Programming Logic and Design

Seventh Edition

Chapter 7

File Handling and Applications

Objectives

In this chapter, you will learn about:

- Computer files
- The data hierarchy
- Performing file operations
- Sequential files and control break logic
- Merging files
- Master and transaction file processing
- Random access files

Understanding Computer Files

- **Computer file**
  - A collection of data stored on **permanent storage devices** such as DVDs, USB drives, and reels of magnetic tape
  - **Text files** (numbers, names, salaries) that can be read by a text editor
  - **Binary files** (images and music)
- File size measured in bytes
  - **Byte** (one character), **kilobyte** (thousands of bytes), **megabyte** (millions of bytes), **gigabyte** (billions of bytes), terabyte (trillions of bytes)

Understanding Computer Files (continued)

- **Organizing files**
  - **Directories and folders**
    - Organization units on storage devices
  - **Path**
    - Combination of disk drive plus the complete hierarchy of directories
    - Example: `C:\Logic\SampleFiles\PayrollData.dat`

Understanding the Data Hierarchy

- **Data hierarchy**
  - Describes the relationships between data components
  - Consists of:
    - **Characters**
    - **Fields**
    - **Records**
    - **Files**

Performing File Operations

- Use data files in your programs
  - **Declaring a file**

```
InputFile employeeData
OutputFile updatedData
```

  - **Opening a file**

```
open employeeData "EmployeeData.dat"
```

  - **Reading data from a file**

```
input name from employeeData
input address from employeeData
input payRate from employeeData
```

Performing File Operations (continued)

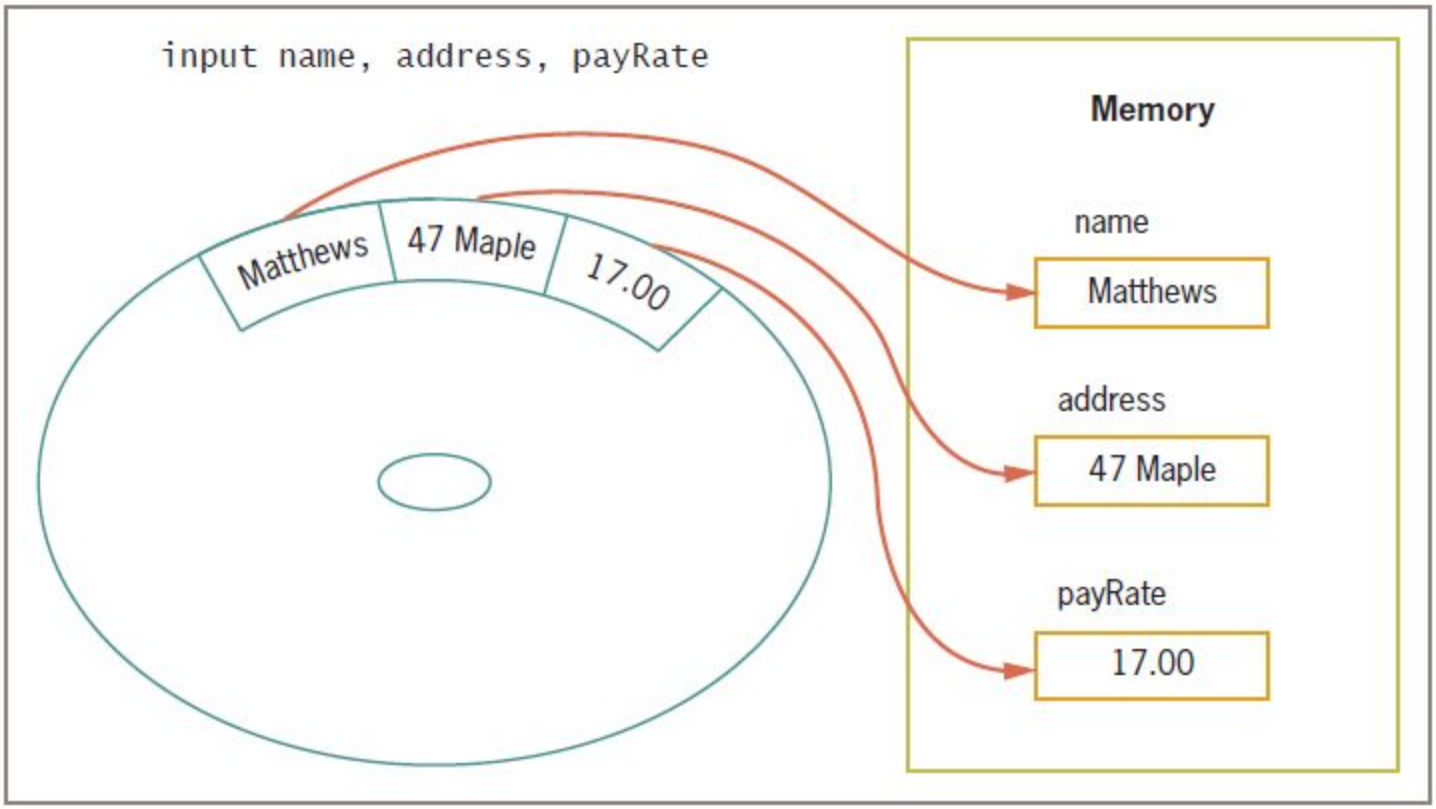


Figure 7-2 Reading three data items from a storage device into memory

Performing File Operations (continued)

- **Writing data to a file**

```
output name, address, payRate to employeeData
```

