

Laboratório de Programação I

Estruturas de Controle: Parte I

Fabricio Breve

Objetivos

- Entender as técnicas básicas de solução de problemas
- Desenvolver algoritmos por meio do processo de refinamento *top-down* gradual
- Usar as estruturas de seleção If/Then e If/Then/Else para escolher entre as ações alternativas
- Usar as estruturas de repetição While. Do While/Loop e Do
- Entender a repetição controlada por controlador e a repetição controlada por sentinela
- Usar os operadores de atribuição

Algoritmos

- Procedimentos para solucionar problemas em termos de:
 - Ações a serem executadas
 - Ordem na qual elas devem ser executadas

Exemplo de Algoritmo

- Levantar-e-preparar-para-sair:
 1. Levante-se da cama
 2. Tire o pijama
 3. Tome um banho
 4. Vista-se
 5. Tome o café da manhã
 6. Vá ao trabalho
- O que aconteceria se essa ordem não fosse respeitada?

Pseudocódigo

- Linguagem informal que ajuda o programador a descrever algoritmos
- Não é executado em computadores
 - Semelhante ao português cotidiano
- Ajudam o programador a “pensar” no programa antes de escrevê-lo em uma linguagem de programação

Estruturas de Controle

- Normalmente os comandos de um programa são executados de forma seqüencial
- O VB inclui vários comandos que permitem mudar essa ordem
 - Transferência de controle
 - Programação Estruturada

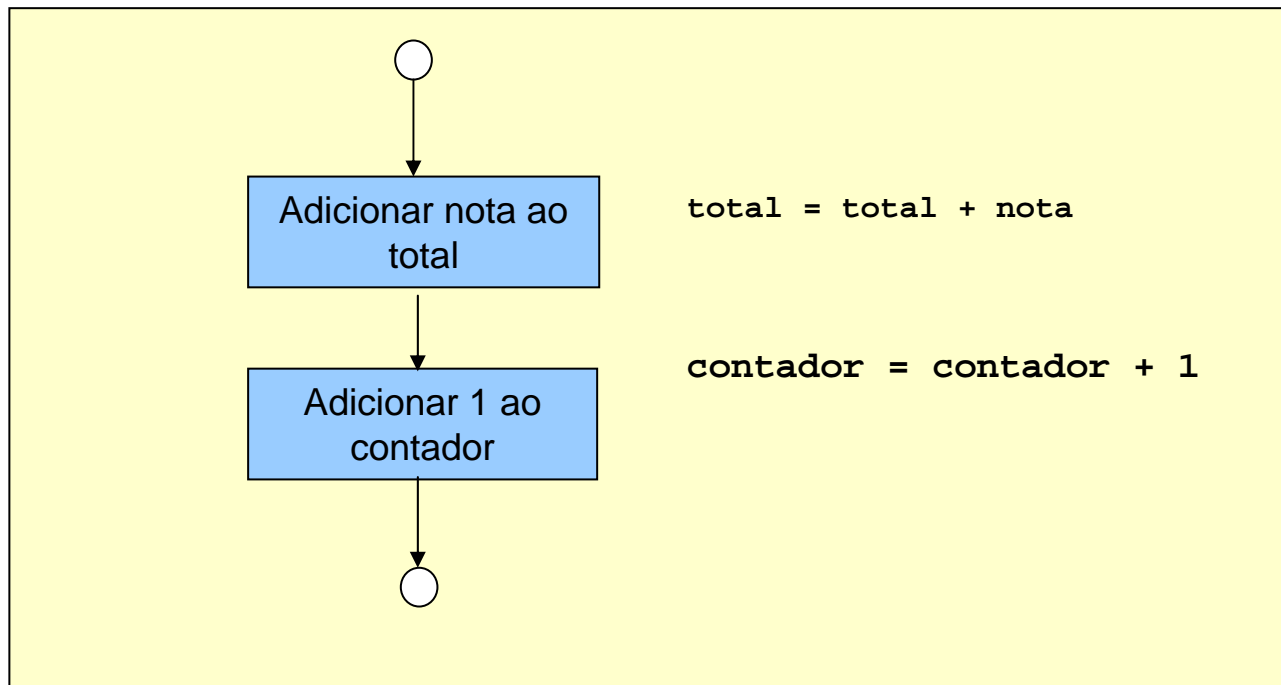
Estruturas de Controle

- Todos os programas podem ser escritos utilizando apenas três estruturas de controle
 - Estrutura de Seqüência
 - Estrutura de Seleção
 - Estrutura de Repetição

Fluxogramas

- Representação gráfica de um algoritmo
- Desenhado usando símbolos especiais:
 - Retângulos
 - Losangos
 - Ovais
 - Círculos
- Conectado por setas chamadas linhas de fluxo

Fluxograma



Estruturas de Seleção

- If/Then
 - Estrutura de seleção simples
- If/Then/Else
 - Estrutura de seleção dupla
- Select Case
 - Estrutura de seleção múltipla

