Disciplina: Introdução à Programação

Aula 8: Estrutura de repetição com variável de controle

Apresentação

Na aula anterior, começamos a estudar as estruturas repetitivas. Vimos que essas estruturas têm a função de repetir determinado trecho do código, e descobrimos que elas podem ser de três tipos diferentes: estrutura repetitiva com pré-teste, estrutura repetitiva com pós-teste e estrutura repetitiva com variável de controle.

Os dois primeiros tipos foram estudados na aula passada. Nesta aula, veremos como funcionam as estruturas do terceiro tipo e analisaremos de que maneira ela se difere das já estudadas.

Objetivos

- Descrever as estruturas de repetição com variável de controle;
- Diferenciar as estruturas de repetição com variável de controle daquelas com pré-teste ou pós-teste;
- Registrar algoritmos e programas utilizando a estrutura de repetição adequada.

Estrutura de repetição com variável de controle

Na aula anterior, aprendemos um novo tipo de estrutura de controle: as estruturas repetitivas.

Vimos que elas nos permitem repetir a execução de determinado trecho do código enquanto uma condição for verdadeira, e exploramos dois tipos diferentes de repetição:

as que testam a condição no **início**.

as que testam a condição no **final**.

Na aula de hoje, vamos ver um terceiro tipo de estrutura de repetição, que é o que utiliza uma variável de controle.

Este tipo de estrutura é muito utilizado quando se sabe, de antemão, o número de iterações a serem realizadas, ou seja, quantas vezes será necessário repetir o trecho definido dentro da estrutura.

Isso acontece porque essa estrutura inclui um contador em sua definição que será automaticamente incrementado ou decrementado.

Você se lembra da variável de controle que é extremamente necessária nos laços com pós e pré-teste? Nesse tipo de repetição, ela é controlada pela **estrutura**.

Vamos ver como é sua sintaxe no Portugol Studio e no C++?

```
Portugol Studio
para (inicialização_contador;condição;incremento_ou_decremento)
{
//Comandos a serem executados enquanto a condição for verdadeira.
}

* C++
for (inicialização_contador;condição;incremento_ou_decremento)
{
//Comandos a serem executados enquanto a condição for verdadeira.

}
```

A condição a ser testada é semelhante àquela que utilizamos nas estruturas de seleção, ou seja, uma expressão que, após avaliada, retornará o valor verdadeiro ou falso. Para entender melhor como funciona a estrutura repetitiva com variável de controle, vamos ver como ficaria o algoritmo para exibição dos números pares entre 10 e 100?

Observe:

Portugol Studio

```
1 programa
2 {
3  funcao inicio()
4  {
5   inteiro num
6   num=10
7   para (num=10;num<=100;num+=2)
8   {
9    escreva(num, "\n")
10   }
11  }
12}
```

A estrutura repetitiva começa na linha 7 e termina na linha 10. As chaves nas linhas 8 e 10 delimitam, respectivamente, o início e o fim do trecho a ser repetido, que está descrito na linha 9.

Ele será repetido enquanto a variável **num** estiver armazenando um valor menor ou igual a 100. Teste o exemplo no Portugol Studio e você verá que o resultado final será a exibição dos números pares entre 10 e 100, um em cada linha, já que usamos “\n”.

Observe que, ao contrário do que fizemos na aula anterior com as estruturas com pré e pós-teste, aqui não foi necessário incluir a alteração da variável de controle (**num**) dentro do loop, devido ao fato de o incremento (**num+=2**) ter sido definido na linha de definição da repetição (linha 7). Vamos ver como ficaria esse algoritmo escrito em C++?

*** Programa em C++**

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5   int num=10;
6   for (num=10;num<=100;num+=2)
7  {
8   cout <<num<<"\n";
9  }
10 }
```

• Você se lembra do problema da tabuada?
• Aquele no qual o usuário informava um número e o algoritmo precisava exibir a tabuada do mesmo?
• Vamos ver como ficaria a solução se utilizássemos a estrutura para?

Observe:

*** Portugol Studio:**