- **Closing a file**
  - Always close every file you open

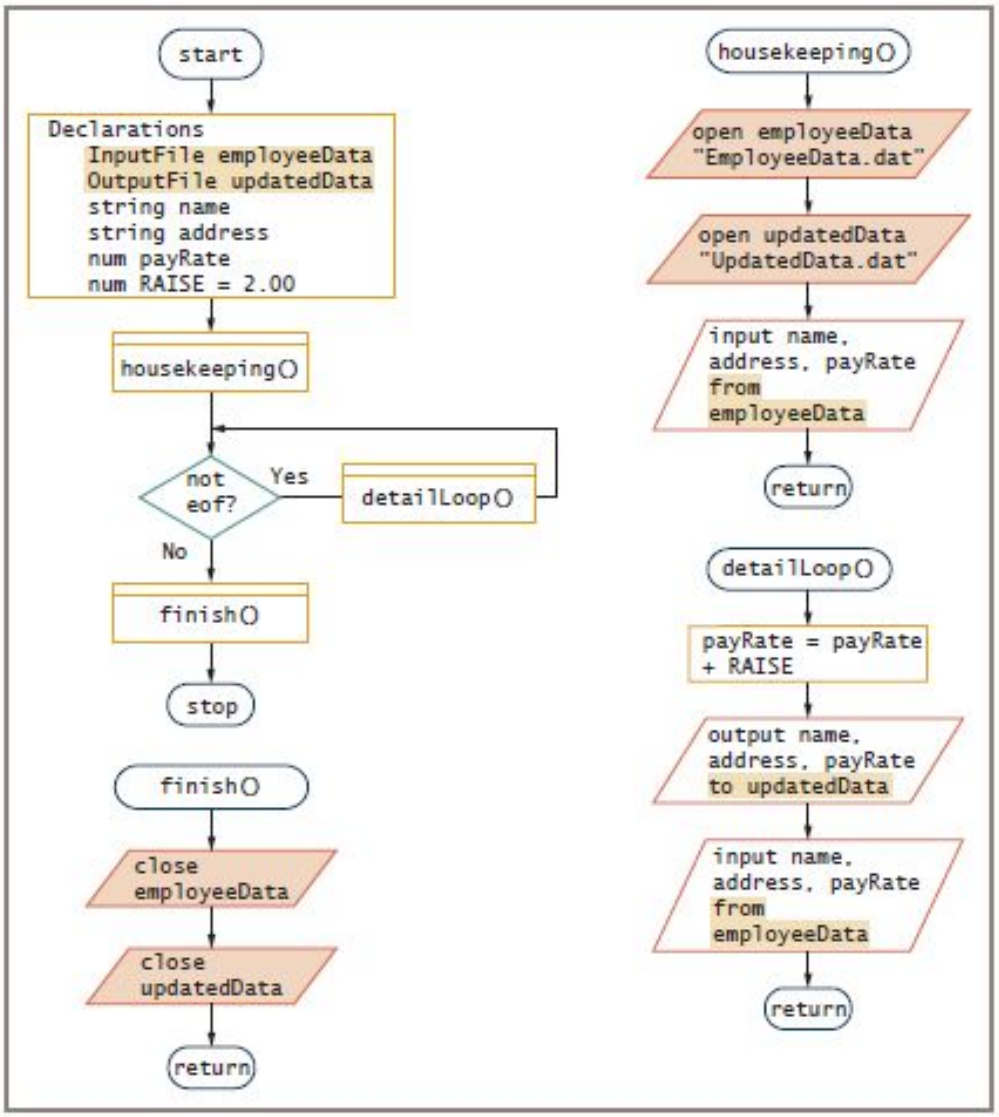


Figure 7-3 Flowchart and pseudocode for a program that uses files

```
start
   Declarations
      InputFile employeeData
      OutputFile updatedData
      string name
      string address
      num payRate
      num RAISE = 2.00
   housekeeping()
   while not eof
      detailLoop()
   endwhile
   finish()
stop

housekeeping()
   open employeeData "EmployeeData.dat"
   open updatedData "UpdatedData.dat"
   input name, address, payRate from employeeData
return

detailLoop()
   payRate = payRate + RAISE
   output name, address, payRate to updatedData
   input name, address, payRate from employeeData
return

finish()
    close employeeData
    close updatedData
return
```

Figure 7-3 Flowchart and pseudocode for a program that uses files (continued)

A Program that Performs File Operations

- **Backup file**
  - A copy that is kept in case values need to be restored to their original state
  - Called a **parent file**
  - A newly revised copy is a **child file**
- **Sorting**
  - The process of placing records in order by the value in a specific field or fields

Understanding Sequential Files and Control Break Logic

- **Sequential file**
  - Records are stored one after another in some order
- Understanding control break logic
  - A **control break** is a temporary detour in a program
  - A **control break program** uses a change in a value to initiate special actions or processing
  - A **control break report** groups similar data together
    - Input records must be in sequential order

Understanding Sequential Files and Control Break Logic (continued)

Company Clients by State of Residence

Name	City	State	
Albertson	Birmingham	Alabama	
Davis	Birmingham	Alabama	
Lawrence	Montgomery	Alabama	
		Count for Alabama	3
Smith	Anchorage	Alaska	
Young	Anchorage	Alaska	
Davis	Fairbanks	Alaska	
Mitchell	Juneau	Alaska	
Zimmer	Juneau	Alaska	
		Count for Alaska	5
Edwards	Phoenix	Arizona	
		Count for Arizona	1

Figure 7-4 A control break report with totals after each state

Understanding Sequential Files and Control Break Logic (continued)

