LÓGICA DE PROGRAMAÇÃO

Esp. Marcelo Uemura

INICIAR

Introdução

Nesta unidade, serão apresentadas as estruturas de repetição utilizadas por algoritmos e linguagens de programação. Essas estruturas permitem que blocos de instruções possam ser repetidos, sem a necessidade de replicar o código.

Para tanto, é necessário que condições de teste ou limites sejam estabelecidos, para que não recaia em uma condição de repetição infinita. Essas condições podem ser testadas no início, no fim, ou através de variáveis de controle.

Os algoritmos vistos para estruturas de repetição, ENQUANTO-FAÇA, REPITA-ATÉ e PARA-ATÉ-FAÇA, serão aqui revistos e um paralelo com as instruções da linguagem de programação C (*while*, *for* e *do*) será utilizado para entender como podem ser aplicados na prática.

Bons estudos!

Estrutura de Repetição

Neste tópico, serão identificados os princípios básicos das formas de construção de programas, usando laço de repetição com teste no início. Com base nisso, o aluno deverá estar apto a desenvolver programas com repetições e avaliar soluções de programas usando laço de repetição com testes no início.

Conceitos de contadores e acumuladores

Vilarim (2004) menciona que uma variável contadora tem por característica armazenar um número referente a uma determinada quantidade de elementos ou interações.

Em um laço de repetição, é comum utilizarmos os recursos de contadores, que consiste em uma variável com valor inicial, que é incrementado a cada repetição executada de um bloco de instruções. O incremento é, basicamente, a soma de um valor constante, normalmente 1 (FORBELLONE; EBERSPÄCHER, 2005).

Por exemplo:

1. início

2. inteiro: contador;

3. enquanto contador < 100 faça

a. ***contador = contador +1;***

b. escreva (contador);

4. fimenquanto;

5. fim

Verificamos que a variável contador é utilizada para contagem em um laço de repetição de estrutura ENQUANTO-FAÇA. A variável é incrementada a cada interação, realizando, assim, uma função de contador. Caso esse incremento não ocorresse, a repetição não teria fim, pois a condição contador < 100 seria sempre verdadeira.

Outro tipo de variável utilizado em repetições é a acumuladora, que armazena uma série de valores, em geral uma soma, conforme Vilarim (2004). Em uma repetição, a função acumuladora permite a soma de diversos números, sucessivamente.

Exemplo:

1. início

2. real: numero, acumulador;

3. numero = 0;

4. acumulador = 0;

5. enquanto numero < 100 faça

a. ***acumulador = acumulador + 10;***

b. numero = numero + 1;

6. fimenquanto;

7. fim

Nesse exemplo, a variável acumulador vai somando sucessivamente, durante as repetições, o seu valor anterior com 10, obtendo um novo valor, até que a repetição encerre com o fim da contagem pela variável numero.

A seguir, teremos a apresentação de uma estrutura de repetição com teste no início do laço.

Estruturas de repetição com teste no início

Segundo Souza et al. (2011) a estrutura ENQUANTO-FAÇA permite a execução repetitiva de comandos, enquanto a condição de controle para a repetição for verdadeira. Essa condição é testada no início do laço.

1. ***enquanto*** *(condição)* ***faça***

a. ação 1

b. ação 2

c. ação N

2. fimenquanto;

Exemplo de uso da estrutura ENQUANTO-FAÇA: escrever um algoritmo que faça a soma de 100 números inteiros lidos.

1. início

2. inteiro: contador, soma, numero;

3. contador = 0;

4. soma = 0;

*5. **enquanto** (contador < 100) **faça***

a. leia (numero);

b. soma = soma + numero;

c. contador = contador + 1;

6. fimenquanto;

7. escreva ("a soma dos números é" soma);

8. fim

No algoritmo acima, são usadas a variável contador para contagem e a variável soma como acumulador. Os números lidos são acumulados a cada repetição em que um novo valor é lido.

Na linguagem de programação C, a instrução para realizar a estrutura de repetição ENQUANTO-FAÇA é:

***while** (condição)*

{

bloco de instruções;

}

A seguir, temos o fluxograma correspondente à instrução *while*:

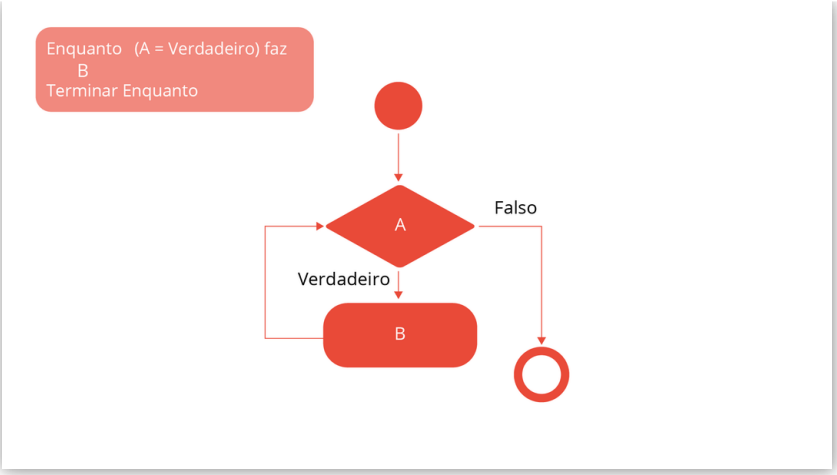


Figura 3.1 - Fluxograma while
Fonte: Zanelato (2018).

Adaptando o algoritmo deste tópico para a linguagem estruturada em C, teremos o seguinte código:

```
#include <stdio.h>

int main()

{

    int contador, soma, numero;

    contador = 0;

    soma = 0;

    while (contador < 100)

    {

        printf("Entre com um número inteiro:");

        scanf(%d, &numero);

        soma = soma + numero;

        contador = contador + 1;
```

```
    }

    printf("a soma dos números é: %d", soma);

    return 0;

}
```

Nesse código, temos o uso da instrução *while* como teste no início contador < 100. A variável contador continua sendo aplicada para contagem e soma como acumulador.

Atividade

As estruturas de repetição com teste no início (ou pré-teste), indicam que um laço só será executado se a condição for satisfeita. Para tanto, na linguagem de programação em C, tendo como base a estrutura de repetição ENQUANTO-FAÇA, deve ser utilizada no início do laço a instrução:

- ○ **a)** Printf
- ○ **b)** Scanf
- ○ **c)** For
- ◉ **d)** While
- ○ **e)** Do-while

Feedback: A alternativa **correta é a letra D**, A instrução while é utilizada para laços de repetição com teste no início, tendo como base a estrutura ENQUANTO-FAÇA. As instruções scanf e printf correspondem a comandos de entrada e de saída, respectivamente. A instrução for é utilizada para laços de repetição baseados na estrutura PARA-ATÉ-FAÇA e a instrução do-while para estrutura REPITA-ATÉ.

Estrutura de Repetição

Neste tópico, serão identificados os princípios básicos das formas de construção de algoritmos usando laço de repetição com teste no final e variável de controle. Também deverão ser desenvolvidos algoritmos com repetição com teste no final e variáveis de controle, analisando o desempenho dessas estruturas de repetição.

3.2.1 Estrutura de repetição com teste no final - do-while

Essa estrutura de repetição segue o modelo REPITA-ATÉ, em que é assegurado que o bloco de instruções seja executado ao menos uma vez, caso a condição de teste no final já seja satisfeita. Caso contrário, o bloco será novamente executado.

Estrutura REPITA-ATÉ

1. ***repita***

    a. ação 1;

    b. ação 2;

c. ação 3;

d. ação n;

*2. **até** (condição de teste);*

Exemplo de uso da estrutura repita-até: algoritmo para fazer a soma dos números pares de 1 até 100.

1. início

2. inteiro: numero, soma;

3. soma = 0;

4. numero = 1;

5. repita

a. se (numero resto 2 = 0) então

i. soma = soma + numero;

ii. numero = numero +1;

6. até (numero > 100);

7. escreve (soma);

Nesse exemplo, a variável numero é inicializada com 1, e a variável soma, com 0. A partir de 1, a primeira interação verifica que o resto da divisão de 1 por 2 não é zero, logo, o número é ímpar. Esse número não é adicionado à variável soma, e a variável numero é incrementada no término de execução do laço. Como a variável numero ainda não chegou a 100, o laço é repetido. Na segunda interação, a variável numero tem o valor 2, que é par, pois o resto da divisão por 2 é zero, e é adicionado ao valor da soma. Essas repetições ocorrerão até que o incremento da variável numero chegue em 101 (numero > 100), quebrando o laço de repetição.

