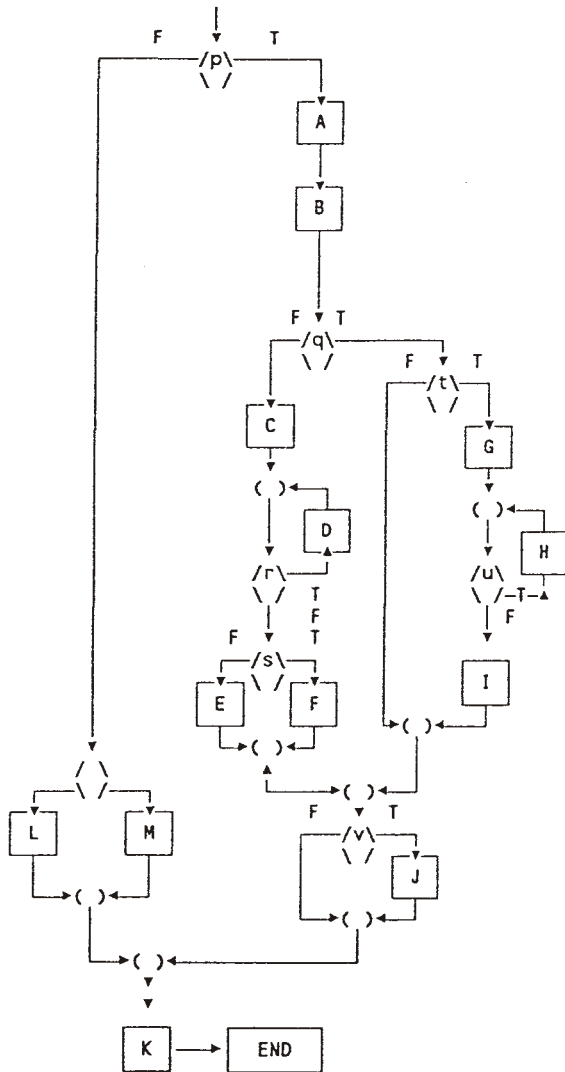
- C Sequence
- C Selection (**IF...ELSE**)
- C Iteration (**PERFORM...UNTIL**)

These three basic logic structures can be combined in a limitless variety of ways to produce any required logic.

- C Entire program is a set of nested modules, each of which has one entry and one exit.
- C Discourages use of the **go to** statement.
- C Includes certain programming restrictions and conventions (convenient, though not necessary for structured programming).
- C Proper formatting (indentation).
- C Break a program down into relatively small modules (one-page modules generally recommended).
- C No module allowed to modify any other module.
- C Use of top-down program design.

## 5 Structured Flowchart and Structured Code: Example

(from IBM Independent Study Program, Structured Programming)



LEVELS OF  
INDENTATION  
1-4

```

IF p THEN
  A function
  B FUNCTION
  IF q THEN
    IF t THEN
      G function
      DOWHILE U
      H function
      ENDDO
      I function
    (ELSE)
  ENDIF
ELSE
  C function
  DOWHILE r
  D function
  ENDDO
  IF s THEN
    F function
  ELSE
    E function
  ENDIF
ENDIF
IF v THEN
  J function
(ELSE)
ENDIF
ELSE
  IF w THEN
    M Function
  ELSE
    L FUNCTION
  ENDIF
ENDIF
K function
END function
    
```

---

**6 Improved Programming Technologies**

Structured programming is one of several "improved programming technologies":

- C Team operations.
- C Top down program development.
- C Code reading and walk-throughs.
- C HIPO (Hierarchy plus Input, Process, Output) documentation technique.
- C Program development support libraries.

Combined use of these techniques has been proven to increase data processing efficiency.

---

**7 Benefits of Structured Programming**

The benefits of structured programming include:

- C Produces programs that are easier to understand and maintain.
- C Easier to test and debug.
- C Greater programmer productivity.
- C Greatly reduces the need for explanatory comments.

---

## **8 History of Structured Programming**

1966 - Corrado Bohm and Jacopini proved that any program logic, no matter how complex, could be expressed using relatively simple structured program control elements.

Dijkstra, the leading name in structured programming, devised a set of ideas and a series of examples for clear thinking in the construction of programs.

Mills of IBM provided mathematical assurance that the technical standards in programming procedures are sound and practical.

- C Mills pointed out psychological effect of structured programming which accompanies the technical benefits.
- C Programmers encouraged by their new power to write programs correctly.
  - Minimizes errors of carelessness.

---

## 9 Structured Design

Definitions of Structured Design:

- C A set of program design techniques and guidelines that are used to specify the functions in a program solution, the data processed by each function and the relationships among the functions.
  
- C The development of a blueprint of a computer system solution to a problem, having the same components and inter-relationships among the components as the original problem has.

Objective:

- C To define program solutions using single, well-defined functions that can be easily programmed, tested and maintained.
  
