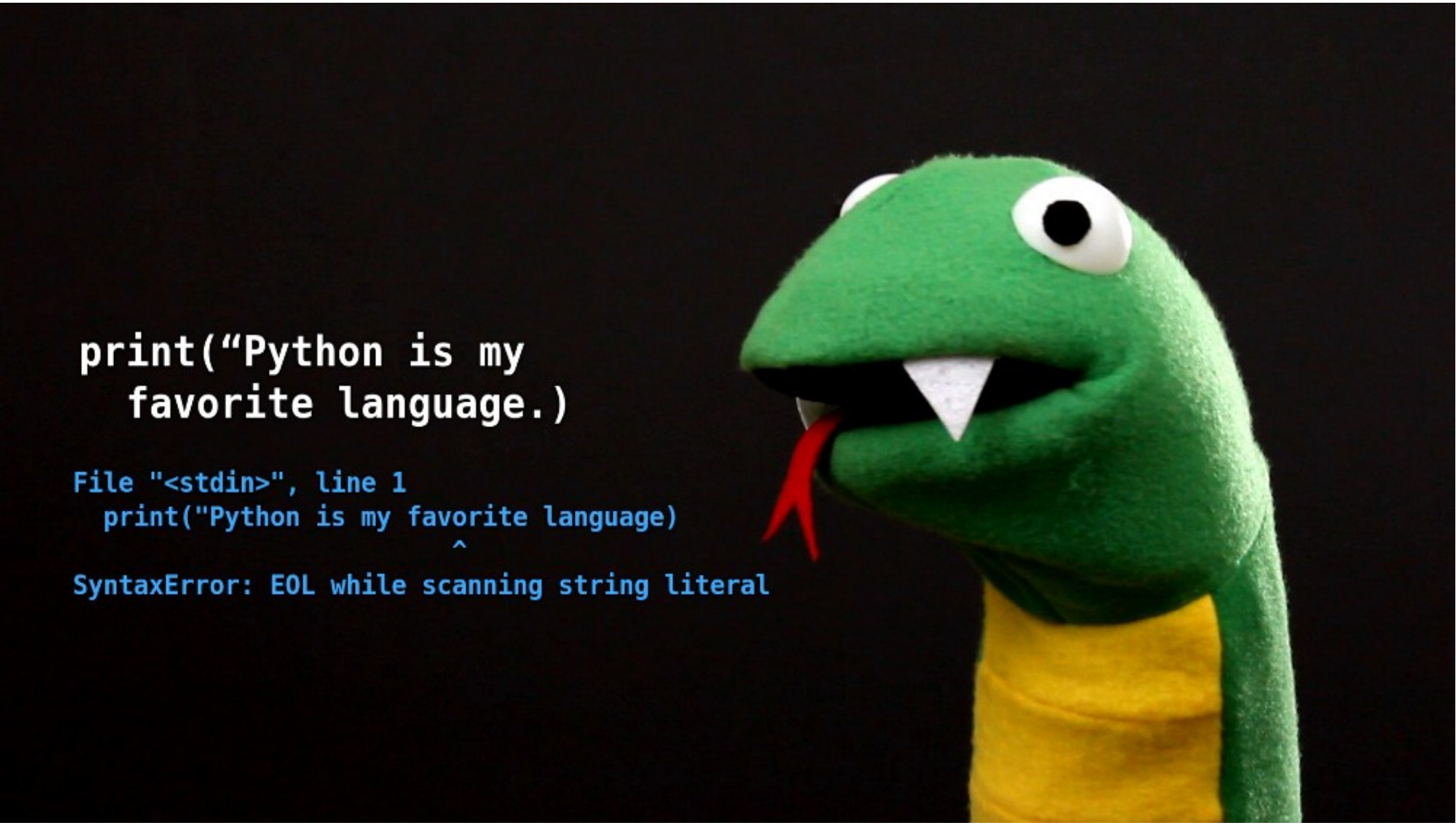
Vetores e Strings



Vetores
(Motivação)

Dada uma sequência de 3 números, imprimi-la na ordem inversa à da leitura.