```
1 programa
2 {
3  funcao inicio()
4   {
5    inteiro num, contador, res
6    escreva("Você deseja ver a tabuada de que número? ")
7    leia (num)
8    para (contador=0;contador<=10;contador++)
9    {
10        res = num * contador
11     escreva(num, " x ", contador, " = ", res, "\n")
12    }
13   }
14 }
```

No exemplo anterior, a repetição é controlada pela variável contador. Veja que ela é incrementada e testada a cada iteração (repetição), no início da estrutura, antes que o trecho a ser repetido (linhas 10 e 11) seja efetivamente executado.

Já vimos que toda repetição precisa de uma variável de controle que seja alterada a cada passada pela repetição; o que é feito na linha 8 com a expressão contador++.

Atenção

Se você se esquecer de alterar o valor da variável de controle, a repetição não terá fim e o código ficará preso em um loop infinito.

O programa em C++ tem uma estrutura muito semelhante.

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5   int num, contador, res;
6   cout <<"Você deseja ver a tabuada de que número? ";
7   cin >>num;
8   for (contador=0;contador<=10;contador++)
9   {
10        res = num * contador;
11     cout <<num<<" x "<<contador<<" = "<<res<<"\n";{
12 }
13 }
```

Agora, imagine que você precisa receber a idade de cinco pessoas e, para cada uma delas, informar se ela já pode ou não tirar a carteira de motorista. Vamos ver como ficaria o algoritmo para solução desse problema?

```
* Portugol Studio:
programa
{
 funcao inicio()
  {
   inteiro num, contador
   para (contador=1;contador<=5;contador++)
   {
        escreva("Qual a ",contador,"a. idade? ")
     leia (idade)
     se (idade>=18)
     {
       escreva("Já pode tirar carteira de motorista!\n")
      }
     senao
     {
      escreva("Ainda não pode tirar carteira de motorista!\n")
     }
  }
 }
}
```

Observe que esse algoritmo utiliza tanto a estrutura seletiva **se** quanto a estrutura repetitiva **para**. Assim como vimos na aula anterior, as estruturas não são excludentes, ou seja, o fato de você utilizar uma delas não significa que não poderá usar nenhuma outra. Lembre-se de que é muito usual que os algoritmos e programas combinem as estruturas em suas soluções.

Atividade

1. Leia os enunciados a seguir e crie os algoritmos e os programas em C++ que solucionem os problemas propostos.
Receba uma sequência de 10 números inteiros e, ao final, exiba a quantidade de múltiplos de 3 que foi informada.
Receba 15 números reais e, ao final, informe o maior número da sequência.

Agora analise atentamente este algoritmo, que exibe a contagem regressiva de um limite fornecido pelo usuário até zero.

*** Portugol Studio:**

```
1 programa
2 {
3  funcao inicio()
4   {
5    inteiro regressiva, contador
6    escreva("Qual o valor para início da contagem regressiva? ")
7    leia(regressiva)
8    para (contador=regressiva;contador>=0;contador--)
9    {
10        escreva(contador, "\n")
11    }
12   }
13  }
```

Notou alguma diferença na linha 8, onde se inicia a estrutura repetitiva? Neste exemplo, a variável contador é inicializada com o valor informado pelo usuário e decrementada a cada iteração.

