

BCM0505-22 – Processamento da Informação

Laços - Parte 2

Carla Negri Lintzmayer
carla.negri@ufabc.edu.br
<http://professor.ufabc.edu.br/~carla.negri/>

Outline

Animações

Estrutura de Repetição: For

Exemplos

Exercícios

Animações

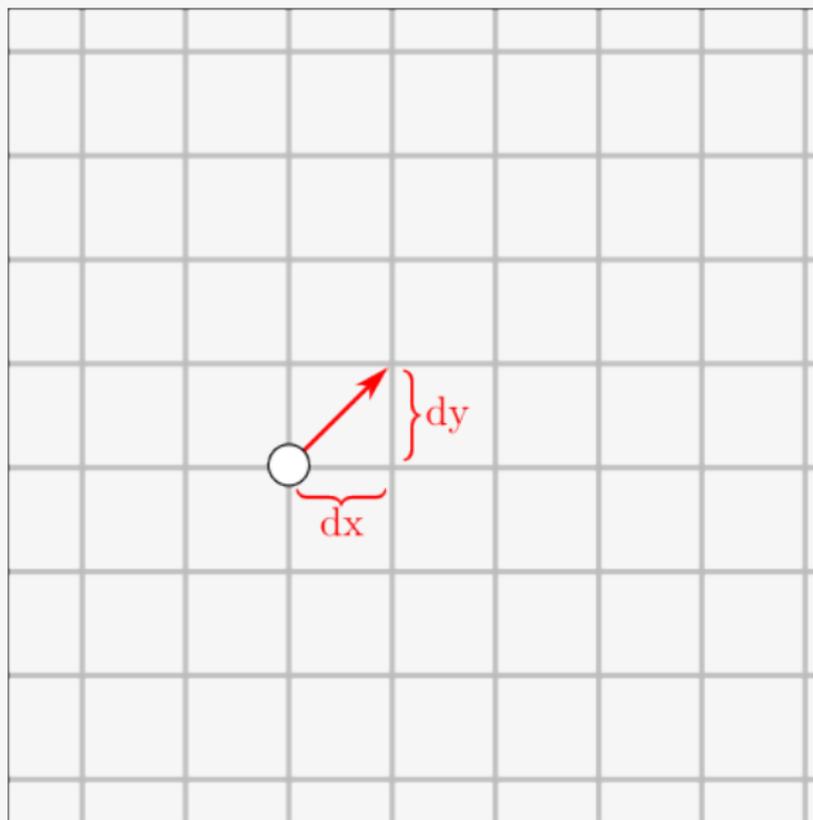
Animações

- Agora que aprendemos a fazer laços em Python, podemos desenhar figuras mais complexas
 - E, até mesmo, fazer animações!