Estrutura de Seleção If/Then

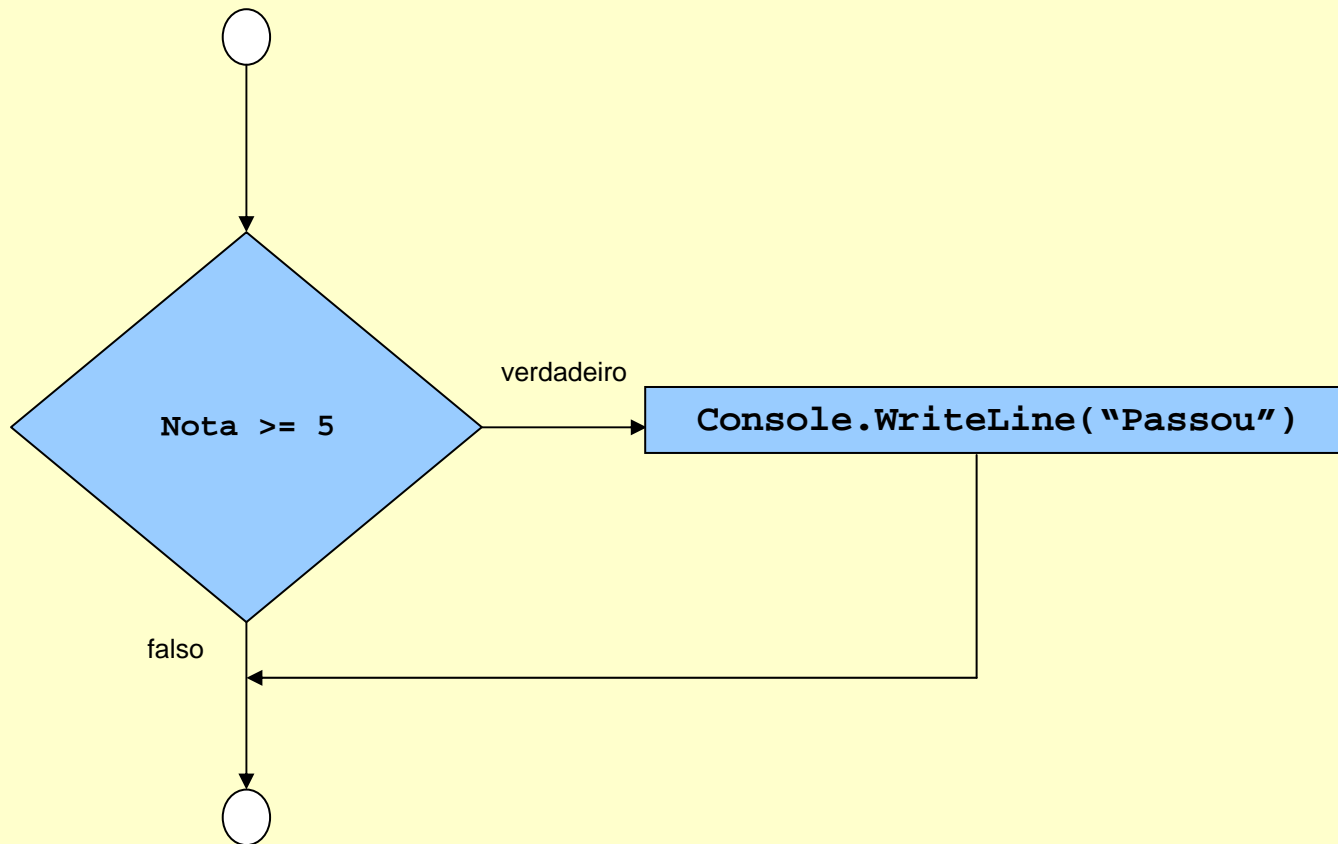
- Pseudocódigo:

Se a nota do estudante é maior ou igual a 5
Imprima "Passou"

- Visual Basic:

```
If Nota >= 5 Then  
    Console.WriteLine ("Passou")  
End If
```

Fluxograma de If/Then



Estrutura de Seleção If/Then/Else

- **Pseudocódigo:**

Se a nota do estudante é maior ou igual a 5

 Imprima "Passou"

Senão

 Imprima "Reprovou"