- Examples of control break reports
  - All employees listed in order by department number, with a new page started for each department
  - All books for sale in a bookstore listed in order by category (such as reference or self-help), with a count following each category of book
  - All items sold in order by date of sale, with a different ink color for each new month

Understanding Sequential Files and Control Break Logic (continued)

- **Single-level control break**
  - A detour based on the value of a single variable
  - Uses a **control break field** to hold the previous value

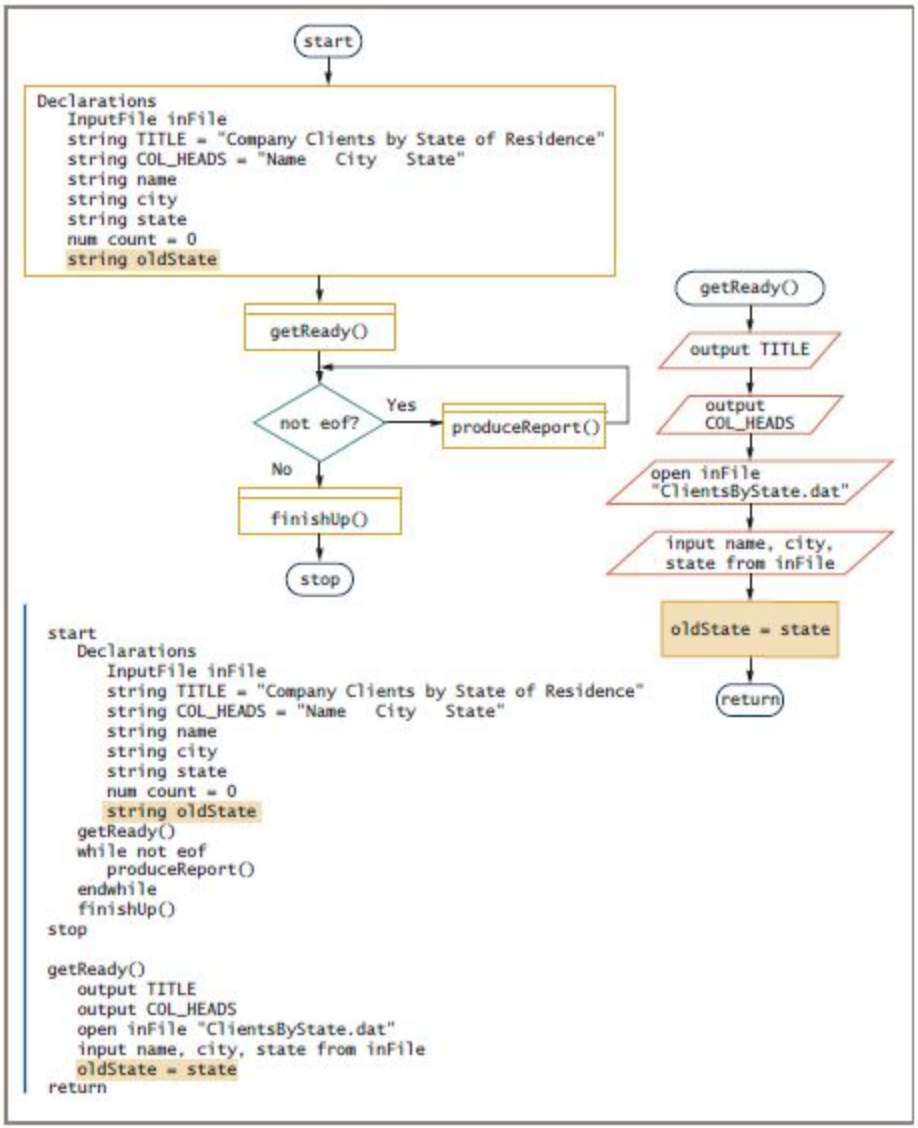


Figure 7-5 Mainline logic and `getReady()` module for the program that produces clients by state report

