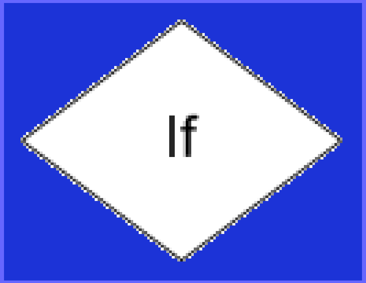
Selection Control Structure

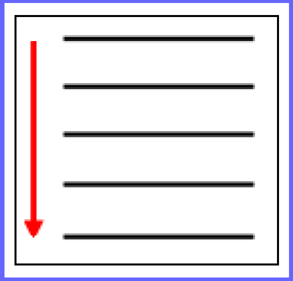


Topics

- Review sequence control structure
- Structure theorem
- Selection control structure
- If statement
- Relational operators
- Types of selection
- Truth tables

Sequence

- Recall that the default control structure is sequence
- Our examples up to this point have all been of this type
- That is, the flow of control through the statements in our algorithms and programs is from top to bottom, sequentially



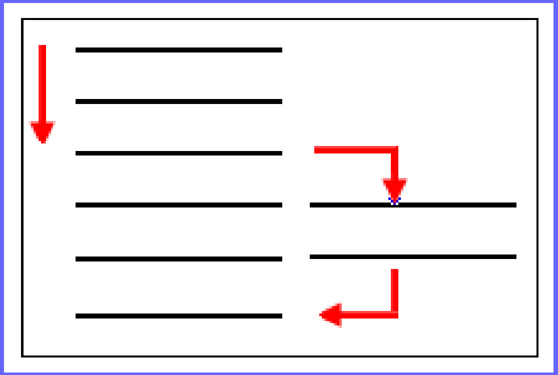
Structure Theorem

- There are two other control structures used in algorithms: Selection and repetition
- It has been shown that these three (sequence, selection, and repetition) are enough control structures to write any computer program—The Structure Theorem

http://en.wikipedia.org/wiki/Structured_program_theorem

Selection

- We will deal with repetition later
- Selection involves making a decision to execute several statements or not
- Computers have the capability of making a decision by comparing one value with another



Selection

- To introduce selection control, we include a condition statement that compares two values such that the result is true or false
- That is, there will be a choice between two cases
  - If the condition is true one choice is selected
  - If it is false the other choice is selected
- The condition statement is written as an If statement and it allows the algorithm and program to make a decision

If Statement

```
If condition is true Then
    Perform these statements
Else
    Perform these statements
EndIf
```

Example:

```
If book is hardback Then
    price = 65.00
Else
    price = 35.00
EndIf
```

condition = “book is hardback” and this will be true or false

Relational Operators

- Also called <u>comparison operators</u>
- We use these operators in our conditions to compare two values:

<	less than	<=	less than or equal
>	greater than	>=	greater than or equal
=	equal	< >	not equal

Examples

Condition	Result	Condition	Result
10 < 5	false	110 <= 100	false
5 < 10	true	12.5 <= 13.0	true
1 > 3	false	5 = 5	true
3 > 1	true	10.5 = 5.75	false
15 >= 10	true	90 < > 80	true
15 >= 15	true	90 < > 90	false

Different Operators

- Note that we are using the same character, the equal sign =, as the <u>assignment operator</u> and the <u>equal comparison operator</u>
- They are different operators
- <u>Examples</u> (these are different operations)

Assignment: tot = tot + 1

Comparison: If tot = 100 Then

Types of Selection

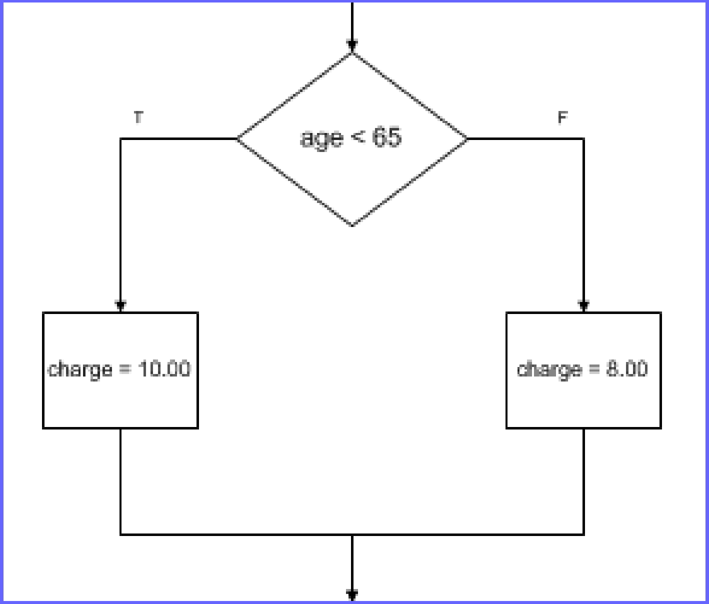
- There are three main varieties of the selection structure:
  - Simple selection – choice between two alternatives
  - Null else selection – perform statements only when condition is true
  - Combined selection – multiple conditions using And or Or
- The following slides illustrate these showing the flowchart, pseudocode, and Small Basic code

Simple Selection

Pseudocode

```
If age < 65 Then
    charge = 10.00
Else
    charge = 8.00
EndIf
```

Flowchart



Small Basic code

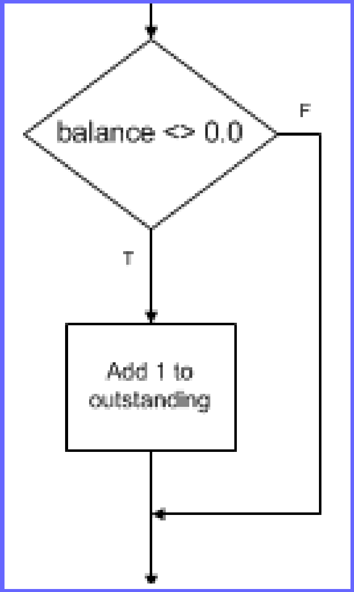
```
If age < 65 Then
  charge = 10.00
Else
  charge = 8.00
EndIf
```

Null Else Selection

Pseudocode

```
If balance <> 0.0 Then
    Add 1 to outstanding
EndIf
```

Flowchart



Small Basic code

```
If balance <> 0.0 Then
  outstanding = outstanding + 1
EndIf
```

Null Else Selection

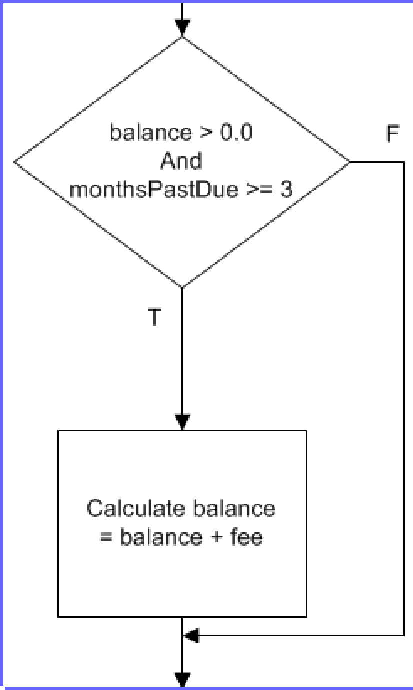
- As shown in the last example, when the Else clause is not needed, one simply doesn’t use the Else keyword
- This situation arises often

Combined Selection (And)

Pseudocode

```
If balance > 0.0 And monthsPastDue >= 3 Then
    Calculate balance = balance + fee
EndIf
```

Flowchart



Small Basic code

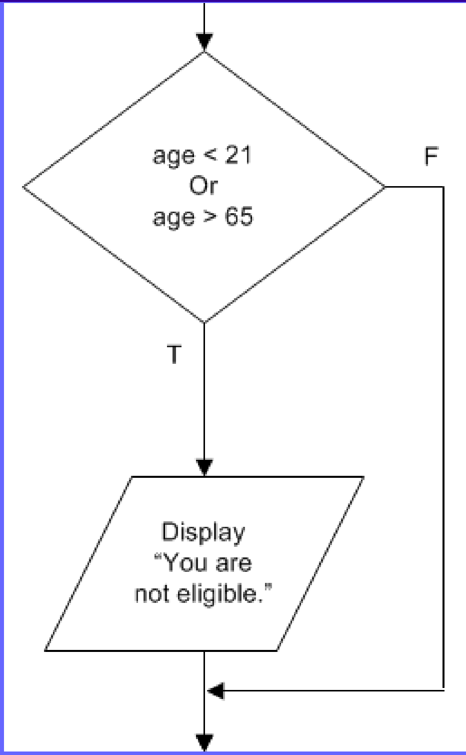
```
If balance > 0.0 And monthsPastDue >= 3 Then
        balance = balance + fee
EndIf
```