- **Visual Basic:**

If Nota >= 5 Then

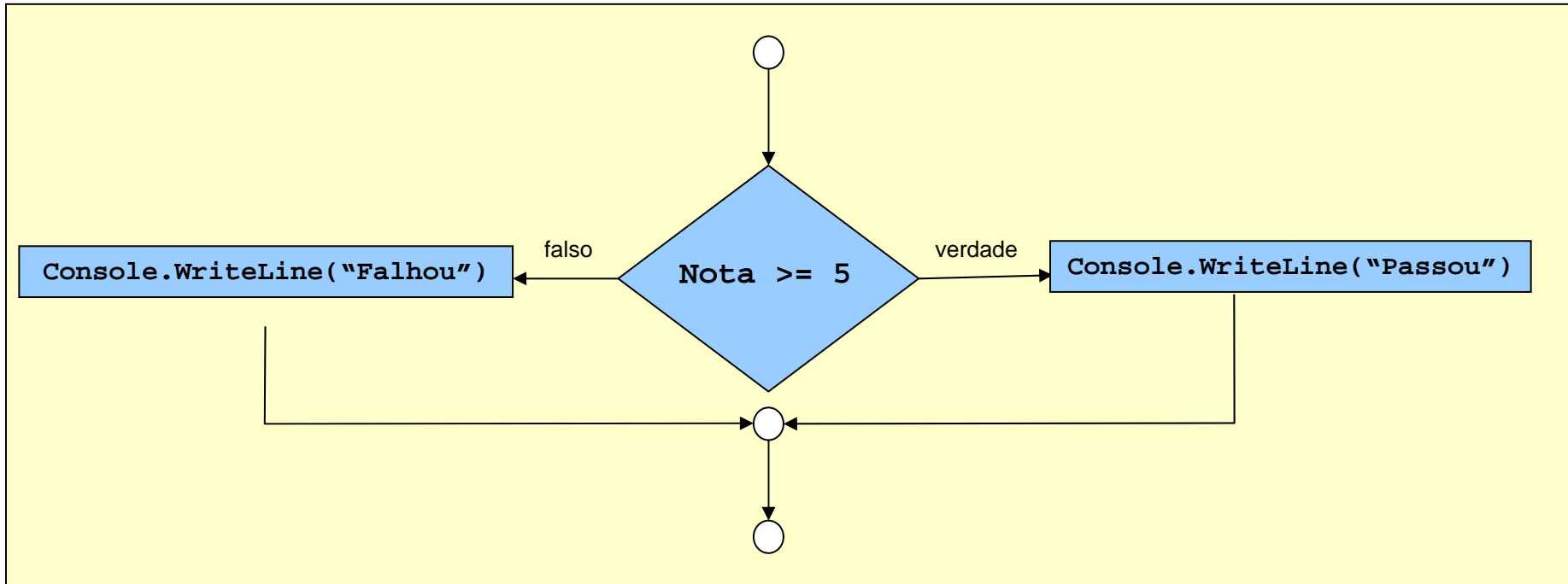
 Console.WriteLine ("Passou")

Else

 Console.WriteLine ("Reprovou")

End If

Fluxograma If/Then/Else



Estruturas de Repetição

- While
- Do While/Loop
- Do/Loop While
- Do Until/Loop
- Do/Loop Until
- For/Next
- For Each/Next

Estrutura de Repetição While

- Permite ao programador especificar uma ação que deve ser repetida, dependendo do valor de uma condição.
- Exemplo (pseudocódigo):

Enquanto houver mais itens em minha lista de compras

 Compre o próximo item

 Risque esse item de minha lista

While.vb

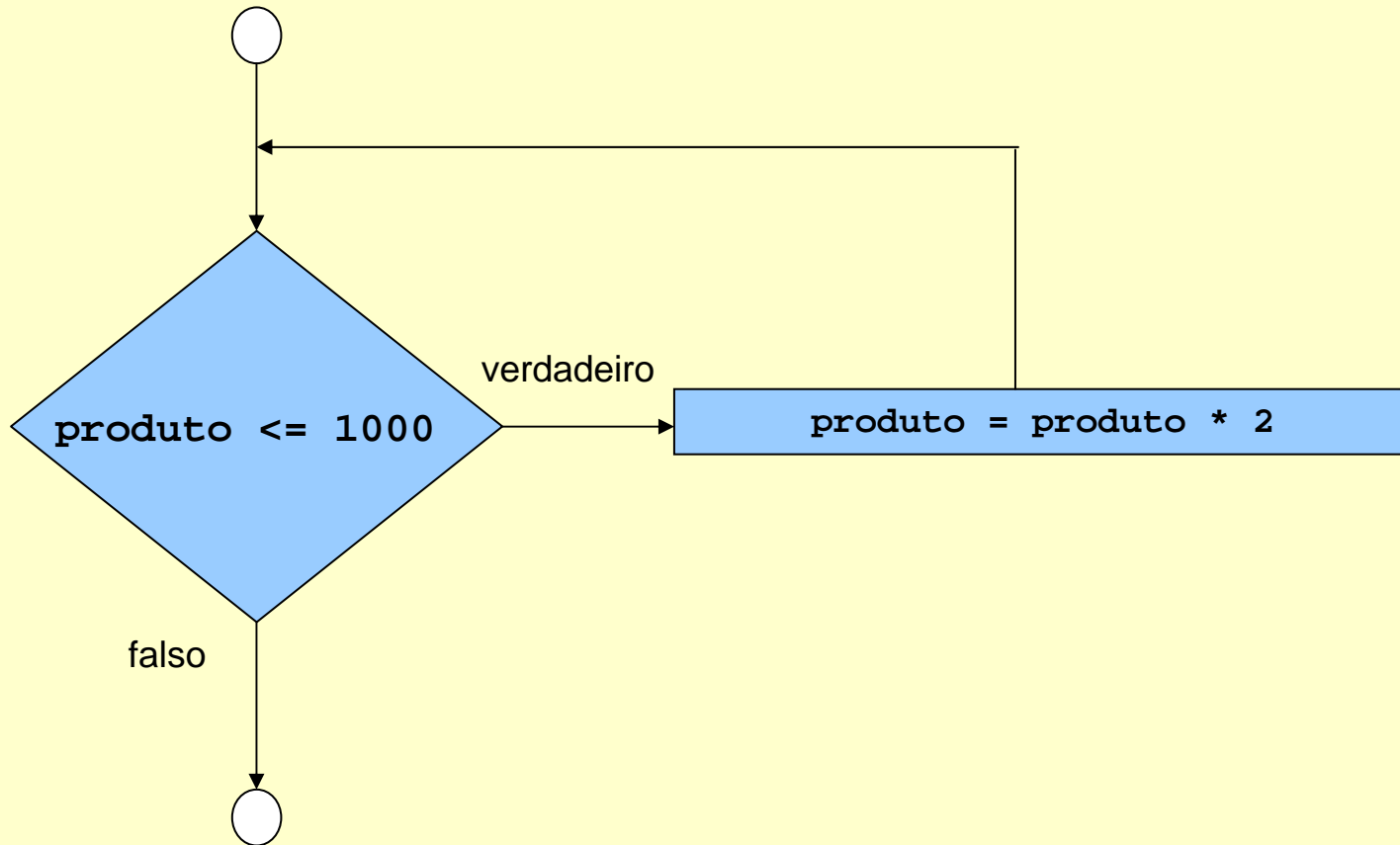
```
1 ' While.vb
2 ' Demonstração da Estrutura While
3
4 Module modWhile
5
6     Sub Main()
7         Dim produto As Integer = 2
8
9         ' estrutura multiplica e exibe o produto
10        ' enquanto o produto for menor ou igual a 1000
11        While produto <= 1000
12            Console.WriteLine("{0} ", produto)
13            produto = produto * 2
14        End While
15
16        Console.WriteLine() ' escreve uma linha em branco
17
18        ' imprime o resultado
19        Console.WriteLine("A menor potência de 2 " & _
20            "maior que 1000 é {0}", produto)
21        Console.ReadLine() ' impede que a janela feche
22    End Sub ' Main
23
24 End Module ' modWhile
```