Na linguagem C, como outras linguagens estruturadas, utilizam os comandos do-while. A sintaxe utilizada é a seguinte/;

1. ***do*** *{*

    a. bloco de instruções

2. *}*

3. ***while*** *(condição);*

No formato de fluxograma, a estrutura *do-while* ficaria como a figura a seguir:

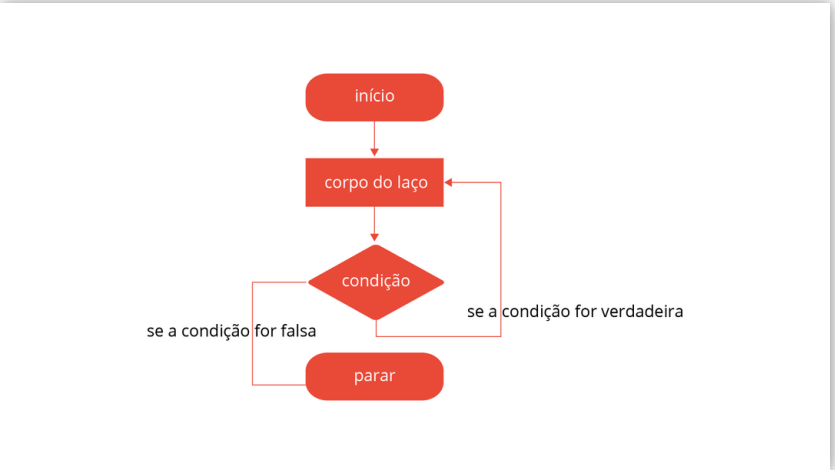


Figura 3.2 - Fluxograma do-while
Fonte: Zanelato (2018).

Utilizando o mesmo algoritmo deste tópico, relacionado à soma de números pares de 1 a 100, podemos fazer o equivalente, utilizando a programação em C, conforme segue:

```
#include <stdio.h>

int main()

{

    int numero, soma;
```

```
    numero = 1;
    soma = 0;
     do
    {
        if (numero % 2 == 0)
       {
              soma = soma + numero;
              numero = numero +1;
       }
       while (numero <= 100);
     }
     printf("A soma dos números pares de 1 a 100 é: %d\n", soma);
    return 0;
}
```

No código acima, o comando *do-while* é utilizado seguindo a filosofia da estrutura FAÇA-ATÉ. A seguir, veremos outra estrutura de repetição de laço, em que uma variável de controle é utilizada, bem como limitando os valores iniciais e finais.

Atividade

Analise o seguinte trecho de código, que traz uma implementação do laço do-while:

```
int contagem;

contagem = 0;

do

{

    contagem = contagem + 1;

    printf (“O número agora é: %d\n”, contagem);

}

while (contagem < 100);
```

A contagem se repetirá até que chegue ao valor:

- ○ **a)** 100
- ○ **b)** 101
- ◉ **c)** 99
- ○ **d)** 99,9
- ○ **e)** 100,1

Feedback: A alternativa **correta é a letra C**, a condição de teste para finalizar o laço de repetição é quando a contagem for menor do que 100, ou seja, quando ocorrer a execução do bloco em que a contagem será incrementada para 100.

Implementação de Algoritmos com Estruturas de Repetição

Neste tópico, serão identificados os princípios básicos das formas de construção de programas usando laço de repetição for com teste no início em uma Linguagem de Programação. Serão apresentados o desenvolvimento de programas com repetições e avaliadas soluções de programas escritos em uma Linguagem de Programação.

Estrutura de repetição com teste no início, teste no final e variável de controle

Em uma estrutura de repetição, é possível utilizar uma variável de controle com limites pré-definidos (FORBELLONE; EBERSPÄCHER, 2005). Essa estrutura é tratada como PARA-ATÉ-FAÇA, como pode ser verificado a seguir:

1. ***para*** *V de vi* ***até*** *vf passo p* ***faça***

    a. ação 1;

    b. ação 2;

c. ação N;

2. fimpara;

No modelo acima, as ações são executadas dentro de um laço, que deve ser repetido desde um valor inicial (vi) para a variável de controle (V) até um valor final (vf), dentro de passos de incremento p.

