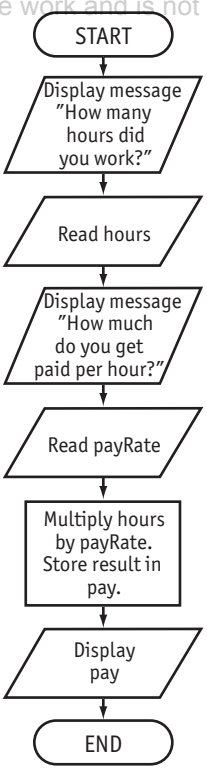
APPENDIX C

Introduction to Flowcharting

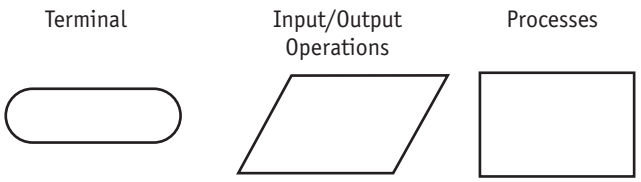
This appendix provides a brief introduction to flowcharting. It includes example flowcharts for programs that appear in Chapters 1 through 6.

NOTE: One of the best tools currently available for creating flowcharts is Flowgorithm, available for free at www.flowgorithm.org. Flowgorithm allows you to create interactive flowcharts that can be executed as programs. It also generates source code in several programming languages, including C++.

A flowchart is a diagram that depicts the "flow" of a program. It contains symbols that represent each step in the program. The figure shown here is a flowchart for Program 1-1, the pay-calculating program in Chapter 1.



Notice there are three types of symbols in this flowchart: rounded rectangles (representing terminal points), parallelograms (representing input/output operations), and a rectangle (representing a process).

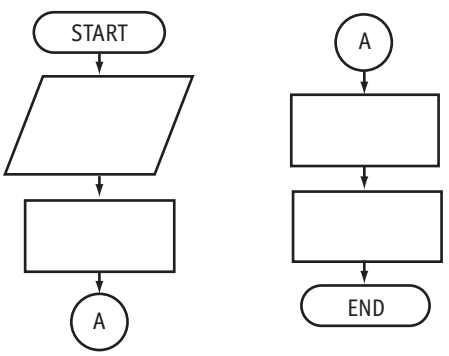


The rounded rectangles, or terminal points, indicate the flowchart's starting and ending points. The parallelograms designate input or output operations. The rectangle depicts a process such as a mathematical computation, or a variable assignment. Notice the symbols are connected with arrows that indicate the direction of program flow.

Connectors

Sometimes a flowchart is broken into two or more smaller flowcharts. This is usually done when a flowchart does not fit on a single page, or must be divided into sections. A connector symbol, which is a small circle with a letter or number inside it, allows you to connect two flowcharts.

In the figure below, the A connector indicates that the second flowchart segment begins where the first flowchart segment ends.



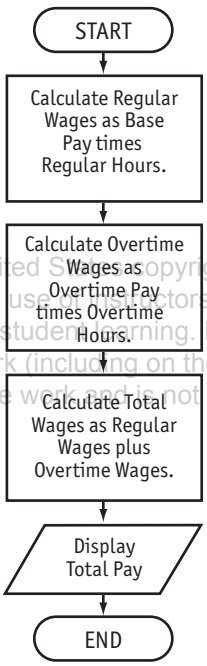
Flowchart Structures

There are four general flowchart structures:

- Sequence
- Decision
- Repetition
- Case

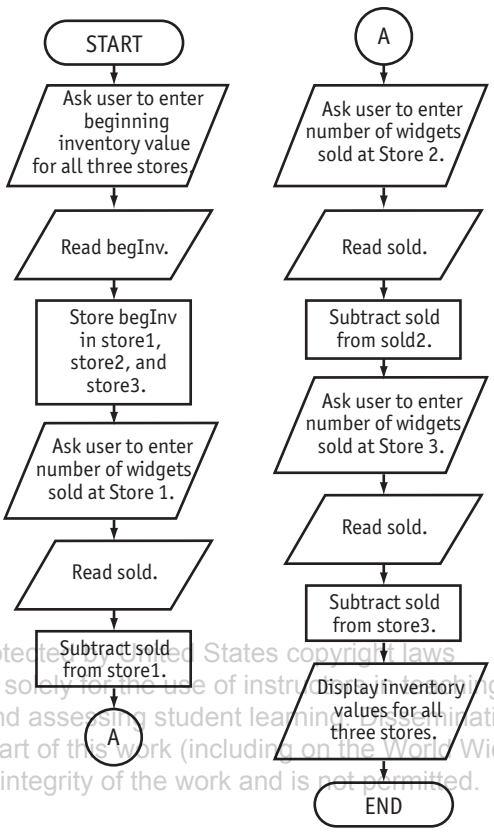
A sequence structure is a series of actions or steps, performed in order. The flowchart for the pay-calculating program is an example of a sequence structure. The following flowchart is also a sequence structure. It depicts the steps performed in Program 2-21, from Chapter 2.