O teste é feito cada vez que a estrutura é executada

```
2  4  8  16  32  64  128  256  512
A menor potência de 2 maior que 1000 é 1024
```

Saída do programa

Fluxograma de While



Estrutura de Repetição Do While/Loop

- Funciona da mesma forma que While

DoWhile.vb

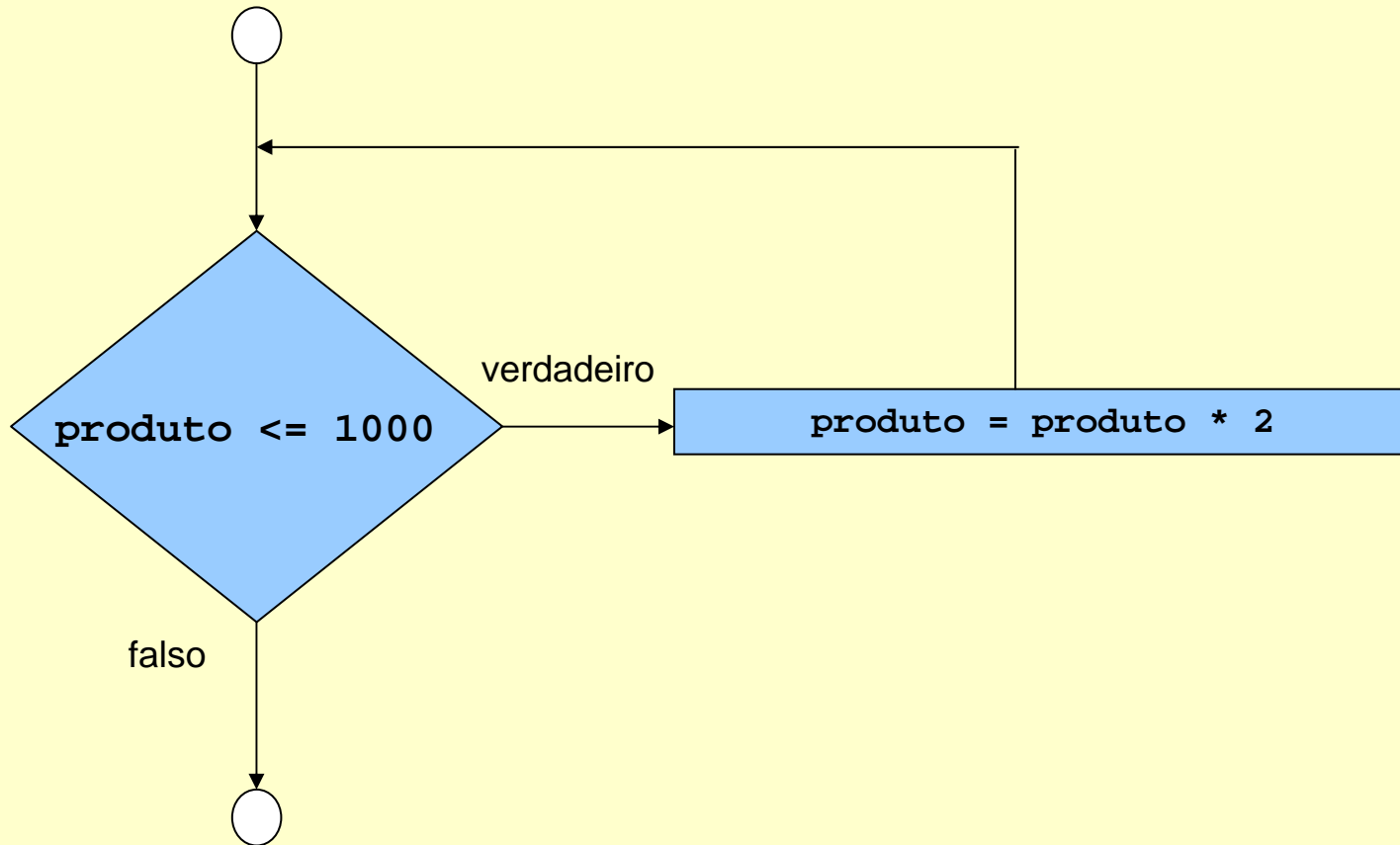
```
1 ' DoWhile.vb
2 ' Demonstração da Estrutura While/Loop
3
4 Module modDoWhile
5
6     Sub Main()
7         Dim produto As Integer = 2
8
9         ' estrutura multiplica e exibe o produto
10        ' enquanto o produto for menor ou igual a 1000
11        Do While produto <= 1000
12            Console.WriteLine("{0} ", produto)
13            produto = produto * 2
14        Loop
15
16        Console.WriteLine() ' escreve uma linha em branco
17
18        ' imprime o resultado
19        Console.WriteLine("A menor potência de 2 " & _
20            "maior que 1000 é {0}", produto)
21        Console.ReadLine() ' impede que a janela feche
22    End Sub ' Main
23
24 End Module ' modDoWhile
```

Esquecer de colocar uma ação que eventualmente torne a condição falsa é um erro lógico e causa um loop infinito

```
2  4  8  16  32  64  128  256  512
A menor potência de 2 maior que 1000 é 1024
```

Saída do programa

Fluxograma de Do While/Loop



Estrutura Do Until/Loop

- Diferentemente de While e Do While/Loop, a estrutura Do Until/Loop executa o loop enquanto o comando avaliado for falso

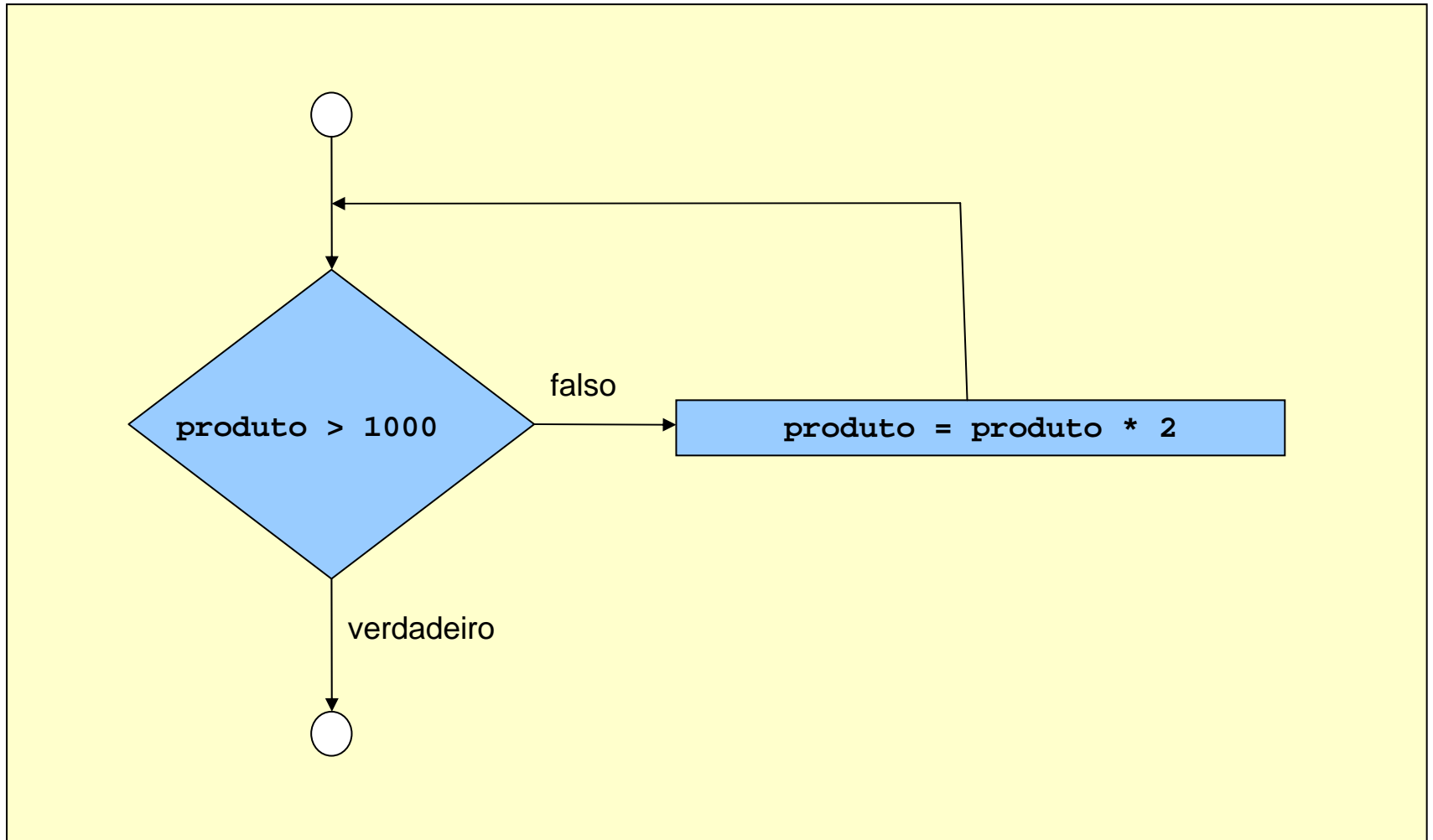
DoUntil.vb

```
1 ' DoUntil.vb
2 ' Demonstração da estrutura Do Until/Loop.
3
4 Module modDoUntil
5
6     Sub Main()
7         Dim produto As Integer = 2
8
9         ' procure primeira potência de 2 maior que 1000
10        Do Until produto > 1000
11            Console.WriteLine("{0} ", produto)
12            produto = produto * 2
13        Loop
14
15        Console.WriteLine() ' escrever uma linha em branco
16
17        ' imprimir o resultado
18        Console.WriteLine("A menor potência de 2 " & _
19            "maior que 1000 é {0}", produto)
20    End Sub ' Main
21
22 End Module ' modDoUntil
```

```
2  4  8  16  32  64  128  256  512
A menor potência de 2 maior que 1000 é 1024
```

