CI1055: Algoritmos e Estruturas de Dados I

Prof. Dr. Marcos Castilho

Departamento de Informática/UFPR

21 de outubro de 2021

Resumo

Repetição de comandos

Objetivos da aula

- Apresentar os conceitos elementares de linguagens de programação
  - o fluxo de execução de um programa
  - os comandos que manipulam dados e permitem interação com o usuário
  - as expressões aritméticas e lógicas
  - o comando de atribuição
  - (*) os comandos que permitem alteração do fluxo de execução do programa
- apresentação de tipos de erros que podem ocorrer

Comandos de repetição

- permitem repetir trechos de códigos
- existem três destes comandos em *Pascal*
- só veremos uma forma por enquanto
- a motivação será a partir de um problema bem simples

Problema: imprimir os números de 1 até 5

```
1  program imprimir_de_1_a_5;
2
3  begin
4      writeln (1);
5      writeln (2);
6      writeln (3);
7      writeln (4);
8      writeln (5);
9  end.
```

- solução muito simples!
- poderia ter uma linha só: `writeln ('1 2 3 4 5');`

Problema

- simples demais!
- não é generalizável para problemas similares
- por exemplo:

Problema: imprimir os números de 1 até 10

```
1  program imprimir_de_1_a_10;
2
3  begin
4      writeln (1);
5      writeln (2);
6      writeln (3);
7      writeln (4);
8      writeln (5);
9      writeln (6);
10     writeln (7);
11     writeln (8);
12     writeln (9);
13     writeln (10);
14 end.
```

- e se fosse imprimir os números de 1 até 100 milhões?

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

- a solução anterior é impossível de se adaptar para este caso!
- em tempo de edição do código, não conhecemos o valor de n
- é preciso encontrar solução melhor

Repetição de comandos

- repetir comandos é fundamental em computação
- o fluxo de execução dos comandos é alterado de maneira controlada
- aqui começa o estudo do conceito de *lógica de programação*

O comando *while*

```
1   while { expressao booleana } do
2       { algum comando };
```

- sintaxe versus semântica
  - acima está a sintaxe do comando *while*
  - a semântica é o *significado* do comando
- a expressão acima signica que este { algum comando } será repetido um certo número de vezes, ou nenhuma vez
- quem define se o comando será repetido, e até quando será repetido, é a { expressão booleana }
- *enquanto* esta { expressão booleana } for verdadeira o { algum comando } será executado

Exemplo

```
1  program exemplo_repeticao;
2  var i: longint;
3  begin
4      read (i);
5      while i < 0 do
6          read (i);
7  end.
```

1. inicia na linha 4 lendo um número do teclado
2. na linha 5, avalia a expressão `i < 0`
3. se a avaliar falso, "pula" para linha 7 e o programa termina
4. se a avaliação resultar verdadeiro:
   - executa o comando da linha 6, lendo outro número do teclado
   - volta para a linha 5 e reavalia a expressão `i < 0`

Exemplo

```
1  program exemplo_repeticao;
2  var i: longint;
3  begin
4      read (i);
5      while i < 0 do
6          read (i);
7  end.
```

O que este programa faz? Qual é o sifnificado dele?

Exemplo

```
1  program exemplo_repeticao;
2  var i: longint;
3  begin
4      read (i);
5      while i < 0 do
6          read (i);
7  end.
```

Enquanto o usuário estiver digitando números negativos, o obriga a digitar outro. Termina quando for digitado um número positivo ou nulo.

Exemplo 2

```
1  program exemplo_repeticao_2;
2  var i: longint;
3  begin
4      read (i);
5      while i < 0 do
6      begin
7          writeln ('numero negativo, digite outro');
8          read (i);
9      end;
10     writeln ('parabens! o numero ',i,' nao eh negativo');
11 end.
```

Para repetir mais de um comando é preciso colocá-los entre **begin** e **end;**

Exemplo de execução do programa anterior

```
marcos@tuco ~ $ ./exemplo_repeticao_2
-1
numero negativo, digite outro
-4
numero negativo, digite outro
9
parabens! o numero 9 nao eh negativo
marcos@tuco ~ $
```

- o comando da linha 7 foi executado duas vezes
  ele está no *escopo* do `while`
- o comando da linha 10 foi executado somente uma vez ele está *fora do escopo* do `while`