Simples!

```
numero1 = input('Informe o número 1: ')
numero2 = input('Informe o número 2: ')
numero3 = input('Informe o número 3: ')
print(numero3)
print(numero2)
print(numero1)
```

Vetores
(Motivação)

Mas e se quisermos generalizar?

Problema 1:

Dada uma sequência de ***n*** números, imprimi-la na ordem inversa à da leitura.

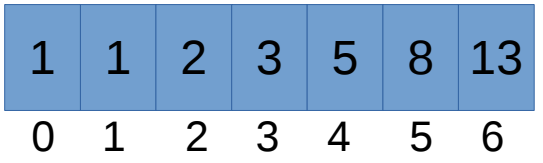
Vetores

Vetores são variáveis compostas, ou seja, variáveis que guardam mais de um valor.

Em python vetores são chamados também de listas.

Ex:

x = [1,1,2,3,5,8,13] →



print (x[0], x[1], x[6]) → 1 1 13

Listas

Listas em python são naturalmente dinâmicas:

x = [1,1,2]

x.append(3) → x vai ser igual à [1,1,2,3]

Listas em python podem possuir tipos distintos de dados:

x = [1, 3.14, ‘p’]

Como usar listas?

Para criar uma lista vazia:

x = []

Para acessar a posição i:

x[i]

Para acrescentar um item y:

x.append(y)

Para remover o item da posição i:

x.pop(i)

Para saber o tamanho da lista:

len(x)

Pertinência:

y in x – Retorna True se y for um item de x ou False caso contrário

x.index(y) – Retorna o índice da primeira ocorrência de y ou erro

Exemplo

```
>>> x = []
>>> x.append('a')
>>> x.append('b')
>>> len(x)
2
>>> print(x)
['a', 'b']
>>> 'b' in x
True
>>> x.pop(1)
'b'
>>> print(x)
['a']
```

Atacando o Problema 1

Dada uma sequência de ***n*** números, imprimi-la na ordem inversa à da leitura.

Resposta

```
def main():
  n = eval(input("Informe o tamanho da sequência:"))
  Lista = []
  i=0
  while i < n:
    x = input("Informe um número:")
    lista.append(x)
    i += 1
  i = n-1
  while i >= 0:
    print(lista[i])
    i += 1
```

Mais sobre listas

Listas podem ser indexadas de trás para frente usando índices negativos:

```
x = [101,102,103]
print(x[-1],x[-2],x[-3])   → 103 102 101
```

Podemos concatenar listas com o operador +:

```
x = [1,2,3]
y = x + [4,5,6]            → y = [1,2,3,4,5,6]
```

Podemos repetir uma lista usando o operador *:

```
x = [1,2,3]
x = x*3                    →  x = [1, 2, 3, 1, 2, 3, 1, 2, 3]
```

Exercício 1

Dados dois inteiros **n** e **m** e **duas sequências**, uma com n itens e a outra com m itens,imprimir os itens que estão presentes em ambas as sequências ao menos uma vez.

Resposta Ex. 1

```
def main():
  n = eval(input("Informe o tamanho da sequência 1 (n):"))
  Lista1 = []
  i = 0
  while i < n :
    x = input("Informe um número:")
    lista1.append(x)
    i += 1
  m = eval(input("Informe o tamanho da sequência 2 (m):"))
  Lista2 = []
  i  = 0
  while i < m :
    x = input("Informe um número:")
    lista2.append(x)
    i += 1
  repetidos = []
  i = 0
  while i < n :
    if (lista1[i] in lista2 and  (lista1[i] in repetidos == false)):
       repetidos.append(lista1[i])
    i += 1
  print(repetidos)
```

Exercício 2

Dados um número inteiro **n** e uma **sequência** de n números reais, imprimir os números distintos que compõem a sequência e o número de vezes que cada um deles ocorre na mesma

Resposta Ex. 2

```
def main():
  n = eval(input("Informe o tamanho da sequência 1 (n):"))
  unicos = []
  quantidades = []
  i = 0
  while i < n :
    x = input("Informe um número:")
    if (x not in unicos):
      unicos.append(x)
      quantidades.append(1)
    else:
      i = unicos.index(x)
      quantidades[i] += 1
    i += 1
 i = 0
 while i < len(unicos) :
    print(unicos[i]," aparece ",quantidades[i], " vezes.")
    i += 1
```

Strings

Tipo de variável que representa informação textual.

