Fundamentos de Programação
Estruturas de Repetição

Dainf - UTFPR

Profa. Leyza B. Dorini
Prof. Bogdan T. Nassu

Estruturas de Repetição

Até agora, vimos como escrever programas capazes de executar comandos de forma linear, e, se necessário, tomar decisões com relação a executar ou não um bloco de comandos. Na sequência, iremos abordar estruturas que permitem executar um bloco de comandos repetidas vezes.

Introdução

Suponha que você precise fazer um programa que escreve na tela os números inteiros de 1 a 4. Com o que já aprendemos até agora, a solução seria:

```
1  int main(){
2
3      printf("1 ");
4      printf("2 ");
5      printf("3 ");
6      printf("4.");
7
8      return 0;
9  }
```

Saída:

```
1  1 2 3 4.
2  Process returned 0 (0x0)   execution time:0.579s
3  Press any key to continue.
```

Introdução - problema 01

Agora, suponha que é preciso escrever na tela os números inteiros de 1 a 100!

```
1 printf("1 ");
2 printf("2 ");
3 printf("3 ");
4 printf("4 ");
5 //repete a linha acima para todos os valores entre 5
      e 99
6 printf("100.");
7
```

Além de trabalhosa, essa solução está sujeita a erros, tais como esquecer ou errar um dígito!

Introdução - problema 02

Agora, suponha que é preciso ler do teclado valores inteiros até que o usuário digite um número negativo. Por exemplo, se o usuário digitar o valor 10, deve ser lido outro número. Entretanto, ao digitar -1, o programa deve parar de ler valores.

```
1 scanf("%d", &a);
2 if(a>0)
3     scanf("%d", &a);
4 if(a>0)
5     scanf("%d", &a);
6
7 // e agora? Como saber quantos if's colocar?
```

Para fazer isso de forma mais robusta, vamos aprender **estruturas de repetição**!

Estruturas de repetição

Vamos dividir em diferentes casos de estudo. Eles estão relacionados à quantidade de vezes que um bloco de comandos deve ser executado:

1. quando tal quantidade é conhecida **antes** das repetições iniciarem, ou seja, é pré-determinada;
2. quando ela depende de algo que só será conhecido **dentro do bloco de comandos** da estrutura de repetição (condição, cálculo, conteúdo de variável, ...).

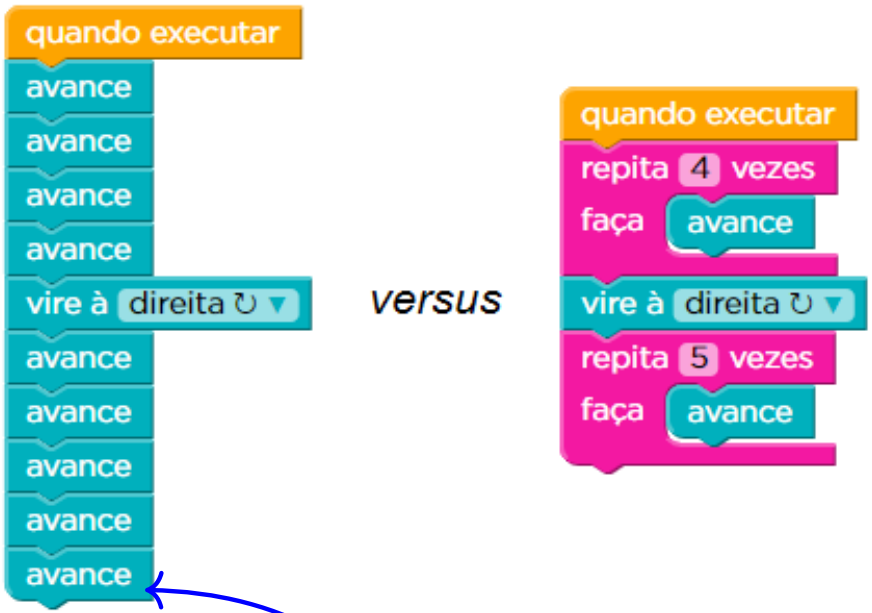
Além disso, veremos também **repetições aninhadas**!

Este tipo de estrutura que repete uma sequência de comandos é chamada de *laço* ou, em inglês, *loop*.