Saída do programa

Fluxograma de Do Until / Loop



Operadores de Atribuição

- Operadores de atribuição podem ser abreviados
- Exemplo:
 - `valor = valor + 3`
 - `valor += 3`
- Isso vale para os operadores binários:
 - `+, -, *, ^, &, / or \`
 - Sintaxe:
 - `variável = variável operador expressão`
 - `variável operador= expressão`

Operadores de Atribuição

Operador de atribuição	Expressão	Explicação	Atribuição
<i>Assuma:</i> $c = 4$, $d =$ "O"			
$+=$	$c += 7$	$c = c + 7$	11 a c
$-=$	$c -= 3$	$c = c - 3$	1 a c
$*=$	$c *= 4$	$c = c * 4$	16 a c
$/=$	$c /= 2$	$c = c / 2$	2 a c
$\backslash=$	$c \backslash= 3$	$c = c \backslash 3$	1 a c
$\wedge=$	$c \wedge= 2$	$c = c \wedge 2$	16 a c
$\&=$	$d \&= \text{"lá"}$	$d = d \& \text{"lá"}$	"Olá" a d

```
1 ' Atribuicao.vb
2 ' Usando um operador de atribuição para calcular a potência de 2.
3
4 Module modAtribuicao
5
6     Sub Main()
7         Dim expoente As Integer ' potência a ser digitada pelo usuário
8         Dim resultado As Integer = 2 ' número a ser elevado a potência
9
10        ' pedir para o usuário digitar a potência
11        Console.WriteLine("Digite um expoente inteiro: ")
12        expoente = Console.ReadLine()
13
14        resultado ^= expoente ' equivale a resultado = resultado ^ expoente
15        Console.WriteLine("Resultado ^= Expoente: {0}", resultado)
16
17        resultado = 2 ' reiniciar o valor da base
18        resultado = resultado ^ expoente
19        Console.WriteLine("Resultado = resultado ^ expoente: {0}", resultado)
20
21    End Sub ' Main
22
23 End Module ' modAtribuicao
```

Mesmo efeito na
variável **resultado**

```
Digite um expoente inteiro: 8
resultado ^= expoente: 256
resultado = resultado ^ expoente: 256
```

Saída do programa

Formulando Algoritmos: Estudo de Caso 1 (Repetição Controlada por Contador)

- Contador: especifica o número de vezes que um conjunto de instruções será executado
- Considere o seguinte problema:
 - Uma classe de dez estudantes fizeram uma prova, as notas (inteiros que variam de 0 a 10) estão disponíveis para você. Determine a média da classe nessa prova.