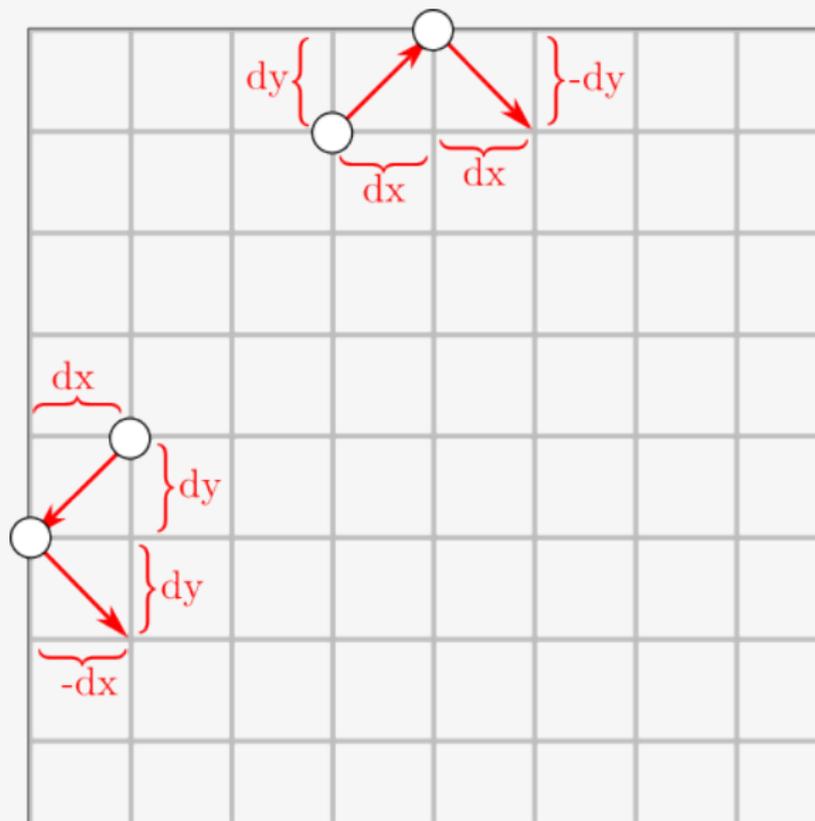
Animações

- Agora que aprendemos a fazer laços em Python, podemos desenhar figuras mais complexas
 - E, até mesmo, fazer animações!
- Vamos fazer uma bola saltitante!
 - Exibir a animação

Bola Saltitante: Ideia



Bola Saltitante: Ideia



Bola Saltitante: Código

```
1 import sys
2 from intropy import stddraw
3
4 DT = 20 #tempo do desenho
5 x = 0.5 #abscissa da posição da bola
6 y = 0.5 #ordenada da posição da bola
7 dx = 0.015 #vetor deslocamento no eixo-x
8 dy = 0.023 #vetor deslocamento no eixo-y
9 RADIUS = 0.05 #raio da bola
10
11 stddraw.setXscale(-1, 1)
12 stddraw.setYscale(-1, 1)
13
14 while True:
15     # a todo momento, incrementar a posição da bola pelos deslocamentos desejados,
16     # limpar a imagem e desenhar o círculo na nova posição
17     # verificar caso ultrapasse o tamanho da tela
18     if abs(x + dx) + RADIUS > 1:
19         dx = -dx
20
21     if abs(y + dy) + RADIUS > 1:
22         dy = -dy
23
24     x += dx
25     y += dy
26
27     stddraw.clear()
28     stddraw.circle(x, y, RADIUS)
29     stddraw.show(DT)
```

Estrutura de Repetição: For

For (Sintaxe)

```
1 for var in S:  
2     instrução_1  
3     instrução_2  
4     ...  
5     instrução_k  
6 instrução_A
```

- O corpo do laço é definido pela indentação
- **instrução_A** é a primeira instrução fora do laço
 - é a primeira instrução cujo nível de indentação é igual ao do **for**
- **S** é uma “sequência de elementos”
- **var** é o nome de uma variável que será criada e cujo valor será atualizado automaticamente

For: Funcionamento

```
1 for var in S:  
2     instrução_1  
3     instrução_2  
4     ...  
5     instrução_k  
6 instrução_A
```

Funcionamento

- Executa o corpo do laço $|S|$ vezes
- Na i -ésima execução, **var** assume o valor do i -ésimo elemento de **S**

For: Funcionamento

```
1 for var in S:  
2     instrução_1  
3     instrução_2  
4     ...  
5     instrução_k  
6 instrução_A
```

Funcionamento

- Executa o corpo do laço $|S|$ vezes
- Na i -ésima execução, **var** assume o valor do i -ésimo elemento de **S**

Exemplo

$S = [1, 3, 4, 9]$

Iteração	var
1	1
2	3
3	4
4	9

Sequência de Elementos

Podemos criar uma sequência de elementos listando seus membros entre colchetes ([]), separados por vírgula ,

Exemplo

```
1 for fruta in ["uva", "limão", "melão", "abacaxi"]:  
2     print(f"hoje eu comi um(a) {fruta}")
```

Função range

Função **range**:

- muito utilizada em conjunto com a instrução **for**
- cria uma sequência de números em um dado intervalo