Caso 01: *loops* determinados

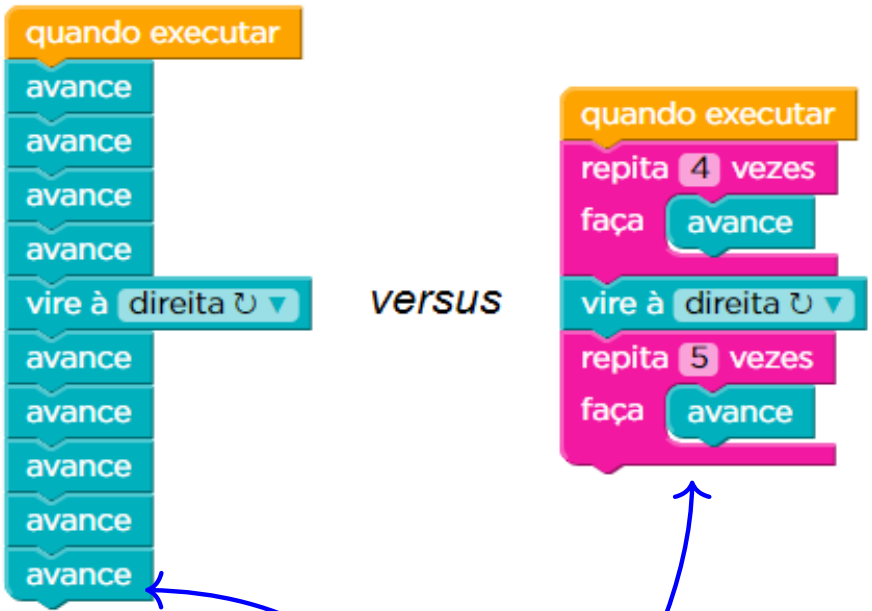
Quando a quantidade de repetições é pré-determinada, ou seja, é conhecida antes o início do *loop*!

Exemplo do Code.org: fazer o zumbi chegar ao girassol



- Solução possível com o que vimos até agora!

Exemplo do Code.org: fazer o zumbi chegar ao girassol



- Solução possível com o que vimos até agora!
- Solução utilizando estruturas de repetição. Observe que é muito mais fácil realizar alterações do programa (seja para mudar o comando "`avance`" ou para alterar a quantidade de repetições, por exemplo).

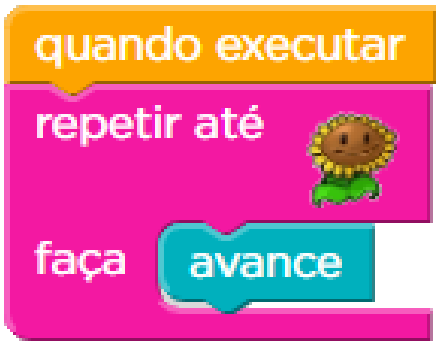
Caso 02: *loops* indeterminados

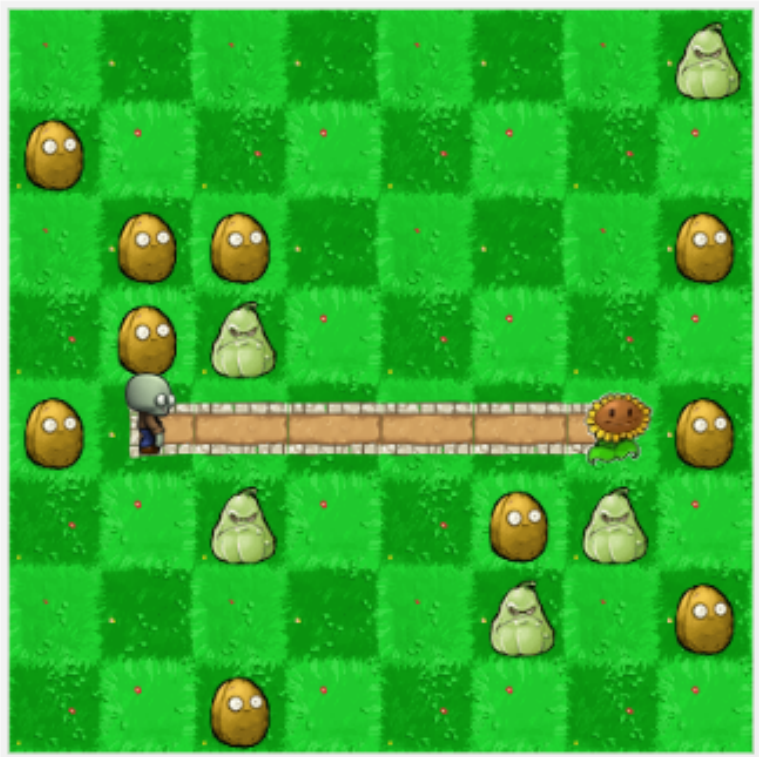
Quando a quantidade de repetições depende do resultado de algo determinado dentro do *loop* (valor de variável, condição, ...).

Exemplo do `Code.org`

Dependendo do contexto, uma outra possibilidade é controlar o *loop* por uma condição tal como "`Enquanto não chegar no girassol`".