Outro exemplo de execução do programa anterior

```
marcos@tuco ~ $ ./exemplo_repeticao_2
9
parabens! o numero 9 nao eh negativo
marcos@tuco ~ $
```

- o comando da linha 7 não foi executado desta vez! o teste resultou falso logo de cara
- o comando da linha 10 foi executado, pois ele está *fora do escopo* do `while`

Repetição de comandos

- entender como comandos são repetidos não é suficiente para aprender a usar repetições ao resolver problemas
- é preciso entender o mecanismo de como isso pode ser utilizado para se resolver um problema particular
- por exemplo, no problema em estudo, como repetir código pode ajudar?

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

- muito bem, em que a repetição ajuda aqui?
- tipo: alguém me deu um prego e um martelo, o que eu faço agora?

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

- vamos observar:
  - para imprimir os número de 1 a n, imprime o 1
  - depois imprime o 2
  - depois imprime o 3
  - depois imprime o 4
  - ...
- é preciso identificar o **padrão repetitivo**
- e **como as informações se transformam de uma iteração para outra**

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

- observação:
  - cada número é o anterior **mais 1**
- isso ajuda?
  - sim, pois se eu tenho um número, na próxima vez basta somar um nele!
  - percebeu o **na próxima vez?**

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

- o **na próxima vez** indica a repetição
- então a solução pode ser construída baseado nisso
  - imprime um número
  - soma um nele
  - repete este processo

Dois problemas

- como terminar o processo?
- como iniciar o processo?

Como terminar o processo?

- lembrando do enunciado, imprimir os números de um até n
- como estamos somando um repetidas vezes, em algum momento o número vai ultrapassar n, certo?
- depende!

Como terminar o processo?

- Sim, depende. O primeiro número tem que ser menor ou igual a n, senão o processo nunca vai terminar!
- Isso nos leva ao outro problema

Como iniciar o processo?

- pelo primeiro número que queremos imprimir, certo?
- mais ou menos, depende de como você construiu seu programa

Primeiro rascunho da solução

```
1  begin
2      while i <= n do
3      begin
4          writeln (i);
5          i:= i + 1;    (* transforma um numero no proximo *)
6      end;
7  end.
```

- nos rascunhos não importa o cabeçalho do programa
- capturamos a ideia central: transformar um numero no seguinte e também como parar a repetição
- falta saber começar

Primeiro rascunho da solução

```
1  begin
2      while i <= n do
3      begin
4          writeln (i);
5          i:= i + 1;    (* transforma um numero no proximo *)
6      end;
7  end.
```

- quando entra no laço, a primeira coisa que ocorre é a impressão do número da vez
- então, **na primeira vez**, qual número queremos imprimir?
- é o 1, certo?
- então tem que começar o `i` com 1 **antes** do laço

Problema: ler um número inteiro positivo n do teclado e imprimir todos os números inteiros entre 1 e n

```
1  program imprimir_de_1_a_n;
2  var i, n: longint;
3
4  begin
5      read (n);
6      i:= 1;
7      while i <= n do
8      begin
9          writeln (i);
10         i:= i + 1;
11     end;
12 end.
```

Outra solução para o mesmo problema

```
1  program imprimir_de_1_a_n_v2;
2  var i, n: longint;
3
4  begin
5      read (n);
6      i:= 0;
7      while i < n do
8      begin
9          i:= i + 1;
10         writeln (i);
11     end;
12 end.
```

Duas soluções?

- qual é a diferença entre estas soluções?
- existem outras ou são somente estas duas?

Outra solução para o mesmo problema

```
1  program imprimir_de_1_a_n_v3;
2  var i, n: longint;
3
4  begin
5      read (n);
6      i:= n;
7      while i > 0 do
8      begin
9          writeln (n - i + 1);
10         i:= i - 1;
11     end;
12 end.
```

Você consegue pensar em outras maneiras?

Exercícios

- fazer os exercícios contidos na seção 5.10.5 do livro [1]

[1] `http://www.inf.ufpr.br/cursos/ci055/livro_alg1.pdf`

Fim do tópico

- o conteúdo desta aula está no livro no capítulo 5, seção 5.7.1
- na próxima aula veremos critérios de parada de uma repetição

Licença

- Slides feitos em LaTeX usando beamer
- Licença

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`