Função range

Função `range`:

- muito utilizada em conjunto com a instrução `for`
- cria uma sequência de números em um dado intervalo

Chamada	Retorno
<code>range(n)</code>	<code>[0, 1, 2, ..., n - 1]</code>
<code>range(ini, fim)</code>	<code>[ini, ini+1, ini+2, ..., fim - 1]</code>
<code>range(ini, fim, inc)</code>	<code>[ini, ini+inc, ini+2*inc, ini+3*inc, ..., x]</code>

onde `x` é o maior número da forma `ini + i * inc` que estritamente menor que `fim`

Função range

Função `range`:

- muito utilizada em conjunto com a instrução `for`
- cria uma sequência de números em um dado intervalo

Chamada	Retorno
<code>range(n)</code>	<code>[0, 1, 2, ..., n - 1]</code>
<code>range(ini, fim)</code>	<code>[ini, ini+1, ini+2, ..., fim - 1]</code>
<code>range(ini, fim, inc)</code>	<code>[ini, ini+inc, ini+2*inc, ini+3*inc, ..., x]</code>

onde `x` é o maior número da forma `ini + i * inc` que estritamente menor que `fim`

Exemplos:

- `range(2, 10, 3) = [2, 5, 8]`
- `range(2, 9, 3) = [2, 5, 8]`

OBS: note que o intervalo em `range` é fechado à esquerda e aberto à direita!

Exemplos

Exemplo de for com range

Dado um n , imprima os números do intervalo $[0, 1, \dots, n - 1]$

```
1 n = int(input())
2 i = 0
3 while i < n:
4     print(i)
5     i += 1
```

Exemplo de for com range

Dado um n , imprima os números do intervalo $[0, 1, \dots, n - 1]$

```
1 n = int(input())
2 i = 0
3 while i < n:
4     print(i)
5     i += 1
```

```
1 n = int(input())
2 for i in range(n):
3     print(i)
```

Soma de números ímpares

Dado um n , imprima a soma dos n primeiros números ímpares

```
1  n = int(input())
2  cont = 1
3  i = 1
4  soma = 0
5  while cont <= n:
6      if i % 2 == 1:
7          soma += i
8          cont += 1
9      i += 1
10 print(f"Soma: {soma}")
```

Soma de números ímpares

Dado um n , imprima a soma dos n primeiros números ímpares

```
1 n = int(input())
2 cont = 1
3 i = 1
4 soma = 0
5 while cont <= n:
6     if i % 2 == 1:
7         soma += i
8         cont += 1
9     i += 1
10 print(f"Soma: {soma}")
```

```
1 n = int(input())
2 soma = 0
3 for i in range(2*n):
4     if i % 2 == 1:
5         soma += i
6 print(f"Soma: {soma}")
```

Soma de números ímpares

Dado um n , imprima a soma dos n primeiros números ímpares

```
1  n = int(input())
2  cont = 1
3  i = 1
4  soma = 0
5  while cont <= n:
6      if i % 2 == 1:
7          soma += i
8          cont += 1
9      i += 1
10 print(f"Soma: {soma}")
```

```
1  n = int(input())
2  soma = 0
3  for i in range(2*n):
4      if i % 2 == 1:
5          soma += i
6  print(f"Soma: {soma}")
```

Pergunta: Quantas vezes os laços executam? Podemos melhorar?

Soma de números ímpares

Dado um n , imprime a soma dos n primeiros números ímpares

```
1 n = int(input())
2 soma = 0
3 for i in range(n):
4     soma += 2*i + 1
5 print(f"Soma: {soma}")
```

Soma de números ímpares

Dado um n , imprime a soma dos n primeiros números ímpares

```
1 n = int(input())
2 soma = 0
3 for i in range(n):
4     soma += 2*i + 1
5 print(f"Soma: {soma}")
```

```
1 n = int(input())
2 i = 1
3 soma = 0
4 while i <= n:
5     soma += 2*i + 1
6     i += 1
7 print(f"Soma: {soma}")
```

Soma de números ímpares

Dado um n , imprime a soma dos n primeiros números ímpares

```
1 n = int(input())
2 soma = 0
3 for i in range(n):
4     soma += 2*i + 1
5 print(f"Soma: {soma}")
```

```
1 n = int(input())
2 i = 1
3 soma = 0
4 while i <= n:
5     soma += 2*i + 1
6     i += 1
7 print(f"Soma: {soma}")
```

Pergunta: Quantas vezes os laços executam? Podemos melhorar?