Solução com o `Code.org`:



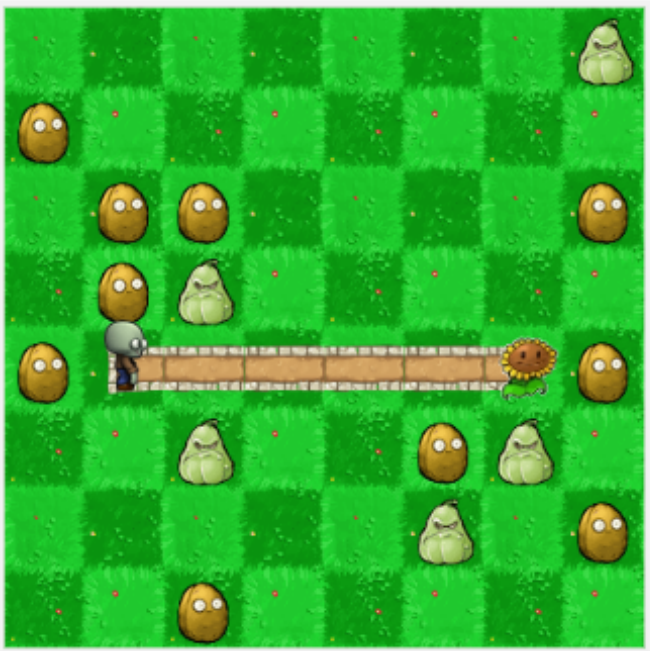


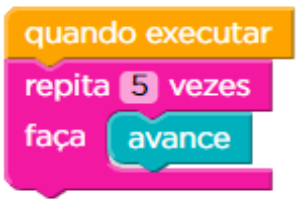
Note que a definição da execução (ou não) do comando "`Avance`" depende da posição atual do zumbi!

loops determinados $\times$ indeterminados

loops determinados *versus* indeterminados

Quando usar cada um deles?

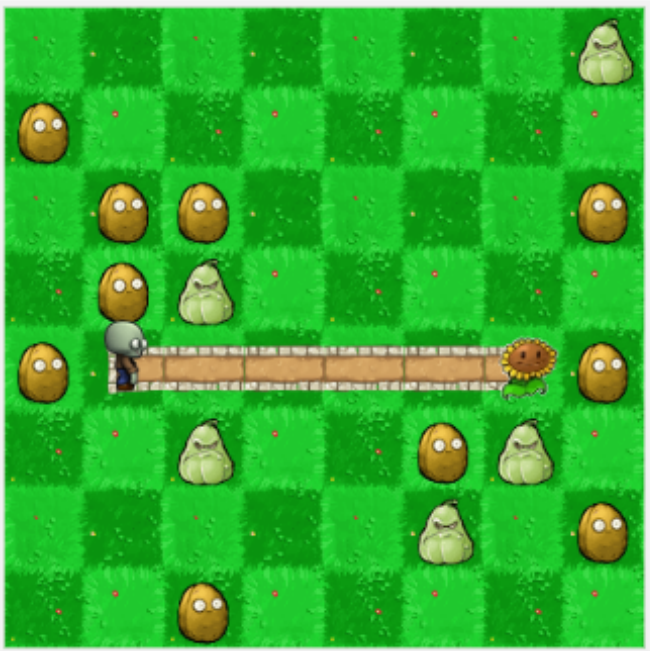


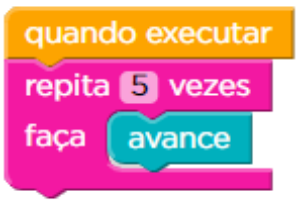


- Determinado: quando sabemos *a priori* (antes de iniciar o *loop*) a quantidade de repetições que devem ser realizadas.

loops determinados *versus* indeterminados

Quando usar cada um deles?





- Determinado: quando sabemos *a priori* (antes de iniciar o *loop*) a quantidade de repetições que devem ser realizadas.
- Indeterminado: quando a quantidade de repetições depende de valores que só serão conhecidos (lidos ou calculados) no próprio *loop*.

loops determinados *versus* indeterminados

Como saber qual deles usar para uma determinado problema?

Leia o problema e procure identificar se você sabe ou não a quantidade de iterações[1] de um determinado bloco de comandos! E pratique muito! Nos próximos materiais iremos ver isso de forma mais detalhada!

[1] Cada repetição é uma *iteração*. Não confunda com ¡Nteração!!!

loops determinados *versus* indeterminados

No caso do exemplo do zumbi:

1. Se soubermos que ele está a 5 passos do girassol, podemos usar um *loop* **determinado**.
2. Se, para chegar no girassol, for necessário observar a cada passo do zumbi se ele ainda está na estrada ou se já chegou no girassol, será necessário um *loop* **indeterminado.**

Caso 03: estruturas de repetição aninhadas

Quando o bloco de comandos de uma estrutura de repetição tem outra estrutura de repetição!

Estruturas de repetição aninhadas

Esse caso nós vamos discutir depois...