Pseudocódigo do problema

Atribua zero ao total

Atribua 1 ao contador de nota

Enquanto o contador de nota for menor ou igual a 10

 Obtenha a entrada da próxima nota

 Adiciona a nota ao total

 Adicione 1 ao contador da nota

Atribua o total dividido por 10 à média de nota

Imprima a média da classe

```
'modMedia.vb
'Usando a repetição controlada por contador
```

modMedia.vb

```
Module modMedia
```

```
Sub Main()
```

```
'declarando as variáveis
```

```
Dim total As Integer
```

```
Dim contador As Integer
```

```
Dim nota As Integer
```

```
Dim media As Double
```

```
'soma das notas
```

```
'contador de notas
```

```
'nota digitada pelo usuário
```

```
'média da classe
```

total acumula a soma das notas digitadas

contador conta o número de notas digitadas

```
'fase de inicialização
```

```
total = 0
```

```
contador = 1
```

```
'atribui 0 ao total
```

```
'prepara o contador para o loop
```

```
'fase de processamento
```

```
While contador <= 10
```

```
'solicita ao usuário que digite uma nota e lê a nota
```

```
Console.WriteLine("Digite uma nota inteira: ")
```

```
nota = Console.ReadLine()
```

```
total += nota
```

```
contador += 1
```

```
'adiciona a nota ao total
```

```
'adiciona 1 ao contador
```

estrutura while se repete enquanto contador for menor ou igual a 10

```
End While
```

contador é incrementado em um para cada nota lida, e eventualmente a condição do while se torna falsa

```
'fase de finalização
```

```
media = total / 10
```

```
'imprime uma linha em branco e exibe a média da classe
```

```
Console.WriteLine()
```

```
Console.WriteLine("Média da Classe é {0}", media)
```

```
Console.ReadKey()
```

```
End Sub
```

```
End Module
```

```
Digite uma nota inteira: 6
Digite uma nota inteira: 3
Digite uma nota inteira: 2
Digite uma nota inteira: 4
Digite uma nota inteira: 7
Digite uma nota inteira: 8
Digite uma nota inteira: 10
Digite uma nota inteira: 9
Digite uma nota inteira: 5
Digite uma nota inteira: 7
```

```
Média da Classe é 6,1
```

Saída do programa

Estudo de Caso 2: Repetição Controlada por Sentinela

- Desenvolva um programa de média de classe que calcule a média de um número arbitrário de notas cada vez que o programa é executado

Refinamento *Top-Down* Gradual

- Começamos com uma descrição em pseudocódigo do topo
 - *Determine a média da classe obtida pela resolução do problema*
- Primeiro refinamento:
 - *Inicializar as variáveis*
 - *Obtenha a entrada, some e conte as notas do problema*
 - *Calcule e imprima a média classe*

Refinamento *Top-Down* Gradual

- Segundo refinamento:
 - *Inicialize total com zero*
 - *Inicialize contador com zero*

 - *Introduza a primeira nota (possivelmente a sentinela)*
 - *Enquanto o usuário não introduzir a sentinela*
 - *Adicione esta nota ao total corrente*
 - *Adicione um ao contador de notas*
 - *Introduza a próxima nota (possivelmente a sentinela)*

 - *Se o Controlador não é igual a zero*
 - *Atribua o total dividido pelo contador à média*
 - *Imprima a média*
 - *Senão*
 - *Imprima “Nenhuma nota foi introduzida”*

```

' Media2.vb
' Usando uma repetição controlada por sentinela para
' calcular a média da sala.
Module modMedia2
    Sub Main()
        Dim total As Integer          ' soma das notas
        Dim contador As Integer       ' quantidade de notas digitadas
        Dim nota As Integer           ' nota digitada pelo usuário
        Dim media As Double           ' média de todas a notas
        ' fase de inicialização
        total = 0                      ' zerar total
        contador = 0                  ' zerar contador
        ' fase de processamento
        ' pedir para o usuário digitar uma nota e ler nota
        Console.WriteLine("Digite uma nota inteira, ou -1 para sair: ")
        nota = Console.ReadLine()
        ' loop controlado por sentinela, onde -1 é a sentinela
        While nota <> -1
            total += nota              ' adiciona a nota ao total
            contador += 1              ' adiciona 1 ao contador
            ' pedir para o usuário digitar nota e ler nota
            Console.WriteLine("Digite uma nota inteira, ou -1 para sair: ")
            nota = Console.ReadLine()
        End While
        ' fase de finalização
        If contador <> 0 Then
            media = total / contador
            ' mostrar a média da sala
            Console.WriteLine()
            Console.WriteLine("A média da sala é {0:F}", media)
        Else ' se nenhuma nota foi digitada
            Console.WriteLine("Nenhuma nota foi digitada")
        End If
        Console.ReadKey()
    End Sub ' Main
End Module ' modMedia2

```

Usando sentinela um valor precisa ser lido antes do While

O prompt deve lembrar o usuário de qual é o valor sentinela

```
Digite uma nota inteira, ou -1 para sair: 9
Digite uma nota inteira, ou -1 para sair: 7
Digite uma nota inteira, ou -1 para sair: 8
Digite uma nota inteira, ou -1 para sair: 6
Digite uma nota inteira, ou -1 para sair: 4
Digite uma nota inteira, ou -1 para sair: 3
Digite uma nota inteira, ou -1 para sair: 5
Digite uma nota inteira, ou -1 para sair: 10
Digite uma nota inteira, ou -1 para sair: 9
Digite uma nota inteira, ou -1 para sair: 7
Digite uma nota inteira, ou -1 para sair: 5
Digite uma nota inteira, ou -1 para sair: 6
Digite uma nota inteira, ou -1 para sair: 8
Digite uma nota inteira, ou -1 para sair: -1
```

```
A média da sala é 6,69
```

Saída do programa

Exercício

1. Os motoristas se preocupam com a quilometragem obtida em seus automóveis. Um motorista registrou vários abastecimentos de tanque de gasolina, anotando os quilômetros rodados e os litros usados em cada abastecimento. Desenvolva um programa que obtenha como entrada os quilômetros dirigidos e os litros usados (ambos como Double) para cada abastecimento. O programa deve calcular e exibir os quilômetros por litro de cada abastecimento e imprimir os quilômetros por litro combinados, obtidos para todos os abastecimentos. Todos os cálculos de média devem produzir resultados em ponto flutuante.

