Strings. Soluções aos exercícios

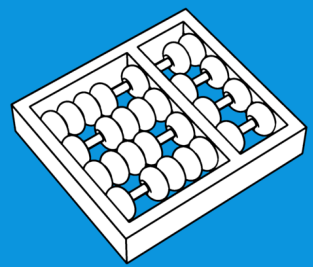
MC102 - Algoritmos e Programação de Computadores

04/25

13

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

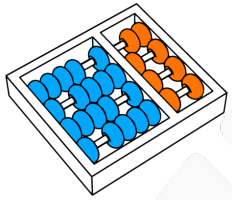
UNICAMP



```
print("You can't print this "text" without error.")
SyntaxError: invalid syntax
```

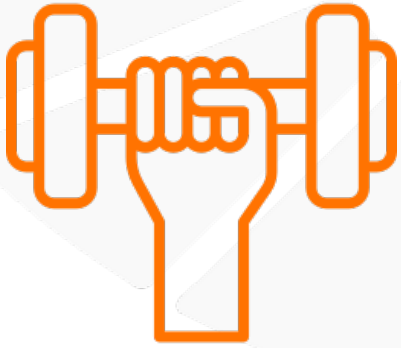
```
print('You can't print this "text" without error.')
SyntaxError: invalid syntax
```

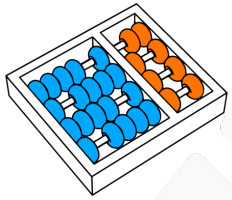
Exercícios



Strings

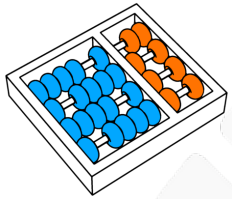
Soluções para os exercícios!





Exercícios de implementar operações com strings

1. Faça uma função que, dado uma string **sep** e uma lista **l** de strings, concatena as strings de **l** usando **sep** como separador:
   - Ex: Se **l == ["A", "B", "C"]** e **sep == ","**, então o resultado deve ser **"A,B,C"**.
2. Faça uma função que, dadas duas strings **s** e **t**, verifica se **s** é prefixo de **t**.
3. Faça uma função que, dadas duas strings **s** e **t**, verifica se **s** é substring de **t**.
4. Faça uma função que, dada uma string **s** e uma string **sep**, devolve uma lista resultante da quebra de **s** em uma ou mais strings, em cada ocorrência de **sep**:
   - Ex: Se **s == "A,B,C"** e **sep == ","**, então o resultado deve ser **["A", "B", "C"]**.



Soluções

Concatena com separador:

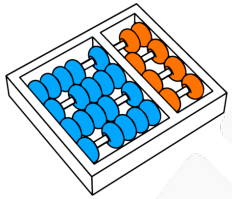
```
1  def concatena(sep, lista):
2      s = ''
3      for i in range (len(lista) - 1):
4          s += lista[i] + sep
5      return s + lista[-1]
```

Prefixo (um prefixo deve ser menor ou igual e ter todos os caracteres iguais):

```
1  def prefixo(s, t):
2      if len(s) > len(t):
3          return False
4      for i in range(len(s)):
5          if s[i] != t[i]:
6              return False
7      return True
```

Substring (uma substring é prefixo de algum sufixo):

```
1  def substring(s, t):
2      for i in range(len(t)):
3          if prefixo(s, t[i:]):
4              return True
5      return False
```



Soluções. Continuação

Split:

```
1  def meu_split(s, sep):
2      l = [] # resultado
3      i = 0 # inicio da substrng atual
4      j = 0 # fim da substring atual
5      while j + len(sep) <= len(s):
6          if s[j: j + len(sep)] == sep: # se encontramos o separador, separamos
7              l.append(s[i:j]) # adicionamos a substring na lista
8              j += len(sep) # atualizamos para continuar após o fim do separador
9              i = j # atualizamos o começo da nova substring
10         else: # senão, tem mais um caractere na substring atual
11             j += 1
12     l.append(s[i:]) # adicionamos a última substring
13     return l
```

Strings. Soluções aos exercícios

MC102 - Algoritmos e Programação de Computadores

04/25

13

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP