Structured Programming

- Only three forms of controls are needed
  - sequence
  - selection
  - repetition
- Rules for forming properly structured programs

Sequence

![Sequence Diagram]

- Only three forms of controls are needed
  - sequence
  - selection
  - repetition
- Rules for forming properly structured programs
Selection

If-Then Structure (Single Selection)

- Condition True → Action
- Condition False

If-Then-Else Structure (Double Selection)

- Action True
- Condition False

Select Case Structure (Multiple Selection)

- Case 1 True → Case 1 action
- Case 1 False
- Case 2 True → Case 2 action
- Case 2 False
- ... Case n
- Case n True → Case n action
- Case n False

Repetition

While/Wend Structure

- Action
- Condition True
- Condition False

Do While/Loop Structure

- Action
- Condition True
- Condition False

Do/Loop Until Structure

- Action
- Condition False
- Condition True
Repetition

Do Until/Loop Structure

Do/Loop While Structure

Repetition

For/Next Structure
Rules for Forming Properly Structured Programs

- Stepwise refinement
  1. replace one statement with multiple statements in a sequence
  2. replace one statement with a control structure
  3. Rules 1 & 2 can be applied in any order

Rule 2 for Forming Properly Structured Programs
Rule 2 for Forming Properly Structured Programs

Proper and Illegal Structures

Stacked building blocks

Nested building blocks

Overlapping building blocks
Illegal Structures

How to Determine the “Structuredness” of a Program

• Apply the three rules in reverse to reduce the flow chart to the simplest flow chart
Private Sub calculate()
    Select Case strOperator
        Case "+"
            txtInput.Text = dblOperand1 + txtInput.Text
        Case "-
            txtInput.Text = dblOperand1 - txtInput.Text
        Case "/"
            If txtInput.Text > -0.00000001 And
                txtInput.Text < 0.0000000001 Then
                txtInput.Text = "Divide by zero"
                blnDivideByZero = True
            Else
                txtInput.Text = dblOperand1 / txtInput.Text
            End If
        End Select
    End Sub

Constructing Flow Charts

• one flow chart for each subroutine or function
• if subroutines are the same except certain parameters, generalize and use one flow chart
Example - Problem 5.21

- Write a program that prints the following diamond shape (the maximum *’s in a row is given by the user).

```
*  
**  
***  
****  
*****

N = 5

******
******
****
***
*  

N = 6

********
******
***
*  
```

```
Start
Get n
Print top half
Print bottom half
Stop

Start
Get n
Print row i
Print bottom half
i = 1
i > n
i = i + 1
Start
Get n
Print column j
Print bottom half
i = 1
i > n
i = i + 1
Stop
```