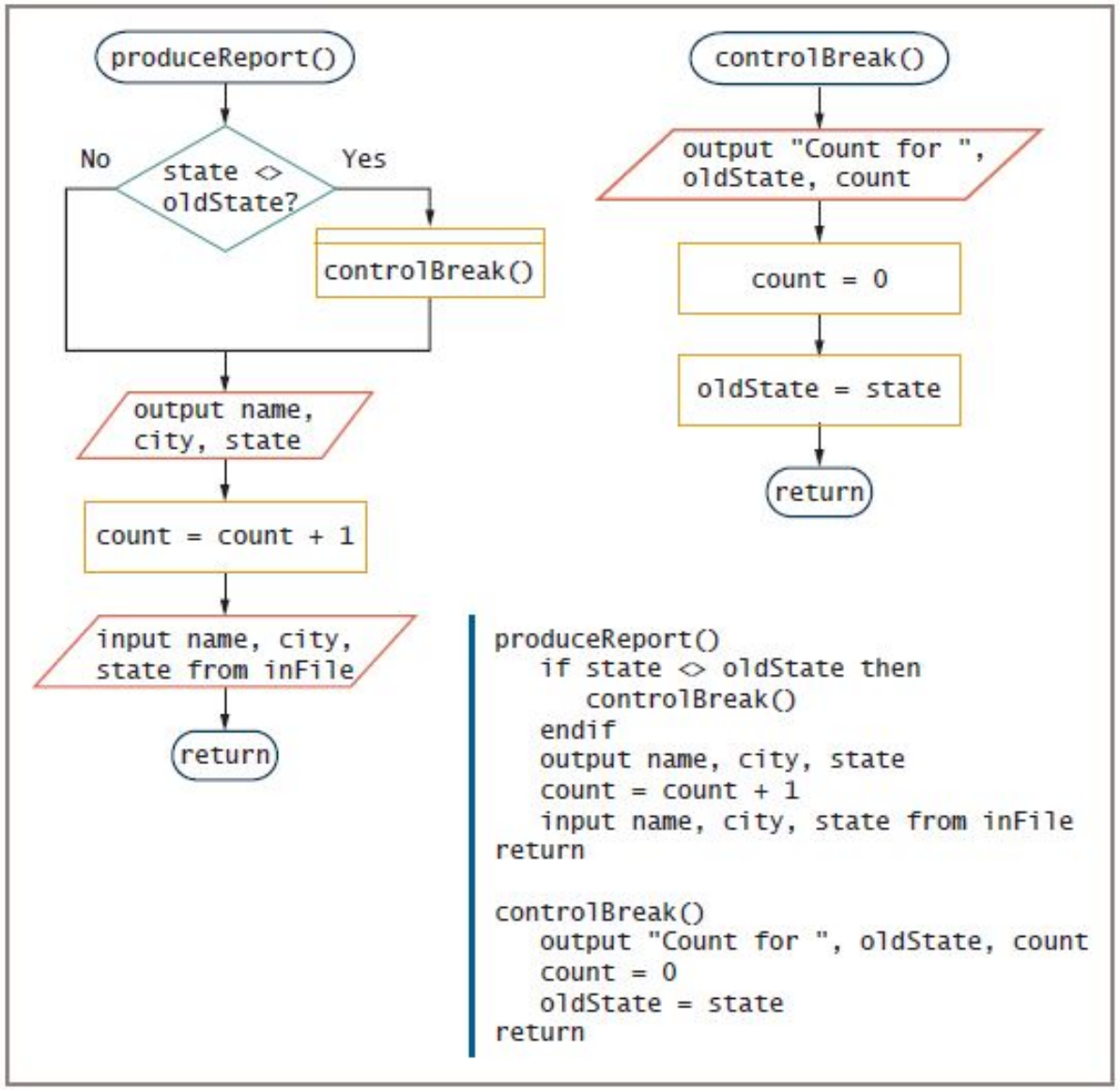


Figure 7-6 The `produceReport()` and `controlBreak()` modules for the program that produces clients by state

Understanding Sequential Files and Control Break Logic (continued)

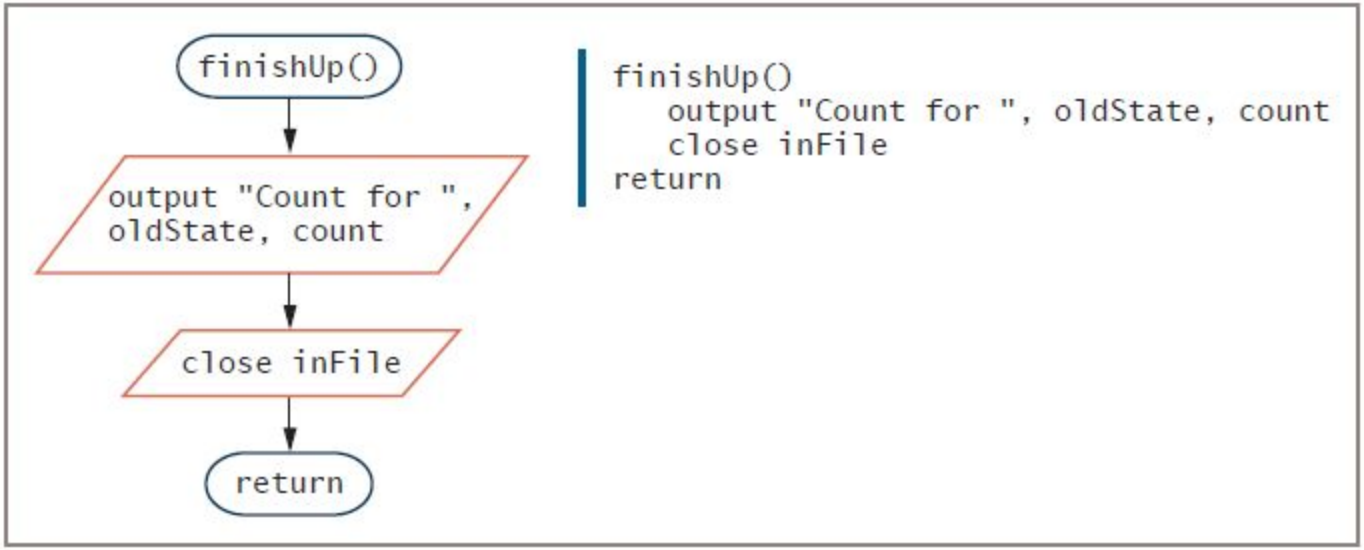


Figure 7-7 The `finishUp()` module for the program that produces clients by state report

Merging Sequential Files

- **Merging files**
  - Combining two or more files while maintaining the sequential order
- Examples
  - A file of current employees in ID number order, and a file of newly hired employees also in ID number order
  - A file of parts manufactured in the Northside factory in part-number order, and a file of parts manufactured in the Southside factory also in part-number order

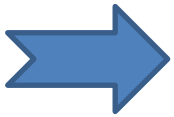
Merging Sequential Files (continued)

- Two conditions required for merging files
  - Each file has the same record layout
  - Sorted in the same order based on the same field
    - **Ascending order** (lowest to highest values)
    - **Descending order** (highest to lowest values)

