TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 6ª feira 15:30-17:00
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando funcionais sobre listas, escreva a função `nega_pares`, que recebe uma lista de inteiros positivos e devolve uma outra lista com os inteiros pares da lista negados. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> nega_pares([1, 2, 3, 4, 5, 6, 7, 8])
[-2, -4, -6, -8]
```

Solução:

```
def nega_pares(lst):
    return transforma(lambda x: -x, \
                      filtra(lambda x : x%2 == 0, lst))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 4ª feira 08:00-09:30
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando os funcionais sobre listas escreva a função `multiplica_pares_lista`, que recebe uma lista de números e devolve o produto dos elementos da lista que são pares. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> multiplica_pares_lista([1, 2, 3, 4, 5, 6])
48
>>> multiplica_pares_lista([1, 2, 3, 5, 7])
2
```

Solução:

```
def multiplica_pares_lista(lst):
    return acumula(lambda x, y: x * y, \
                   filtra(lambda x: x % 2 == 0, lst))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 2ª feira 10:30-12:00
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando os funcionais sobre listas escreva a função `conta_multiplos`, que recebe uma lista de inteiros e um número inteiro *n*, e devolve o número de elementos da lista que são múltiplos de *n*. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> conta_multiplos([1, 2, 3, 4, 5, 6], 3)
2
```

Solução 1:

```
def conta_multiplos(lst, n):
    return len(filtra( lambda x: x % n== 0, lst))
```

Solução 2:

```
def conta_multiplos(lst, n):
    return acumula(lambda x,y: x+y, \
                   transforma(lambda x: 1, \
                   filtra(lambda x: x % n == 0, lst)))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 2ª feira 9:00-10:30
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando os funcionais sobre listas escreva a função `num_ocorrencias`, que recebe uma lista e um elemento e devolve o número de vezes que o elemento ocorre na lista. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> num_ocorrencias([1, 5, 5, 4, 5], 5)
3
>>> num_ocorrencias([1, 5, 5, 4, 5], 7)
0
```

Solução 1:

```
def num_ocorrencias(lst,el):
    return len(filtra(lambda x: x == el, lst))
```

Solução 2:

```
def num_ocorrencias(lst, el):
    return acumula(lambda x,y: x+y, \
                   transforma(lambda x: 1, \
                   filtra(lambda x: x == el, lst)))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 5ª feira 10:30-12:00
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando funcionais sobre listas escreva uma função de ordem superior, `satisfaz_entre_vals`, que recebe um número inteiro positivo *inf,* um número inteiro positivo maior *sup* e um predicado *pred*, e devolve a lista de inteiros positivos entre *inf (inclusive) e sup (inclusive)* que satisfazem o predicado *pred*. Por exemplo:

```
>>> satisfaz_entre_vals(10, 30, lambda x : x%5==0)
[10, 15, 20, 25, 30]
>>> satisfaz_entre_vals(1, 5, lambda x : x%2==1)
[1, 3, 5]
```

Solução 1:

```
def satisfaz_entre_vals(inf, sup, pred):
    return filtra(pred, list(range(inf, sup+1)))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 6ª feira 14:00-15:30
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Usando funcionais sobre listas, escreva a função `quadrados_menores_que`, que recebe uma lista de inteiros e um número inteiro positivo `n` e devolve uma lista com os quadrados dos elementos menores que `n` da lista recebida. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> quadrados_menores_que([2, 3, 4, 9, 15], 5)
[4, 9, 16]
>>> quadrados_menores_que([2, 3, 4, 9, 15], 20)
[4, 9, 16, 81, 225]
```

Solução:

```
def quadrados_menores_que(l, n):
    return transforma(lambda x : x*x, filtra(lambda x: x < n, l))
```

TÉCNICO LISBOA

Fundamentos de Programação - 2019/2020 Aula Prática 09 (30 minutos) Turno 6ª feira 12:30-14:00
Nome:
Número:
Data:
Curso:

Capítulo 6.2 - Funções de ordem superior

Utilizando funcionais sobre listas, escreva a função `produto_numeros`, que recebe uma lista *lst* de números inteiros e um predicado *pred* e devolve o produto de todos os elementos de *lst* que satisfazem o predicado. A sua função deve conter apenas uma instrução, a instrução `return`. Por exemplo:

```
>>> produto_numeros([1, 2, 3, 4, 5], lambda x: x%2!=0)
15
>>> produto_numeros([2, 3, 4, 9], lambda x: x%3==0)
27
```

Solução:

```
def produto_numeros(pred, lst):
      return acumula(lambda x,y : x*y, filtra(pred, lst))
```