Um exemplo para o uso desse modelo pode ser dado com o exemplo a seguir, em que deve ser calculada a média da idade em uma turma de 30 alunos.

1. início

a. real: idade, media, soma;

b. inteiro: controle;

c. soma = 0;

d. para controle de 1 até 30 passo 1 faça

i. leia (idade);

ii. soma = soma + idade;

e. fimpara;

f. media = soma divide 30;

2. fim

No algoritmo acima, é realizada a soma das idades dos 30 alunos, seguindo um laço que repete desde o primeiro até o último aluno (30), um a um. No final, divide por 30 para encontrar a média.

Utilizando uma linguagem de programação C, temos a instrução for, que pode ser utilizada para a estrutura PARA-ATÉ-FAÇA. A sintaxe para essa instrução é como a que segue:

```
for (contador=valor inicial; contador <=valor final; contador++)

{

    bloco de instruções

}
```

O fluxograma que simboliza a instrução for é como se segue:

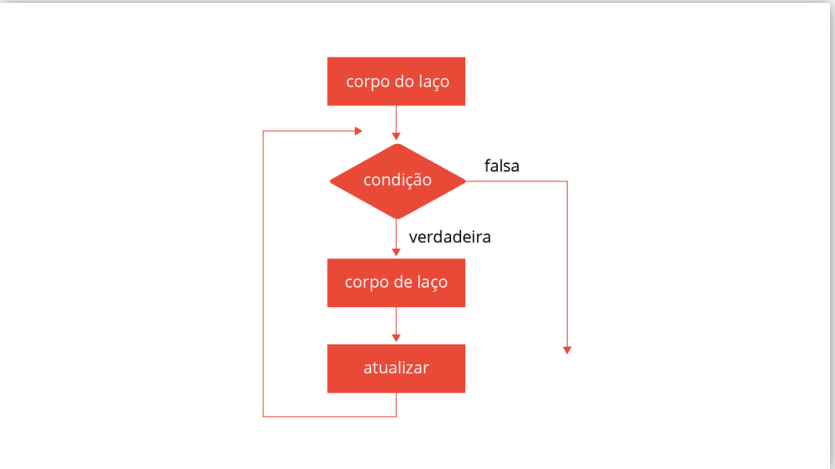


Figura 3.3 - Fluxograma for
Fonte: Zanelato (2018).

Enquanto a condição for não for atingida, o laço deve ser repetido. Se adequarmos o algoritmo apresentado neste tópico para a linguagem C, teremos o seguinte código:

```
#include <stdio.h>

int main()

{

    int aluno;

    float soma, idade, media;
```

```
 soma = 0;
for (aluno=1; aluno <=30; aluno++)
{
        printf("Escreva o nome da idade do aluno:");
        scanf("%d", &n);
        soma = soma + idade;
}
media = soma/aluno;
printf("Media da idade dos alunos: %d\n", media);
return 0;
}
```

No código acima, a variável de controle será “aluno”, que terá os limites de 1 até 30, incrementando no passo de 1. Dentro das repetições, a variável “soma” é adicionada com a idade de cada aluno. Ao final, após finalizar a soma da idade dos 30 alunos, a variável “media” é calculada.

Atividade

No trecho de código a seguir:

```
int i, a, b;

a = 0;

b = 0;

for (i = 0; i<=5; i++)

{

    b = a+b;

    a = a+2;

}
```

O valor de b após a execução das repetições é:

- ○ **a)** 12
- ○ **b)** 30
- ○ **c)** 20
- ◉ **d)** 6
- ○ **e)** 4

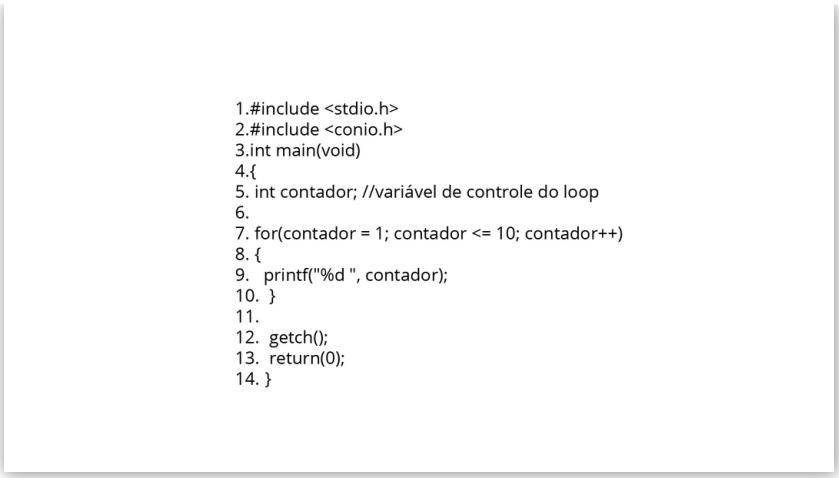
Feedback: A alternativa **correta é a letra B**, após 5 interações, a variável b vai adicionando o valor de a, que é incrementado com 2 a cada interação. As outras alternativas não apresentam o valor correto de b para o algoritmo apresentado.

Implementação de Algoritmos com Estrutura de Repetição

Neste tópico, serão tratados os princípios básicos das formas de construção de programas utilizando laços de repetição, objetivando ter a base para o desenvolvimento de programas com repetições e avaliar soluções de programas escritos em uma linguagem de programação.

Análise entre for, *while* e *do-while*

O uso da instrução *for* segue a estrutura PARA-ATÉ-FAÇA. Nesta estrutura de repetição, o *for* utiliza uma variável de controle, que determinará o início e o fim das repetições, bem como o passo para o incremento dessa variável. Esse tipo de laço também é conhecido como laço incondicional (MANZANO; OLIVEIRA, 2010), pois não existe uma definição de condição de teste para interrupção das repetições. Logo, esse seria um critério para o desenvolvedor no uso dessa estrutura de repetição, em que o valor de início e de fim são determinados para uma variável de controle.



```
1.#include <stdio.h>
2.#include <conio.h>
3.int main(void)
4.{
5. int contador; //variável de controle do loop
6.
7. for(contador = 1; contador <= 10; contador++)
8. {
9.   printf("%d ", contador);
10.  }
11.
12.  getch();
13.  return(0);
14. }
```

Figura 3.4 - Exemplo de for
Fonte: Casavella (2019).

No exemplo acima, a variável contador está sendo utilizada como variável de controle, e a execução sairá da repetição quando esta chegar ao valor de 10.

reflita

Reflita

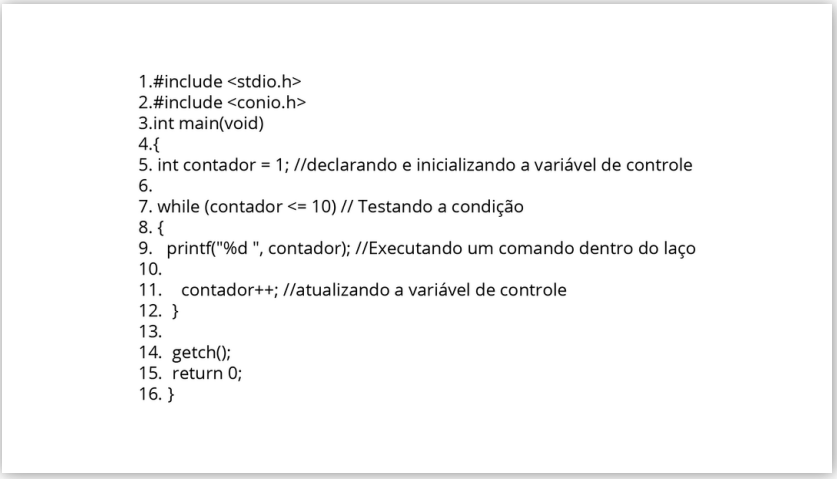
As estruturas de repetição permitem que blocos de instruções possam ser executados de forma repetitiva, através de condições que podem ser testadas no início, no fim ou por meio de valores limitados. Reflita sobre os cuidados que devem ser tomados quanto ao uso de cada tipo de estrutura de repetição.

Fonte: Elaborada pelo autor.

No caso da instrução *while*, o teste é realizado no início, seguindo a estrutura ENQUANTO-FAÇA. Isso indica que o bloco de instruções só será executado se a condição assim permitir. Por outro lado, a instrução *do-while* coloca o teste após a execução do bloco de instrução, conforme a estrutura REPITA-ATÉ, em

que o teste é feito no final (VILARIM, 2004). Com base nesse critério, o desenvolvedor pode definir qual estrutura utilizar, de acordo com a posição do teste da condição, se no início ou no fim da execução de um bloco de instruções que será repetido.

