Estrutura de repetição `for` e função `range`

Aula 8

Diego Padilha Rubert

Faculdade de Computação
Universidade Federal de Mato Grosso do Sul

Algoritmos e Programação

Conteúdo da aula

1. Introdução
2. Estrutura de repetição `for`
3. Função `range`
4. Exercícios

Introdução

- Outra estrutura de repetição na linguagem Python: `for`, equivalente à estrutura de repetição `while`
- Uma função muito útil para a estrutura de repetição `for`: `range`

Introdução

- Outra estrutura de repetição na linguagem Python: `for`, equivalente à estrutura de repetição `while`
- Uma função muito útil para a estrutura de repetição `for`: `range`

Estrutura de repetição `for`

- A estrutura de repetição **`for`** serve para iterar sobre algum conjunto de dados
- Até agora, o único conjunto de dados que conseguimos iterar é as letras de uma cadeia de caracteres (string)

Estrutura de repetição `for`

- A estrutura de repetição **`for`** serve para iterar sobre algum conjunto de dados
- Até agora, o único conjunto de dados que conseguimos iterar é as letras de uma cadeia de caracteres (string)

Estrutura de repetição `for`

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# Conta quantos espaços a string lida tem

texto = input()
cont = 0

for letra in texto:
   if letra == " ":
      cont += 1

print("O texto digitado tem %d espaços" % cont)

exit(0)
```

Função `range`

- A função **`range(x,y)`** define os **inteiros** em um intervalo $[x, y[$ ou $[x, y)$
- Por exemplo, **`range(2,8)`** devolve os inteiros no intervalo $[2, 8[$
- Se o primeiro valor é omitido, assumimos que ele é zero
- Portanto, **`range(10)`** devolve os inteiros no intervalo $[0, 9[$

```
# Mostra de 2 formas diferentes os 100 primeiros inteiros positivos

for numero in range(100):
    print("%d" % (numero+1))

for numero in range(1,101):
    print("%d" % (numero))
```

Exercícios

1. Qualquer número natural de quatro algarismos pode ser dividido em duas dezenas formadas pelos seus dois primeiros e dois últimos dígitos.
   Exemplos:
   1297: 12 e 97; 5314: 53 e 14.
   Verifique se o programa a seguir imprime todos os números naturais de quatro algarismos cuja raiz quadrada é a soma das dezenas formadas pela divisão acima. Por exemplo, 9801 é um dos números a ser impresso, já que $\sqrt{9801} = 99 = 98 + 01$. Faça pelo menos uma simulação da execução passo a passo do programa.

Exercícios

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# Imprime os números inteiros positivos de 4 dígitos
# cuja raiz quadrada é igual à soma dos seus dois
# primeiros e dois últimos dígitos */

for numero in range(1000, 10000):
   DD = numero / 100
   dd = numero % 100
   if (DD + dd) * (DD + dd) == numero:
      print("%d" % (numero))

exit(0)
```

Exercícios

2. Verifique se o programa a seguir soluciona o seguinte problema: dado um número inteiro não-negativo n, escreva um programa que determine quantos dígitos o número n possui. Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Recebe um número inteiro não-negativo e im-
# prime a quantidade de dígitos que possui

n = int(input("Informe n: "))

digitos = 0
if n == 0:
   digitos = 1

while n > 0:
   n /= 10
   digitos += 1
print("O número tem %d dígitos" % (digitos))

exit(0)
```

Exercícios

3. Dado um número natural na base binária, transformá-lo para a base decimal.
   Exemplo:
   Dado 10010 a saída será 18, pois
   $1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 18$.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.
4. Dado um número natural na base decimal, transformá-lo para a base binária.
   Exemplo:
   Dado 18 a saída deverá ser 10010.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

3. Dado um número natural na base binária, transformá-lo para a base decimal.
   Exemplo:
   Dado 10010 a saída será 18, pois
   $1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 18$.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.
4. Dado um número natural na base decimal, transformá-lo para a base binária.
   Exemplo:
   Dado 18 a saída deverá ser 10010.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

5. Dado um número inteiro positivo n, transformá-lo e imprimi-lo na ordem inversa de seus dígitos.
   Exemplo:
   Dado 26578 a saída deverá ser 87562.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

6. Dizemos que um número natural n é **palíndromo** se
   - o primeiro algarismo de n é igual ao seu último algarismo;
   - o segundo algarismo de n é igual ao se penúltimo algarismo
   - e assim sucessivamente.

   Exemplos:

   567765 é palíndromo
   32423 é palíndromo
   567675 não é palíndromo.

   Dado um número natural n, verificar se n é palíndromo. Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

7. Dados um número inteiro positivo n e uma sequência de n números inteiros, determinar quantos segmentos de números iguais consecutivos compõem essa sequência.
   Exemplo:

   Para $n = 9$, a sequência $\overbrace{5}, \overbrace{-2, -2}, \overbrace{4, 4, 4, 4}, \overbrace{1, 1}$ é formada por 4 segmentos de números iguais.
   Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

8. Uma sequência de números $a_1, a_2, \ldots, a_n$, com $n \geqslant 1$, é chamada uma **sequência crescente** se para cada par a_i, a_{i+1} de números consecutivos da sequência, com $1 \leqslant 1 < n$, vale que $a_i \leqslant a_{i+1}$.
Dados um número inteiro positivo n e uma sequência de n números inteiros, determinar o comprimento de um segmento crescente de comprimento máximo.
Exemplos:

   Na sequência $5, 10, 6, \overbrace{2, 4, 7, 9}, 8, -3$ o comprimento do segmento crescente máximo é 4.

   Na sequência $10, 8, 7, 5, 2$ o comprimento do segmento crescente máximo é 1.

Faça pelo menos uma simulação da execução passo a passo da sua solução.

Exercícios

9. Observe o algoritmo abaixo, que determina quantos espaços uma frase contém.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

frase = input("Informe uma frase: ")
esp = 0
for c in frase:
    if c == " ":
        esp += 1
print("A frase tem %d espaços" % (esp))

exit(0)
```

Utilize a estrutura de repetição **for** para, dada uma cadeia de caracteres, contar a quantidade de letras minúsculas, letras maiúsculas, dígitos, espaços e símbolos de pontuação que essa cadeia possui.

Exercícios

9. Observe o algoritmo abaixo, que determina quantos espaços uma frase contém.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

frase = input("Informe uma frase: ")
esp = 0
for c in frase:
   if c == " ":
      esp += 1
print("A frase tem %d espaços" % (esp))

exit(0)
```

Utilize a estrutura de repetição **for** para, dada uma cadeia de caracteres, contar a quantidade de letras minúsculas, letras maiúsculas, dígitos, espaços e símbolos de pontuação que essa cadeia possui.

Exercícios

9. Observe o algoritmo abaixo, que determina quantos espaços uma frase contém.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

frase = input("Informe uma frase: ")
esp = 0
for c in frase:
   if c == " ":
      esp += 1
print("A frase tem %d espaços" % (esp))

exit(0)
```

Utilize a estrutura de repetição **for** para, dada uma cadeia de caracteres, contar a quantidade de letras minúsculas, letras maiúsculas, dígitos, espaços e símbolos de pontuação que essa cadeia possui.