Learning Objectives

- Understand using Trailer record to control Repetitive structures
- Understand using End if Be (EOF) condition to control Repetitive structures
- Design programs using Trailer record logic and EOF condition.

Recall (Algorithm vocabulary)

<table>
<thead>
<tr>
<th>SYMBOLS</th>
<th>NAME</th>
<th>USE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>START</td>
<td>Terminal interrupt symbols</td>
</tr>
<tr>
<td></td>
<td>STOP</td>
<td>Terminal point (start, stop, or break)</td>
</tr>
<tr>
<td></td>
<td>Input/Output symbol</td>
<td>Reading data from an input medium or writing data to an output medium</td>
</tr>
<tr>
<td></td>
<td>Process symbol</td>
<td>Processing input data</td>
</tr>
</tbody>
</table>
Recall (Algorithm vocabulary)

<table>
<thead>
<tr>
<th>SYMBOLS</th>
<th>NAME</th>
<th>USE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flowline symbol</td>
<td>Sequence of operations and direction of data flow</td>
<td></td>
</tr>
<tr>
<td>Decision symbol</td>
<td>Decision-making operations</td>
<td></td>
</tr>
<tr>
<td>Predefined-process symbol</td>
<td>Operations specified elsewhere (not in the current algorithm)</td>
<td></td>
</tr>
</tbody>
</table>

Recall (Algorithm vocabulary)

<table>
<thead>
<tr>
<th>SYMBOLS</th>
<th>NAME</th>
<th>USE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector symbol</td>
<td>Exit to, or entry from, another part of the Flowchart</td>
<td></td>
</tr>
<tr>
<td>Preparation symbol</td>
<td>Control operations: set limit on loop-control variables, initialize accumulators, etc.</td>
<td></td>
</tr>
<tr>
<td>Annotation symbol</td>
<td>Addition explanation, comments.</td>
<td></td>
</tr>
</tbody>
</table>

Problem: Defective parts

A manufacturing company produces a large number of automotive parts each year in its two plants. Some of these plants are returned to the main sale Office because of defects in manufacture. An input record is prepared for each defective part, containing a code, part number, type of part, and date returned. The code is either 1 or 2, indicating whether plant 1 or plant 2 made the defective part.

<table>
<thead>
<tr>
<th>Plant Code</th>
<th>Part #</th>
<th>Part Type</th>
<th>Date Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>L45603</td>
<td>TAILLIGHT</td>
<td>04/16/02</td>
</tr>
<tr>
<td>1</td>
<td>W07722</td>
<td>WIPER</td>
<td>04/17/02</td>
</tr>
<tr>
<td>1</td>
<td>C19654</td>
<td>CLOCK</td>
<td>04/16/01</td>
</tr>
<tr>
<td>2</td>
<td>D33045</td>
<td>DOMELIGHT</td>
<td>04/20/01</td>
</tr>
</tbody>
</table>

Example of file containing input records:

A trailer record containing a plant code of 9 to signal the end of the input file
Problem: Defective parts

The company needs a read-and-print program to process the records and produce a listing of their contents. In addition, the number of defective parts manufactured by each plant is to be totaled for future reference.

The output of the program is to consist of a printed listing of the contents of the input records, followed by the total computed for each plant.

Solution: Using Trailer record
(Program Flowchart)

Solution: Using Trailer record
(Pseudocode)

Start
COUNT1 = 0
COUNT2 = 0
Read CODE, PART, TYPE, DATE
DO WHILE CODE ≠ 9
  IF CODE = 1 THEN
    COUNT1 = COUNT1 + 1
  ELSE
    COUNT2 = COUNT2 + 1
  ENDIF
Write CODE, PART, TYPE, DATE
Read CODE, PART, TYPE, DATE
ENDDO
Write COUNT1, COUNT2
Stop
Questions

- What if the Priming Read is omitted?
- What if the Loop Read is omitted?
- What if the DOWHILE statement (in the previous slide) is replaced by an IF statement?

Solution: Using Trailer record
(Program Flowchart)

Solution: Using Trailer record
(Pseudocode)

Start
Write header(s)
COUNT1 = 0
COUNT2 = 0
Read CODE, PART, TYPE, DATE
DOWHILE CODE ≠ 9
IF CODE = 1 THEN
COUNT1 = COUNT1 + 1
ELSE IF CODE = 2 THEN
COUNT2 = COUNT2 + 1
ELSE
Write 'Bad Input'
ENDIF
ENDIF
Write COUNT1, COUNT2
Stop
Identify main tasks performed by the program

Task 1: Write the headings
Task 2: Initialize counters
Task 3: Write detail records
Task 4: Write totals for each plant.

Note: Many ways for identifying main tasks

Structure Chart (or Hierarchy Chart)
- Shows relationships of all modules within a program

Overall Control Program Flowchart & Pseudocode
Solution: Using Trailer record & Modules

[Flowchart and Pseudocode]

Solution: Using Trailer record & Modules

[Flowchart and Pseudocode]

Solution: Using End-Of-File condition

- Not always necessary to physically place a trailer record
- Most programming languages support a built-in function for testing the End-of-File
Redo the Weekly Payroll problem done during Week 2 (see next 3 slides):
1) Using a Trailer record (NUM = 999)
2) Using an End-Of-File condition

Exercise : Weekly Payroll problem

Construct a program flowchart and corresponding pseudocode to solve the following problem: ABC company needs a weekly payroll report for its salespeople. Input to the program is a salesperson's name, number, and weekly sales. Output is the salesperson’s name, number, and pay. Each salesperson receives a base pay of $300 as well as a 10% commission on his or her total sales up to and including $500. Any sales over $500 merit a 15% commission for the employee. (For example, if sales = $600, then pay = $300 + $50 (or .10 * 500) + $15 (.15 * 100) = $350). Use a DOWHILE loop and a counter to compute the weekly payroll for exactly 20 employees.

Initial Solution: (Program Flowchart)
Initial Solution: (Pseudocode)

Start
COUNT = 0
DOWHILE COUNT < 20
  Read NAME, NUM, SALES
  IF SALES > 500 THEN
    PAY = 300 + (500 * .10) + ((SALES - 500) * .15)
  ELSE
    PAY = 300 + (SALES * .10)
  ENDIF
  Write NAME, NUM, PAY
  COUNT = COUNT + 1
ENDDO
Stop

Solution: Using Trailer record
(To be done in class)

Solution: Using Trailer record
(Pseudocode)
(To be done in class)
Solution: Using EOF (Program Flowchart) (To be done in class)

Solution: Using Trailer record (Pseudocode) (To be done in class)