- C Structured design is **not** modular programming or top-down design, but rather an extension of the two techniques.

---

**10 Structured Walk-through**

Series of reviews of a programmer's work by his/her peer group, each with different objectives and each occurring at a different time in the application development cycle.

Goes hand in hand with top-down design, coding and testing.

Early error detection.

- C During the design phase.
- C After programmer has received analyst's specifications and developed the logic to implement the solution.
- C During the coding stage.

Tracks progress in the application development cycle.

Roles: reviewee, reviewers, moderator and secretary.

Criteria for successful walk-throughs:

- C Structured walk-throughs are used to find programming problems, not to evaluate programmers.
- C Emphasis on error detection, not correction.
- C Everyone's work is reviewed.

---

## 11 Structured Flowcharts

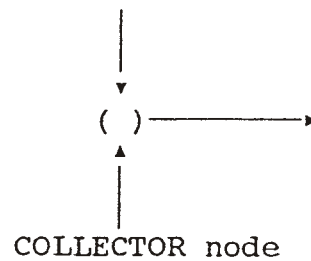
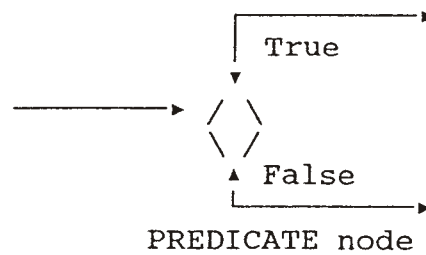
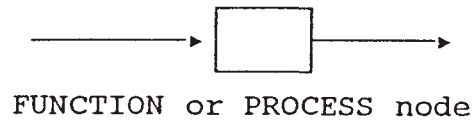
Structured flowcharts are:

- C Pictorial representation of structured program logic.
- C Valuable tool for the programmer, but not a good documentation technique because they are seldom updated.

Structured flowcharts use only three basic elements or "nodes":

- C FUNCTION (or PROCESS) node.
  - Represents a function or some sort of data transformation.
- C PREDICATE node.
  - Represents a test on data, resulting in either a true or false condition.
- C COLLECTOR node.
  - Represents a junction which always has two entries and one exit.
  - Needed only with **PERFORM...UNTIL** statement and to rejoin the two paths of a decision box.

### 11.1 Three Basic Nodes Used in Structured Flowcharts

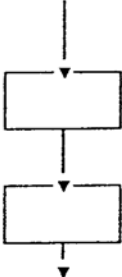


The three basic nodes are combined into the three basic logic structures of structured programming (sequence, selection and iteration).

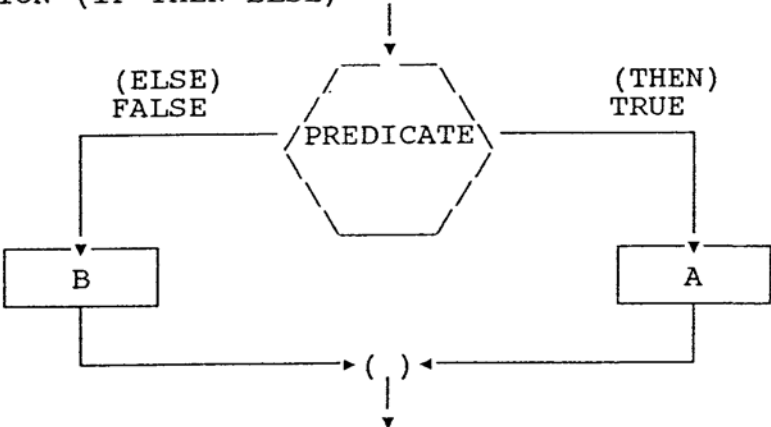
- Any "proper" program can be written using only the three basic nodes.
- A "proper" program is one in which:
  - there is precisely one entry point and one exit.
  - for every node, there exists a path from the input line through that node to the output line (i.e., no infinite loops or dead code).
- Being replaced by pseudocode.

