Lógica de Programação

Aula 06

Professor: Henrique Augusto Maltauro

Fecomércio PR
Sesc Senac IFPD
Senac

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

A **estrutura de decisão** escolha-caso, vai definir um **bloco de código** que que irá validar se uma **variável é igual a alguma das opções descritas**.

Nessa **estrutura de decisão**, é possível somente a validação de igual.

Para definir essa **estrutura de decisão** é necessário duas estruturas.

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

A primeira estrutura é definida pela **palavra reservada escolha**, seguida dos **delimitadores de parênteses ( )**, o qual deve receber a **variável** que será validada.

Seguido dos **delimitador de chaves { }**, para definir o **bloco de código**.

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

```
inteiro numero
leia(numero)
escolha (numero) {
    // Bloco de código
}
```

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

A segunda estrutura é definida de forma um pouco mais complexa:

1. Primeiramente, pela **palavra reservada** caso.
2. Seguida do **valor** que será utilizado para **validar a variável**.
3. Seguida do símbolo de **dois pontos**.
4. Seguido dos **comandos** que você deseja **executar naquela opção**.
5. Seguido da **palavra reservada** pare, para indicar que a **opção chegou ao fim**.

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

```
inteiro numero
leia(numero)
escolha (numero) {
    caso 1:
        escreva("Digitado o número 1")
        pare
}
```

Pseudocódigo

- Estruturas de Decisão: escolha-caso

Essa segunda estrutura, pode ser repetida várias vezes.

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

```
inteiro numero
leia(numero)
escolha (numero) {
    caso 1:
        escreva(“Digitado o número 1”)
        pare
    caso 2:
        escreva(“Digitado o número 2”)
        pare
}
```

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

No possibilidade de a execução do código não cair em nenhuma daquelas opções, nós fazemos novamente o uso da segunda estrutura, porém **no lugar do valor utilizado para validar a variável**, utilizamos a **palavra reservada contrario**.

Pseudocódigo

- **Estruturas de Decisão: escolha-caso**

```
inteiro numero
leia(numero)
escolha (numero) {
      caso 1:
            escreva("Digitado o número 1")
            pare
      caso 2:
            escreva("Digitado o número 2")
            pare
      caso contrario:
            escreva("Opção Inválida!")
            pare
}
```

Pseudocódigo

- **Estruturas de Repetição**

Dentro da **programação**, nós temos as **estruturas de repetição**, que determinam um **bloco de código** que será executado uma ou mais vezes de acordo com decisões tomadas por **condições lógicas**.

Pseudocódigo

- **Estruturas de Repetição**

Temos 3 **estruturas de repetição**:

1. **enquanto**
2. **faca-enquanto**
3. **para**

Pseudocódigo

- **Estruturas de Repetição: enquanto**

A **estrutura de repetição** enquanto, vai definir um **bloco de código** que será executado **enquanto** uma **condição lógica for verdadeira**.

Para definir essa **estrutura de repetição** é necessário a **palavra reservada** enquanto, seguida de dois **delimitadores**.

Pseudocódigo

- **Estruturas de Repetição: enquanto**

1. **Delimitador de parênteses ( ), o qual é utilizado para definir a condição lógica que será validada.**
2. **Delimitador de chaves { }, para definir o bloco de código da estrutura de repetição.**

Pseudocódigo

- **Estruturas de Repetição: enquanto**

```
inteiro numero = 0
enquanto (numero <= 10) {
    // Bloco de código
}
```

Muito Cuidado Aqui!!

Fecomércio PR
Sesc Senac IFPD

Senac

Pseudocódigo

- **Estruturas de Repetição: Loop Infinito**

Em todas as **estruturas de repetição**, mas mais precisamente no **enquanto** e no **faca-enquanto**, deve se ter cuidado para não criar um **loop infinito**.

O **loop infinito** é uma **estrutura de repetição** que **nunca para de repetir**.

Isso pode acontecer se, a **condição da estrutura de repetição não for bem controlada**.

Pseudocódigo

- **Estruturas de Repetição: Loop Infinito**

Para evitarmos o **loop infinito**, devemos sempre pensar a frente, e garantir que em algum momento das repetições, **aquela condição** que foi definida para a repetição, **retornará um valor falso**.

Pseudocódigo

- Estruturas de Repetição: enquanto

```
inteiro numero = 0
enquanto (numero <= 10) {
    numero++
}
```

Pseudocódigo

- **Estruturas de Repetição: faca-enquanto**

A **estrutura de repetição** do **faca-enquanto**, vai ser muito similar a estrutura do **enquanto**, e também vai definir um **bloco de código** que será executado **enquanto** uma **condição lógica for verdadeira**, mas com algumas diferenças.

A definição dessa **estrutura de repetição** é um pouquinho mais elaborada.

Pseudocódigo

- **Estruturas de Repetição: faca-enquanto**

1. Primeiramente, precisa da **palavra reservada faca**.
2. Seguido do **delimitador de chaves { }**, para definir o **bloco de código** da **estrutura de repetição**.
3. Seguida da **palavra reservada enquanto**.
4. Seguido do **delimitador de parênteses ( )**, para definir a **condição lógica** que será validada.

Pseudocódigo

- **Estruturas de Repetição: faca-enquanto**

```
inteiro numero = 0
faca {
    // Bloco de código
}
enquanto (numero <= 10)
```

Pseudocódigo

- **Estruturas de Repetição: enquanto vs faca-enquanto**

E no que essas duas estruturas de repetição se diferenciam?

Basicamente, o **enquanto** vai **validar a condição lógica antes** de executar o **bloco de código**.

Já o **faca-enquanto** vai **executar o bloco de código primeiro**, e só **depois** vai **validar a condição lógica**.

Pseudocódigo

- **Estruturas de Repetição: enquanto vs faca-enquanto**

E no que isso influencia a execução do código?

Imaginemos que a execução do código chegue até a **estrutura de condição**, e já de cara a **condição lógica** dela retorna um **resultado falso**.

No **enquanto**, a estrutura não será executada.

Já no **faca-enquanto**, a estrutura será executada uma única vez, e depois não vai mais ser executada.

Pseudocódigo

- **Estruturas de Repetição: para**

E por fim, a **estrutura de repetição** para, vai definir um **bloco de código** que será executado utilizando uma **condição lógica** incremental.

Muito comum, quando se sabe exatamente a quantidade de vezes que se quer repetir aquele **bloco de código**.

A definição dessa **estrutura de repetição** também é um pouquinho mais elaborada.

Pseudocódigo

- **Estruturas de Repetição: para**

1. Primeiramente, é usado a **palavra reservada** para.
2. Seguido do **delimitador de parênteses ( )**, para definir três **declarações**, separadas por **ponto e vírgula**.
    a. Na primeira declaração, é **criada uma variável de controle**, que **será executado uma única vez**.
    b. Na segunda declaração, **definido a condição de repetição usando a variável de controle**.
    c. Na terceira declaração, é **definido o processo de controlar essa variável**, e que será **executado toda vez que a estrutura for repetida**.
3. Seguido do **delimitador de chaves { }**, para definir o **bloco de código** da **estrutura de repetição**.

Pseudocódigo

- **Estruturas de Repetição: para**

```
para (inteiro i = 0; i <= 10; i++) {
    // Bloco de código
}
```

Exercício

Exercício

gg.gg/LogicaSenac06

github.com/hmaltaurodev/Slides

Fecomércio PR
Sesc Senac IFPD
Senac