Combined Selection (Or)

Pseudocode

```
If age < 21 Or age > 65 Then
    Display "You are not eligible."
EndIf
```

Flowchart



Small Basic code

```
If age < 21 Or age > 65 Then
        TextWindow.WriteLine("You are not eligible.")
EndIf
```

Combined Selection

- The examples above for combined selection don’t have an Else clause
- But if the situation warrants it the Else clause can be used with combined selection, for example:

```
If age < 21 Or age > 65 Then
   Display “You are not eligible.”
Else
   Display “You are eligible.”
EndIf
```

Multiple Statements

- The above examples show just one statement in the Then and Else clauses of the If statement
- But if needed there may be multiple statements in those clauses, for example:

```
If age < 21 Or age > 65 Then
   totNotEligible = totNotEligible + 1
   Display “You are not eligible.”
Else
   totEligible = totEligible + 1
   Display “You are eligible.”
EndIf
```

Structured Programming

- Notice how the indentation in the above examples helps us to see which statements are dependent on others
- This is one feature of what is called “structured programming,” that is, using good programming style
- There are multiple ways to write an algorithm and program, but some ways are better than others

Structured Programming

- One feature of good programming style, is to make our algorithms and programs typographically readable by indenting statements properly
- Look at the following two examples
- Which do you think uses good programming style and is more readable?

Structured Programming

<u>Example 1</u>

```
If age < 21 Or age > 65 Then
    totNotEligible = totNotEligible + 1
    Display “You are not eligible.”
Else
    totEligible = totEligible + 1
    Display “You are eligible.”
EndIf
```

<u>Example 2</u>

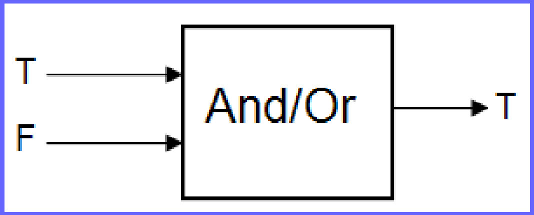
```
If age < 21 Or age > 65 Then
totNotEligible = totNotEligible + 1
Display “You are not eligible.”
Else
totEligible = totEligible + 1
Display “You are eligible.”
EndIf
```

Structured Programming

- It is not enough to write algorithms and programs that work, we need to follow good programming style so that they are easier to read and maintain
- That is, well written algorithms and programs allow the person who writes the statements to make fewer errors, and allows the person maintaining the code to do that more easily

Truth Tables for And & Or

- When combining conditions with And or Or, we are effectively combining true’s and false’s to arrive at a new true or false
- That is, And and Or are binary operations that take two true/false values and produce one true/false value:



- The way these operations do this is shown in the following truth tables

Truth Tables for And & Or

And	T	F
T	T	F
F	F	F

Or	T	F
T	T	T
F	T	F

- For example, (1 = 2 And 3 = 3) is false because 1 = 2 is false and 3 = 3 is true, but the And truth table indicates that false And true is false
- Also, (4 < 5 Or 6 > 7) is true because 4 < 5 is true and 6 > 7 is false, but the Or truth table indicates that true Or false is true

Summary

- Control structures refer to the order in which statements are executed in algorithms and programs
- There are three main control structures: Sequence, selection, and repetition
- In selection, there is a comparison of values that determines which statement(s) are executed next

Summary

- We use the If statement in pseudocode and Small Basic to implement the selection control structure
- The relational operators are used to make the comparison
- Types of selection include:
  - Simple selection
  - Null else selection
  - Combined selection
- Truth tables for And and Or help us to understand how the combined selection structure works

Terminology

- Selection control structure
- Structure theorem
- Condition
- If statement
- Relational operators
- Simple selection
- Null else selection
- Combined selection
- Structured programming
- Programming style
- Truth tables

End