Flowchart for Program 2-21



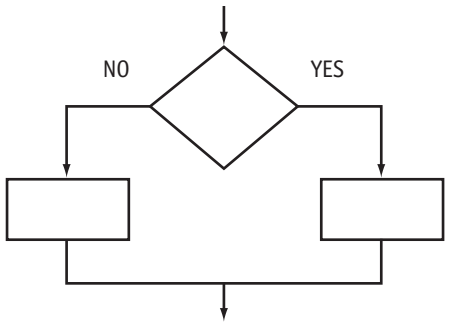
The following flowchart, which is another sequence structure, depicts the steps performed in Program 3-11 (from Chapter 3). Notice the use of connector symbols to link the two flowchart segments.

Flowchart for Program 3-11

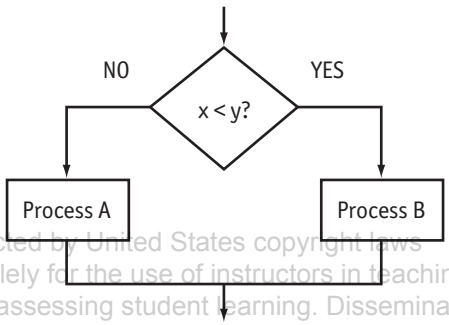


The Decision Structure

In a decision structure, one of two possible actions is performed, depending on a condition. In a decision structure, a new symbol, the diamond, represents a yes/no question. If the answer to the question is "yes," the program flow follows one path. If the answer to the question is "no," the program flow follows another path. The following figure shows the general form of a decision structure:

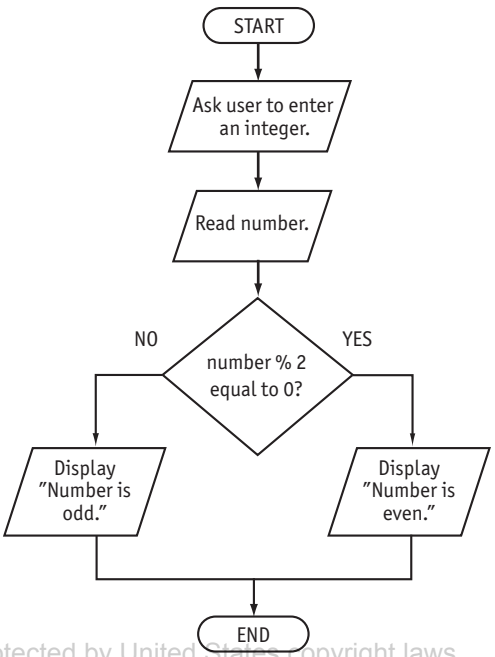


In the following flowchart segment, the question "is x < y?" is asked. If the answer is "no," then process A is performed. If the answer is "yes," then process B is performed.



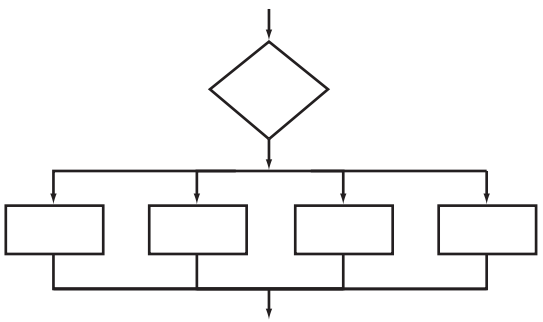
The following flowchart depicts the logic of Program 4-8, from Chapter 4. The decision structure shows that one of two possible messages is displayed on the screen, depending on the value of the expression `number % 2`.