12 Three Basic Logic Structures of Structured Programming

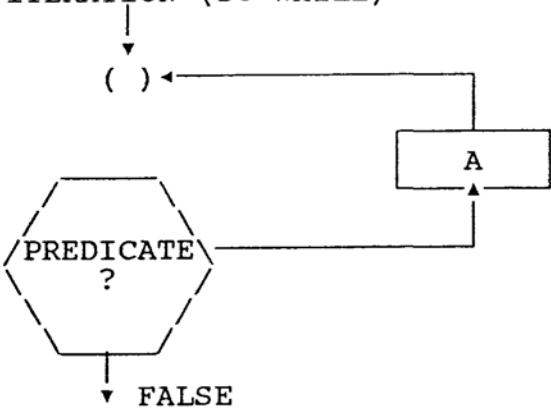
(a) SEQUENCE



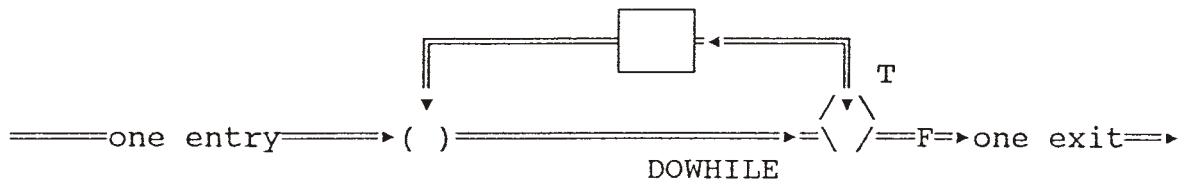
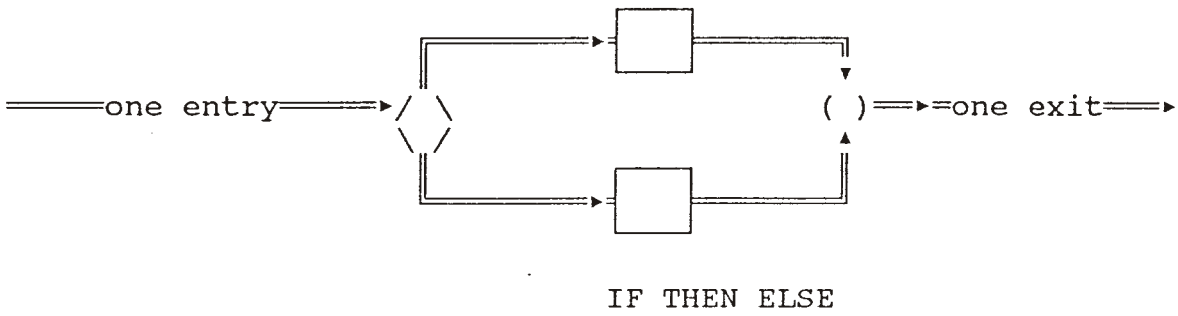
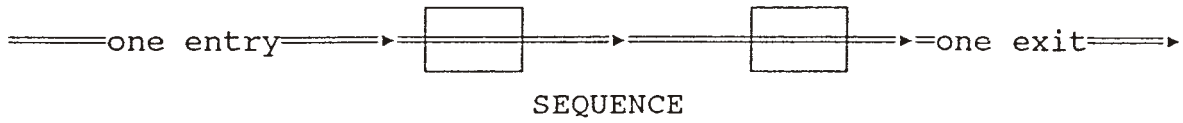
(b) SELECTION (IF THEN ELSE)



(c) ITERATION (DO WHILE)



13 Fundamental Nodes as  
Used in Proper Programs



---

## 14 Pseudocode

The function and purpose of pseudocode:

- C Design aid and/or documentation technique associated with structured programming.
- C Uses instructions similar to those of a computer language to describe program logic.
- C No precise syntactical rules or rules for indentation.
- C Indentation used to represent levels of hierarchy for various "nested" functions; makes logical structure clearer.
  - To pair **dowhile** with **enddo**, **if** with **else** and **endif**
- C Functional equivalent of a flowchart.
- C Allows the programmer to concentrate on logic instead of form.
- C Easier to update than a flowchart.

---

## 14.1 Pseudocode: Example

```
1      Initialize
2      Open Files
3      Initial reads for OLD-MASTER and TRANSACTION files
4  +))) PERFORM until no more data on both files
5  *  +)) IF OLD-MASTER social security number < TRANSACTION social security number
6  *  *  +)) PERFORM
7  *  *  *      Copy old master record to new master record
8  *  *  *      Set switch to read OLD-MASTER only
9  *  *  .))) ENDPERFORM
10 *  *      ELSE IF OLD-MASTER social security number = TRANSACTION social security number
11 *  *  +)) PERFORM
12 *  *  *      IF addition-write error indicating duplicate addition
13 *  *  *      ELSE IF deletion-delete old master record
14 *  *  *      ELSE IF salary-change-do salary update
15 *  *  *      ENDIF
16 *  *  *      Set switches to read both OLD-MASTER and TRANSACTION files
17 *  *  .)) ENDPERFORM
18 *  *      ELSE IF OLD-MASTER social security number > TRANSACTION social security number
19 *  *  +)) PERFORM
20 *  *  *      IF addition-add record to NEW-MASTER-FILE
21 *  *  *      ELSE write error indicating no match
22 *  *  *      ENDIF
23 *  *  *      Set switch to read TRANSACTION-FILE only
24 *  *  .)) ENDPERFORM
25 *  .)) ENDIF
26 *      Read OLD-MASTER and/of TRANSACTION files as appropriate
27 .)) ENDPERFORM
28      Close files
29      Stop run
```