Estudo de Caso 3: Estruturas de Controle Aninhadas

- Uma instituição oferece um curso preparatório para o exame de Certificação Cisco CCNA. No último ano, 10 estudantes que fizeram o curso prestaram o exame. A instituição deseja saber o quão bem seus alunos foram no exame. Foi solicitado que você escreva um programa que resuma os resultados, e assim recebeu uma lista de dez estudantes. Ao lado de cada nome está escrito um “A” de Aprovado ou um “R” de Reprovado.
- Seu programa deve analisar os dados da seguinte forma:
 - Obtenha a entrada de cada resultado de exame (isto é, um A ou um R)
 - Conte o número de aprovados reprovados
 - Exiba um resumo dos resultados, indicando o número de aprovados e reprovados
 - Se mais que 8 estudantes foram aprovados imprima a mensagem “Aumente o preço do curso”

Analizando o programa

- O programa deve ter um loop para ler o conceito de dez estudantes
- Cada resultado será uma string “A” ou “R”
- Dois contadores armazenam os resultados do exame
 - Um conta os aprovados
 - O outro conta os reprovados
- Depois de processar todos os exames o programa deve verificar se mais do que 8 alunos foram aprovados

Refinamento Top Down:

- Pseudocódigo do topo:
 - *Analise os resultados do exame e decida se o custo do curso deve ser aumentado*
- Primeiro refinamento:
 - *Inicialize as variáveis*
 - *Obtenha a entrada das dez notas de exame e conte os aprovados e reprovados*
 - *Imprima um resumo dos resultados do exame e decida se o custo do curso deve ser aumentado*

Refinamento Top Down:

- Segundo refinamento:
 - *Inicialize os aprovados com zero*
 - *Inicialize os reprovados com zero*
 - *Inicialize o contador de estudantes com 1*

 - *Enquanto o contador de estudantes for menor ou igual a dez*
 - *Obtenha a entrada do próximo resultado de exame*
 - *Se o estudante foi aprovado*
 - *Adicione 1 aos aprovados*
 - *Senão*
 - *Adicione 1 aos reprovados*
 - *Adicione 1 ao contador de estudantes*

 - *Imprima o número de aprovados*
 - *Imprima o número de reprovados*
 - *Se mais do que oito estudantes foram aprovados*
 - *Imprima “Aumente o preço do curso”*

```

' modAnalise.vb
' Usando uma repetição controlada por controlador
' para mostrar resultados do exame.
Module modAnalise
    Sub Main()
        Dim aprovados As Integer = 0           ' número de aprovados
        Dim reprovados As Integer = 0         ' número de reprovados
        Dim estudante As Integer = 1          ' contador de estudantes
        Dim resultado As String               ' resultado de um exame
        ' processar os 10 resultados, através de um loop com contador
        While estudante <= 10
            Console.WriteLine("Digite o resultado (A = aprovado, R = reprovado): ")
            resultado = Console.ReadLine()
            ' estruturas de controle aninhadas
            If resultado = "A" Then
                aprovados += 1                ' incrementar o número de aprovados
            Else
                reprovados += 1              ' incrementar o número de reprovados
            End If
            estudante += 1                    ' incrementar o contador de estudantes
        End While
        ' mostrar resultados do exame
        Console.WriteLine("Aprovados: {0}{1}Reprovados: {2}", aprovados, _
            vbCrLf, reprovados)
        ' aumentar preço se mais de 8 estudantes passaram
        If aprovados > 8 Then
            Console.WriteLine("Aumentar o preço do curso")
        End If
        Console.ReadKey()
    End Sub ' Main
End Module ' modAverage

```

Estrutura If/Else aninhada dentro da estrutura While

Retorno de carro com mudança de linha

```
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): R
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): R
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): A
Digite o resultado (A = aprovado, R = reprovado): A
Aprovados: 8
Reprovados: 2
```

Saída do programa

Exercício

2. Modifique o programa do Estudo de Caso 3 para aceitar as letras “A”, “a”, “R” e “r”. Se qualquer outra entrada String for digitada, uma mensagem deve ser exibida informando ao usuário que a entrada é inválida. Apenas incrementa o contador se uma das quatro Strings mencionadas anteriormente for introduzida

Referências Bibliográficas

- MSDN: <http://msdn.microsoft.com/vstudio/>
- [DEITEL, Harvey M.; DEITEL, Paul J.; NIETO, Tem R. *Véual Basic.NET: Como Programar*. Prentice-Hall, 2004](#)