Flowchart for Program 4-8



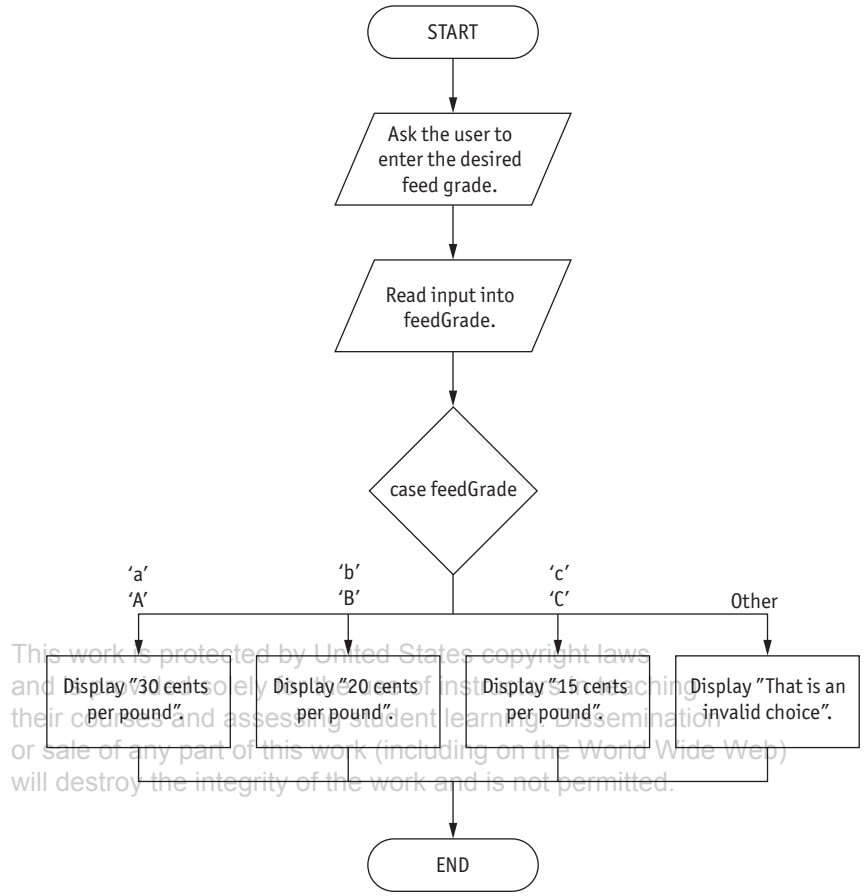
The Case Structure

In a case structure, one of several possible actions is taken, depending on the contents of a variable. The following flowchart segment shows the general form of a case structure:



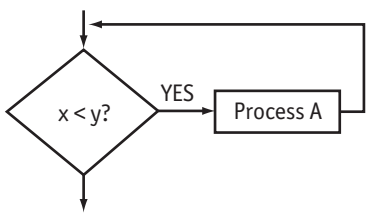
The following flowchart depicts the logic of Program 4-26, which is also from Chapter 4. One of four possible paths is followed, depending on the value stored in the variable `feedGrade`.

Flowchart for Program 4-26



Repetition Structures

A repetition structure represents part of the program that repeats. This type of structure is commonly known as a loop. The following figure shows an example of a repetition structure:

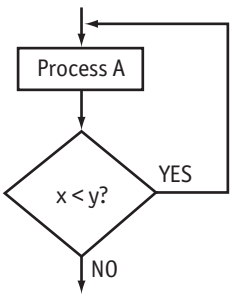


Notice the use of the diamond symbol. A repetition structure tests a condition, and if the condition exists, it performs an action. Then it tests the condition again. If the condition still exists, the action is repeated. This continues until the condition no longer exists. In the flowchart segment above, the question "is x < y?" is asked. If the answer is yes, then Process

A is performed. The question "is x < y?" is asked again. Process A is repeated as long as x is less than y. When x is no longer less than y, the repetition stops and the structure is exited.

There are two forms of repetition structure: pre-test and post-test. A pre-test repetition structure tests its condition *before* it performs an action. The flowchart segment above shows a pre-test structure. Notice Process A does not execute at all if the condition "x < y" is not true. The pre-test repetition structure is coded in C++ as a `while` loop.

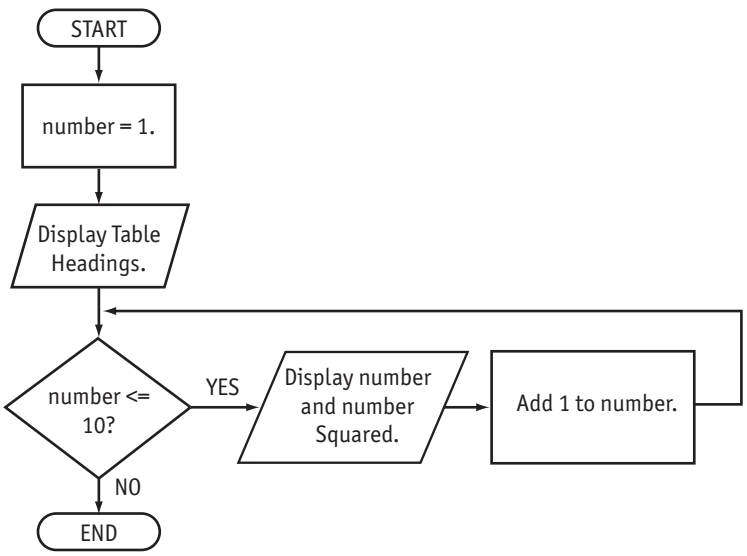
A post-test repetition structure tests its condition *after* it performs an action. This type of loop always performs its action at least once. The following flowchart segment shows an example:



The post-test repetition structure is coded in C++ as a `do-while` loop.

The following flowchart depicts the logic of Program 5-6, which appears in Chapter 5.

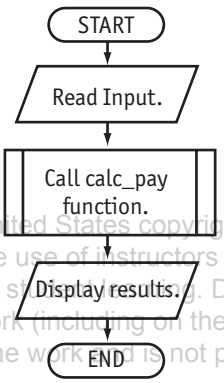
Flowchart for Program 5-6



Modules

A program module (such as a function in C++) is represented by the special symbol shown below:

The position of the module symbol indicates the point the module is executed, as shown in the following figure:



A separate flowchart can then be constructed for the module. The following figure shows a flowchart for Program 6-19 from Chapter 6. The `getBasePay` and `getOvertimePay` modules appear as separate flowcharts.

Flowchart for Program 6-19