Para o uso da estrutura *while*, segue um exemplo de código em linguagem de programação C:

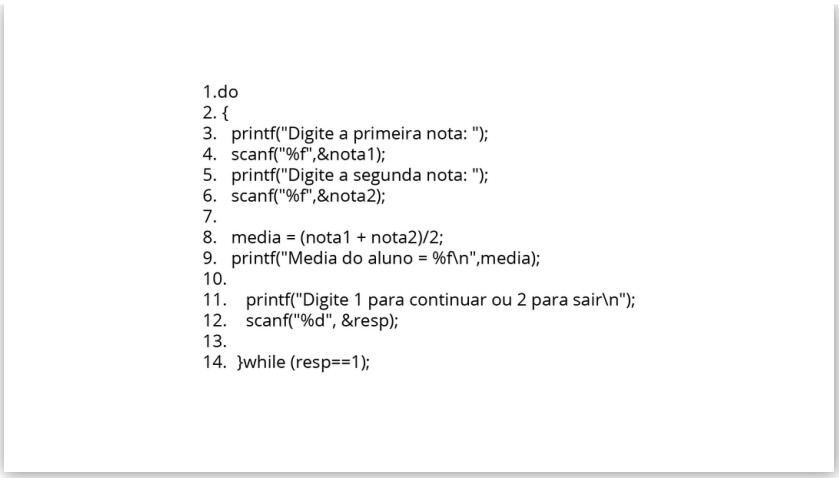


```
1.#include <stdio.h>
2.#include <conio.h>
3.int main(void)
4.{
5. int contador = 1; //declarando e inicializando a variável de controle
6.
7. while (contador <= 10) // Testando a condição
8. {
9.   printf("%d ", contador); //Executando um comando dentro do laço
10.
11.    contador++; //atualizando a variável de controle
12.  }
13.
14.  getch();
15.  return 0;
16. }
```

Figura 3.5 - Exemplo de while
Fonte: Casavella (2019).

Nesse exemplo, o uso da variável contador apresenta um teste já no início, antes do bloco de execução.

Um exemplo de uso para estrutura *do-while* pode ser em casos em que os laços são repetidos até que o usuário decida finalizar.



```
1.do
2. {
3.   printf("Digite a primeira nota: ");
4.   scanf("%f",&nota1);
5.   printf("Digite a segunda nota: ");
6.   scanf("%f",&nota2);
7.
8.   media = (nota1 + nota2)/2;
9.   printf("Media do aluno = %f\n",media);
10.
11.    printf("Digite 1 para continuar ou 2 para sair\n");
12.    scanf("%d", &resp);
13.
14.  }while (resp==1);
```

Figura 3.6 - Exemplo de do-while
Fonte: Casavella (2019).

Nesse exemplo, a solicitação de dados (nota 1 e nota 2), bem como o cálculo da média será repetido até que o usuário digite 2, saindo do laço.

Em síntese, repetições com *for* utilizam variável de controle, repetições com *while* realizam testes no início e repetições do tipo *do-while* adotam testes no final do laço.

saiba mais

Saiba mais

Python é uma linguagem que está se tornando muito popular e atrativa para programadores. As estruturas de repetição também estão presentes nessa linguagem de programação, e no link a seguir é apresentado como podem ser utilizadas.

Fonte: Devmedia (2019, on-line).

ACESSAR

Atividade

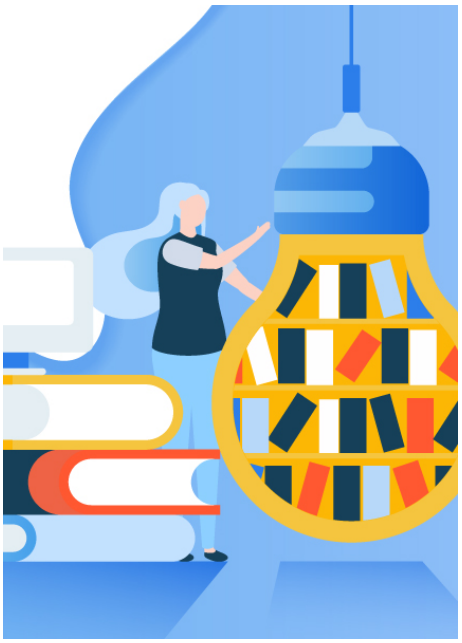
Nas estruturas de repetição PARA-ATÉ-FAÇA, como a usada na instrução em C for, devem ser definidos os limites de início e fim para os quais a repetição deve ocorrer, bem como o passo. O nome da variável utilizada para tal finalidade é:

- (●) **a)** Variável de controle;
- ( ) **b)** Variável contadora;
- ( ) **c)** Variável acumuladora;
- ( ) **d)** Variável global;
- ( ) **e)** Variável local.

Feedback: A alternativa **correta é a letra A**, é por meio de uma variável de controle que a instrução *for* controla as repetições do laço. A variável contadora é utilizada para contagem dentro do laço, e a acumuladora, para acumular valores. As variáveis global e local não estão relacionadas para uso específico na instrução *for*.

indicações

Material Complementar



LIVRO

Algoritmos - Lógica para Desenvolvimento de Programação de Computadores

José Augusto N. G. Manzano e Jayr Figueiredo de Oliveira

Editora: Érica

ISBN: 978-85-365-0221-2

Comentário: esse é um livro de técnicas de programação, com exemplos de algoritmos. Reforça o estudo realizado acerca de estruturas de repetição e contém alguns programas exemplos codificados em Pascal, Basic, C e C++.

WEB

Laços de Repetição

Ano: 2017

Comentário: esse vídeo proporciona um reforço acerca de algoritmos de repetição por meio do uso de Portugol, de forma prática e interessante.

Para saber mais sobre o vídeo, acesse o link.

ACESSAR

conclusão

Conclusão

Nesta unidade, vimos o uso de estruturas de repetição para a lógica de programação. Essas estruturas permitem que blocos de instruções possam ser repetidos de acordo com critérios que definam os limites.

A estrutura de repetição ENQUANTO-FAÇA, tem, na linguagem de programação C, a instrução *while*, permitindo que a condição de teste seja realizada no início do bloco de instruções do laço, enquanto na REPITA-ATÉ, com a instrução *do-while*, o teste da condição é realizado no final do laço.

Por fim, foi apresentada a estrutura incondicional PARA-ATÉ-FAÇA, em que é utilizada a instrução *for* nas linguagens de programação, tendo uma variável de controle, que é utilizada para a limitação das repetições.

referências

Referências Bibliográficas

CASAVELLA, E. Comando do while em C. **Intelectualle Tecnologia e Treinamento**. Disponível em: <http://linguagemc.com.br/comando-do-while/>. Acesso em: 3 maio 2019.

_____. O comando for em C. **Intellectuale Tecnologia e Treinamento**. Disponível em: <http://linguagemc.com.br/a-estrutura-de-repeticao-for-em-c/>. Acesso em: 3 maio 2019.

ESTRUTURAS de condição e repetição em Python. **Devmedia**, 2016. Disponível em: <https://www.devmedia.com.br/estruturas-de-condicao-e-repeticao-em-python/37158>. Acesso em: 3 maio 2019.

FARRER, H. BECKER, C. G.; FARIA, E. C.; MATOS, H. F.; SANTOS, M. A.; MAIA, M. L. **Algoritmos estruturados**. Rio de Janeiro: LTC, 2013.

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. **Lógica de Programação**: a construção de algoritmos e estruturas de dados. São Paulo: Prentice Hall, 2005.

MANZANO, J. A. N. G.; OLIVEIRA, J. F. **Algoritmos**: lógica para Desenvolvimento de Programação e Computadores. São Paulo: Erica, 2010.

SOUZA, M. A. F.; GOMES, M. M.; SOARES, M. V.; CONCILIO, R. **Algoritmos e Lógica de Programação**. São Paulo: Cengage Learning, 2011.

VILARIM, G. **Algoritmos**: programação para iniciantes. Rio de Janeiro: Editora Ciência Moderna, 2004.

ZANELATO, J. Lógica de programação: estruturas de repetição. **Pod Programar**, fev. 2018. Disponível em: <https://podprogramar.com.br/logica-de-programacao-estruturas-de-repeticao/>. Acesso em: 3 maio 2019.

IMPRIMIR