Soma de números ímpares

Dado um n , imprime a soma dos n primeiros números ímpares

```
1 n = int(input())
2 soma = 0
3 for i in range(1, 2*n+1, 2):
4     soma += i
5 print(f"Soma: {soma}")
```

Soma de números ímpares

Programador

```
1 n = int(input())
2 soma = 0
3 for i in range(n):
4     soma += 2*i + 1
5 print(f"Soma: {soma}")
```

Soma de números ímpares

Programador

```
1 n = int(input())
2 soma = 0
3 for i in range(n):
4     soma += 2*i + 1
5 print(f"Soma: {soma}")
```

Bacharel em Ciência da Computação

```
1 n = int(input())
2 soma = n ** 2
3 print(f"Soma: {soma}")
```

Exercícios

Exercícios

IMPORTANTE todos os exercícios devem ser feitos usando `for`, e não `while`!

Ajudando o Bart

O Diretor Skinner, cansado das travessuras de Bart, decidiu que era hora de “modernizar” os castigos. Em vez de fazer o Bart escrever a mesma frase várias vezes em um quadro ultrapassado, ele agora teria que digitar a frase diversas vezes em um computador (não tão ultrapassado assim).

Bart, sendo Bart, logo pensou em uma nova travessura: contratar alguém para escrever um programa que fizesse todo o trabalho por ele. E é aí que você entra!

Bart lhe pediu para criar um programa que, dado um número n e uma frase s , escreva a frase s n vezes.



Números pares

Dado um inteiro n , escreva um programa que imprima todos os números pares do intervalo $[0, \dots, n]$.

Contagem Regressiva

Dado um inteiro n , escreva um programa que imprima uma contagem regressiva começando de n .

Por exemplo, se $n = 5$, então o seu programa deveria imprimir algo como a seguinte saída:

```
5
4
3
2
1
ACABOU!!!
```

FizzBuzz

Escreva um programa que imprima a sequência *FizzBuzz*!

A sequência *FizzBuzz* de n consiste de todos os números do intervalo $[1, 2, \dots, n]$ com exceção dos números divisíveis por 3 ou 5. Esses números divisíveis são trocados pelas palavras *Fizz*, *Buzz* e *FizzBuzz* de acordo com a seguinte regra:

- Se o número é divisível por 3 e 5, troque o número pela palavra “FizzBuzz”
- Se o número é divisível por 3, mas não por 5, troque o número pela palavra “Fizz”
- Se o número não é divisível por 3, mas é por 5, troque o número pela palavra “Buzz”

Assim, a sequência para $n = 15$ é

```
1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz
```

Plotando uma função

Escreva um programa que plota a função $f(x) = \frac{x^3 + 3x^2 - 6x - 8}{4}$ no intervalo $(-6, 5)$.

Além de desenhar a função, o seu programa deve:

- Desenhar o eixo x e y
- Marcar os eixos com um traço
- Desenhar a malha do plano cartesiano

Obs.: Nesse exercício, é necessário usar `while` em um momento.



Série de Taylor

Podemos usar a Série de Taylor para aproximar funções.

A seguir podemos ver como aproximar a função e^x pela Série de Taylor:

$$e^x = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \dots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

Escreva uma função em Python chamada `exp(n)` que compute e^x usando a expansão da Série de Taylor.

Faça experimentações para definir o número de termos que você usará na sua função (já que infinitos termos será impossível):

- Muitos termos tornam a função precisa, mas, potencialmente, lenta
- Poucos termos tornam a função rápida, porém, potencialmente, imprecisa