Merging Sequential Files (continued)

East Coast File		West Coast File	
eastName	eastBalance	westName	westBalance
Able	100.00	Chen	200.00
Brown	50.00	Edgar	125.00
Dougherty	25.00	Fell	75.00
Hanson	300.00	Grand	100.00
Ingram	400.00		
Johnson	30.00		

Figure 7-8 Sample data contained in two customer files



mergedName	mergedBalance
Able	100.00
Brown	50.00
Chen	200.00
Dougherty	25.00
Edgar	125.00
Fell	75.00
Grand	100.00
Hanson	300.00
Ingram	400.00
Johnson	30.00

Figure 7-9 Merged customer file

Merging Sequential Files (continued)

- Mainline logic similar to other file-processing programs, except for handling two files
- With two input files, must determine when both files are at `eof`
  - Define a flag variable to indicate that both files have reached `eof`
  - Must define two input files
  - Read one record from each input file

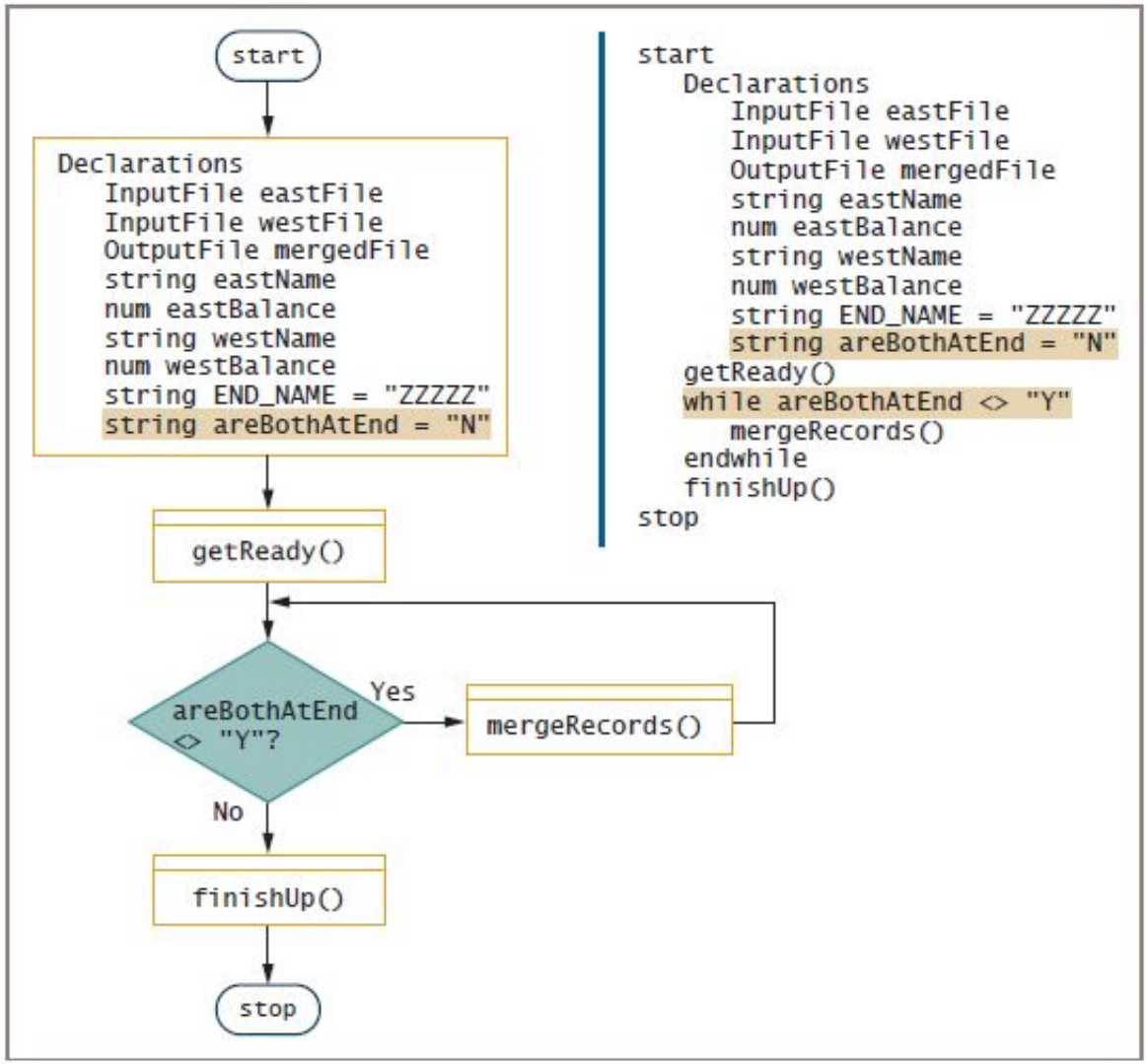


Figure 7-10 Mainline logic of a program that merges files

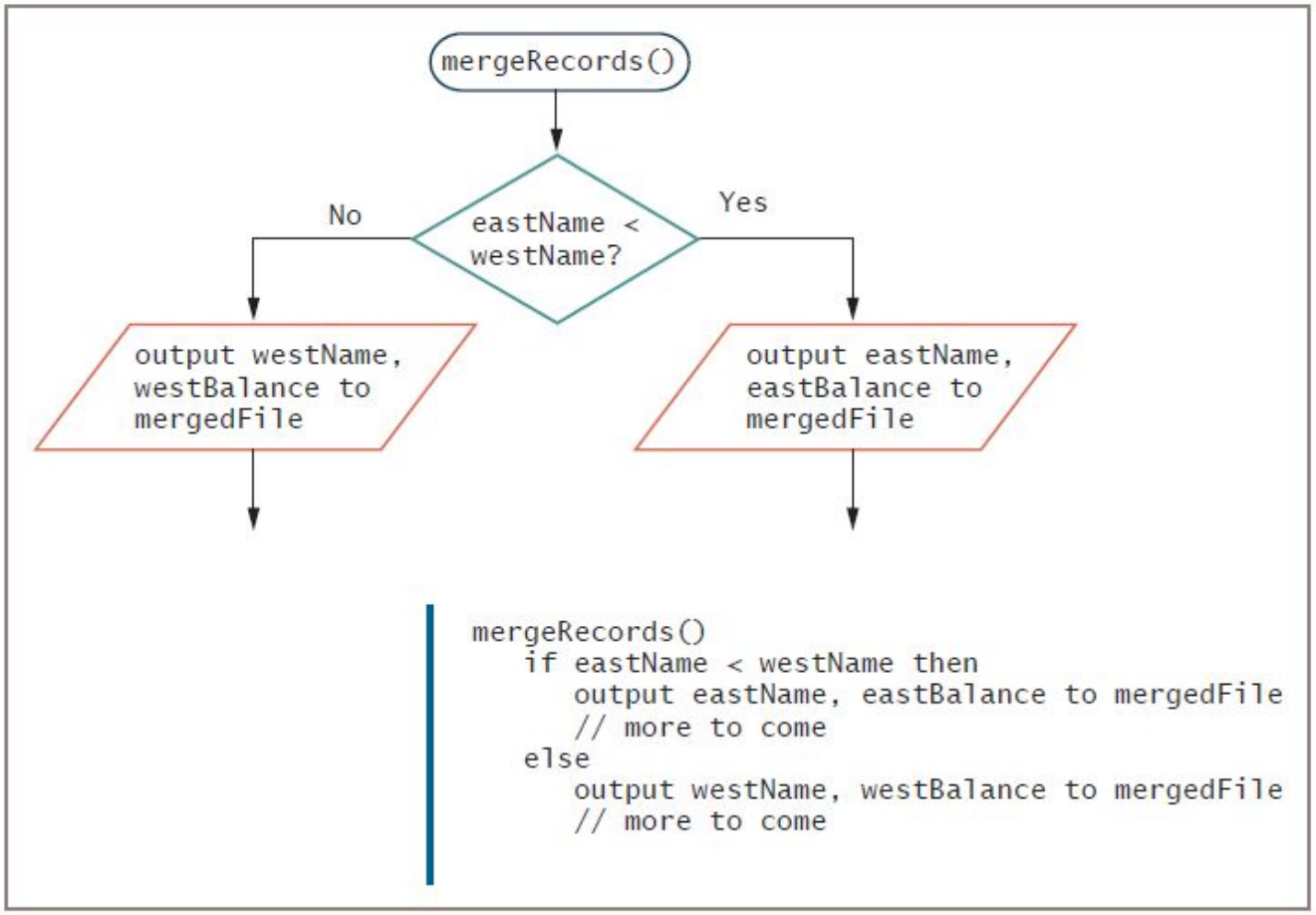


Figure 7-12 Start of merging process

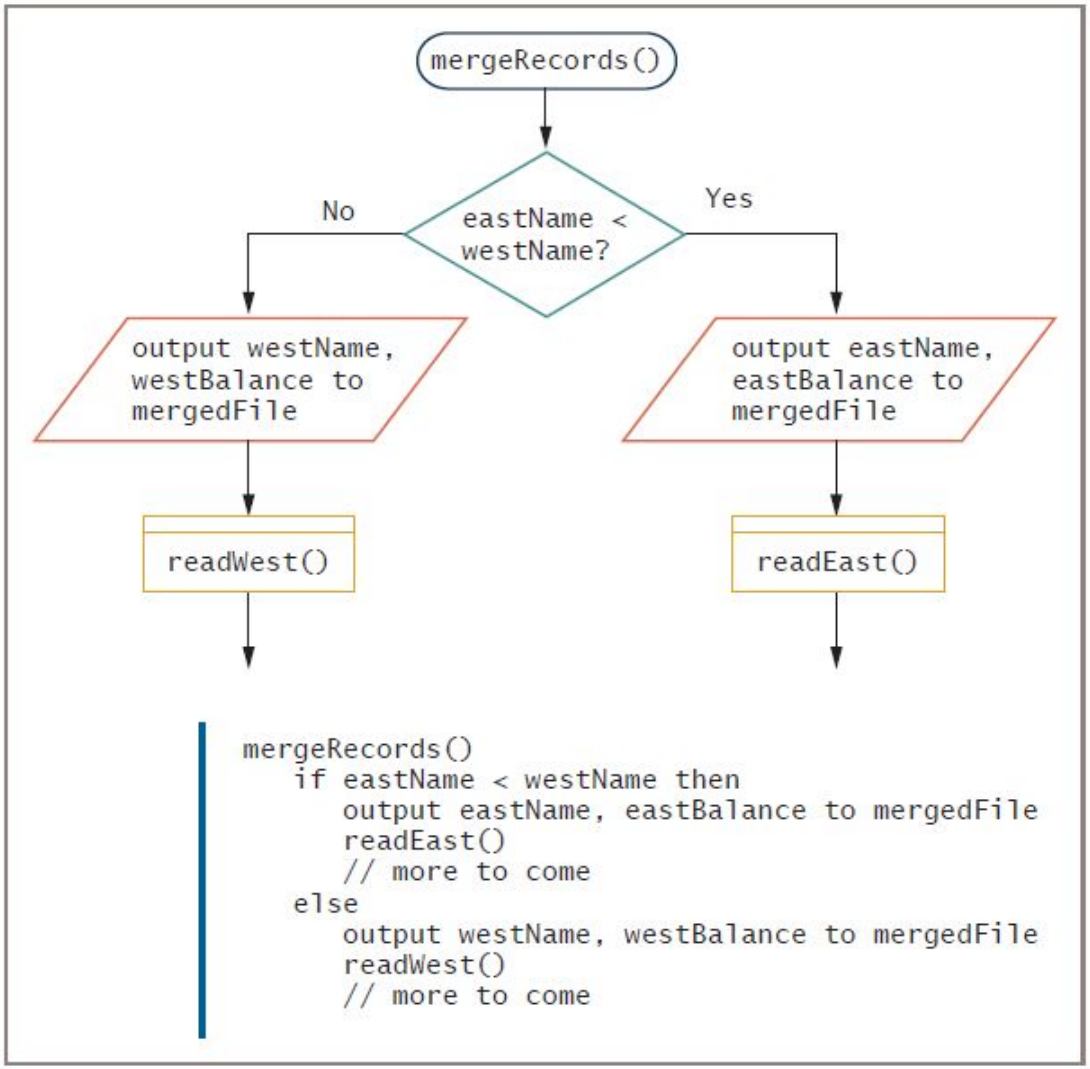


Figure 7-13 Continuation of merging process

Master and Transaction File Processing

- Some related files have a master-transaction relationship
- **Master file**
  - Holds relatively permanent data
- **Transaction file**
  - Contains temporary data to be used to update the master file
- **Update the master file**
  - Changes to values in its fields based on transactions

Master and Transaction File Processing (continued)

- Examples