Vejamos como ficaria este algoritmo se transformado em um programa em C++:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5   int num, contador, res;
6   cout <<"Qual o valor para início da contagem regressiva? ";
7   cin >>regressiva;
8   for (contador=regressiva;contador>=0;contador--)
9   {
10        cout <<contador<<"\n";
11  }
12 }
```

Atividade

2. Leia o enunciado a seguir e escreva o algoritmo e o programa em C++ que solucionem o problema proposto.

a) Crie uma estrutura de repetição que se inicia em 1 e vai até um limite definido pelo usuário. Dentro dela, faça com que sejam contabilizados os números pares e os números ímpares que compõem a sequência. Ao final da repetição, exiba quantos números com estas características foram informados.

Também é possível utilizar as estruturas enquanto e faca..enquanto para criar uma repetição na qual o valor inicial da variável de controle seja maior do que o valor final e na qual essa variável seja decrementada a cada iteração. Quer experimentar? Então, reescreva o algoritmo e o programa da contagem regressiva utilizando uma estrutura de repetição com pré-teste ou com pós-teste.

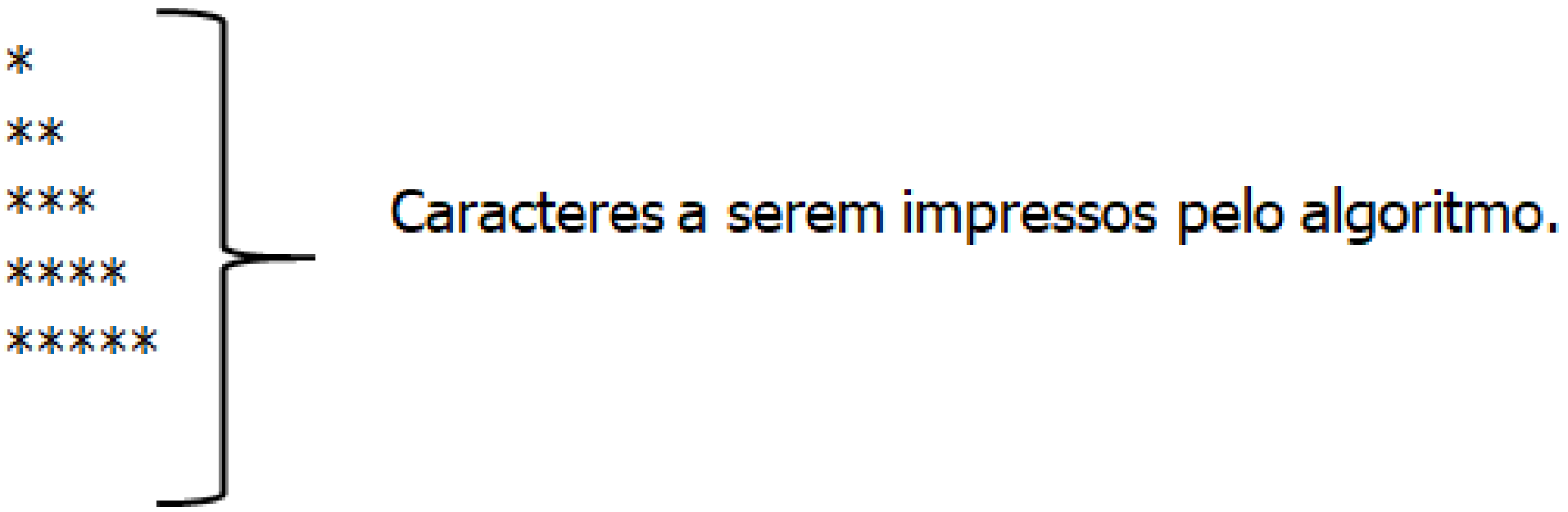
Nesta aula, encerramos as estruturas repetitivas. Você verá que, assim como as estruturas seletivas, elas são muito utilizadas nas soluções de programação; o que faz com que seja essencial conhecê-las bem e saber utilizá-las sem dificuldade.

Na próxima aula, veremos um tipo especial de variável que, para ser manipulado, contará com o apoio da estrutura repetitiva com variável de controle. Por isso, faça as atividades com atenção e reveja os tópicos apresentados caso tenha alguma dúvida, combinado?

Atividade

3. Crie um algoritmo que seja capaz de desenhar a metade de um triângulo com "*". A quantidade de linhas do triângulo será informada pelo usuário. Observe o exemplo de funcionamento a seguir:

Quantas linhas deve ter o triângulo? 5



a. Escreva o algoritmo anterior em C++.

b. Você sabe o que é o fatorial de um número? Trata-se do produto dos números inteiros entre 1 e este número cujo fatorial se deseja obter. Ciente disso, escreva o algoritmo que calcule e exiba o fatorial de um número fornecido pelo usuário.

c. A série de Fibonacci é uma sequência de números inteiros iniciada com 0 e 1 na qual os termos subsequentes correspondem à soma dos dois termos anteriores. Observe:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34...

Sabendo disso, escreva um programa que exiba a quantidade de termos da série que o usuário deseja ver. Por exemplo, se ele disse que quer ver os 4 primeiros termos da série, exiba "0, 1, 1, 2".

Notas
Referências

MANZANO, J. A. N. G., OLIVEIRA, J. F. **Algoritmos**: lógica para desenvolvimento de programação de computadores. 28.ed. São Paulo: Érica, 2016.

PUGA, S.; RISSETTI, G. **Lógica de programação e estruturas de dados com aplicações em Java.** 2.ed. São Paulo: Prentice Hall, 2005.

Próxima aula

- Vetores e como funcionam;
- Comportamento de algoritmos e programas que utilizam vetores;
- Códigos com vetores como recurso de armazenamento.

Explore mais

Você muito provavelmente já ouviu falar do Minecraft , o jogo no estilo Lego que insere os jogadores em mundos aleatórios nos quais eles conseguem criar suas próprias construções e engenhocas a partir de cubos texturizados. Que tal embarcar em uma aventura com um dos personagens desse jogo <https://studio.code.org/s/aquatic/stage/1/puzzle/1> ?