Strings são um tipo especial de vetor: vetores de caracteres.

```
>>>texto = “Este é um exemplo de string”

>>> texto = '''

... Assim podemos escrever

... um texto bem grande,

... inclusive com quebras de linha. '''

>>> texto

'\nAssim podemos escrever\num texto bem grande,\ninclusive com quebras de linha. '
```

Strings

Podem ser acessadas como vetores (listas):

```
texto = “Este é um exemplo de string”
print(texto[0],texto[1],texto[2],texto[3])
```

Mas não podem ser modificadas:

```
texto[3] = ‘a’
```

TypeError: 'str' object does not support item assignment

É possível usar os operadores +, *, in e a len()

Funções para Strings

Buscando caracteres

```
string = "Adoro PYTHON"
>>> string.count("o")
2
>>> string.find(" ")
5
>>> string.index("PY")
6
```

Funções para Strings

Alterando maiúsculas e minúsculas

```
>>> string = “aDoRo pYTHon”
>>> string.upper()
ADORO PYTHON
>>> string.lower()
adoro python
>>> string.title()
Adoro Python
>>> string.capitalize()
Adoro python
```

Exercício 3

Dada uma linha de texto de entrada, imprima a quantidade de palavras e a quantidade de caracteres.

Resposta Ex. 3

```
def main():
  texto = input("Digite um texto:")
  palavras = 0
  caracteres = len(texto)
  separadores = [" ","\t",",",".",";",":","?","!"]
  i = 1
  while i < len(texto):
    if (texto[i] in separadores and (texto[i-1] in separadores == false) or (i == (len(texto)-1) and (texto[i] in separadores == false))):
      palavras += 1
    i += 1
  print("O texto possui ",palavras," palavras e ",caracteres, " caracteres.")
```

Exercício 4

Dada uma linha de texto de entrada, decida se é um palíndromo ou não.

Exemplo de palíndromos:

ovo, aba, arara, osso.

Resposta Ex. 4

```
def main():
  palindromo = True
  texto = input("Digite uma palavra:")
  i = 0
  while i < len(texto):
    if texto[i] != texto[len(texto)-i-1]:
      palindromo = False
      i = len(texto)
    i += 1
  if (palindromo):
    print("É palíndromo")
  else:
    print("Não é palíndromo")
```

Exercício 5

Dadas duas sequências de texto, remover a primeira ocorrência da primeira sequência na segunda e imprimir o resultado.

Exemplo:

“ai” e “Paiython” → “Python”

Resposta Ex. 5

```
def main():
  seq1 = input("Digite uma sequência:")
  seq2 = input("Digite outra sequência:")
  resultado = seq2
  if (seq1 in seq2):
     resultado= ""
     i = seq2.index(seq1)
     f = i + len(seq1)
     j = 0
     while j < i:
       resultado += seq2[j]
       j += 1
     j = f
     while j < len(seq2)):
       resultado += seq2[j]
       j += 1
  print(resultado)
```

Exercício 6

Dadas duas sequências de caracteres, contar quantas vezes a primeira sequência está contida na segunda.

Resposta Ex. 6

```
def main():
  seq1 = input("Digite uma sequência de caracteres:")
  seq2 = input("Digite uma sequência de caracteres:")
  repeticoes = 0
  i = 0
  while i < len(seq2):
      ii = i
      j=0
      while j < len(seq1):
        if (ii < len(seq2) and seq1[j] != seq2[ii]):
          break
        elif (ii < len(seq2)):
          ii += 1
        j += 1
      if (ii - i == len(seq1)):
        repeticoes += 1
      i += 1
  print("Exitem ", repeticoes, "repeticoes da primeira sequencia na segunda.")
```