  - A library maintains a master file of all patrons and a transaction file with information about each book or other items checked out
  - A college maintains a master file of all students and a transaction file for each course registration
  - A telephone company maintains a master file of every telephone line (number) and a transaction file with information about every call

Master and Transaction File Processing (continued)

- Updating approaches
  - Change information in master file
  - Copy master file and change new version
- Begin with both files sorted in the same order on the same field

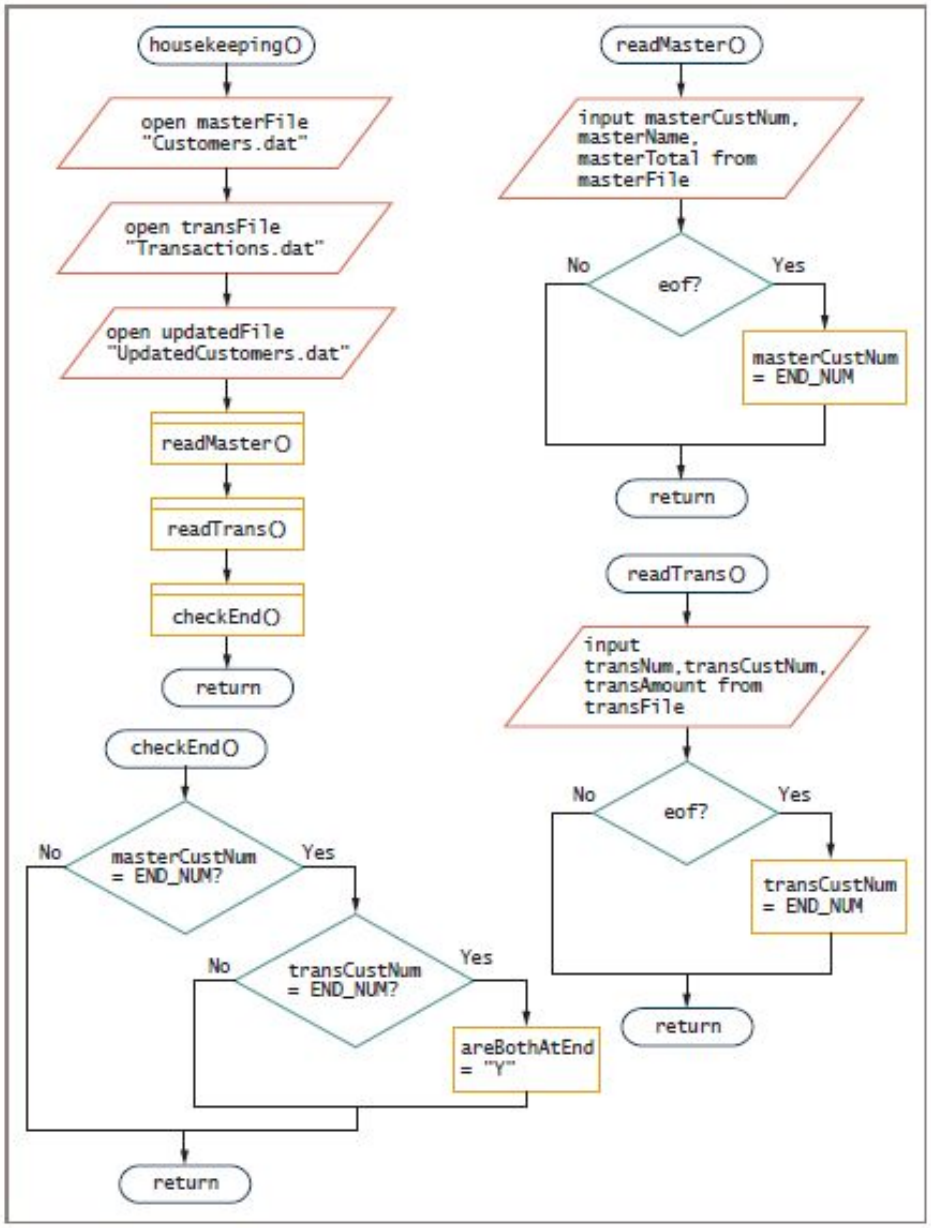


Figure 7-16 The `housekeeping()` module for the master-transaction program, and the modules it calls

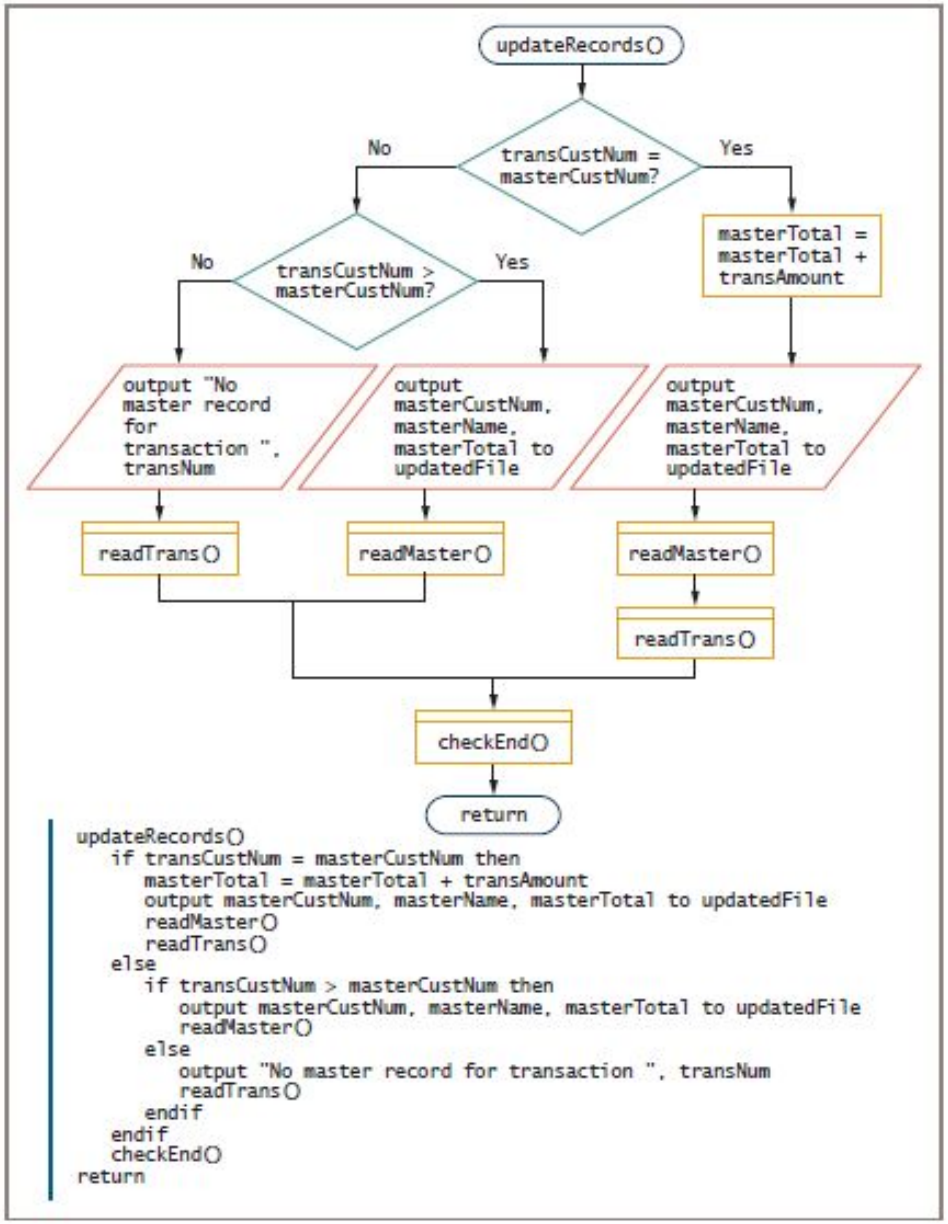


Figure 7-17 The `updateRecords()` module for the master-transaction program

Master and Transaction File Processing (continued)

Master File

masterCustNum	masterTotal
100	1000.00
102	50.00
103	500.00
105	75.00
106	5000.00
109	4000.00
110	500.00

Transaction File

transCustNum	transAmount
100	400.00
105	700.00
108	100.00
110	400.00

Figure 7-18 Sample data for the file-matching program

Random Access Files

- **Batch processing**
  - Involves performing the same tasks with many records, one after the other
  - Uses sequential files
- **Real-time** applications
  - Require that a record be accessed immediately while a client is waiting
- **Interactive program**
  - A program in which the user makes direct requests

Random Access Files (continued)

- **Random access files**
  - Records can be located in any order
  - **Instant access files**
    - Locating a particular record directly
  - Also known as **direct access files**

Summary

- Computer file
  - A collection of data stored on a nonvolatile device in a computer system
- Data items are stored in a hierarchy
- Using a data file
  - Declare, open, read, write, close
- Sequential file: records stored in some order
- Merging files combines two or more files
  - Maintains the same sequential order

Summary

- Master files
  - Hold permanent data
  - Updated by transaction files
- Real-time applications
  - Require random access files
  - Records stored in any order
  - Records accessed immediately