TÉCNICO LISBOA

Fundamentos da Programação
Ano lectivo 2020-21
Segundo Exame
19 de Fevereiro de 2021

Aluno ist199096

Pergunta 1 (val. 1)

```
Usando um ou mais dos funcionais sobre listas (filtra, transforma, acumula),
escreva a função num_cubos_maiores, que recebe uma lista de inteiros e um
inteiro, n, e devolve o número dos cubos dos seus elementos que são maiores
que n. A sua função deve conter apenas uma instrução, a instrução return.
Não é necessário validar os dados de entrada. Por exemplo:
>>> num_cubos_maiores([1, 2, 4, 3], 30)
1
```

Pergunta 2 (val. 3.0)

```
Suponha que existe o predicado de um argumento eh_primo, que recebe um
número natural e devolve verdadeiro apenas se o seu argumento é um número
primo.
Escreva a função soma_nao_primos que recebe um número inteiro positivo, n, e
devolve a soma de todos os números inferiores ou iguais a n que não são
primos. Não é necessário validar os dados de entrada. Por exemplo,
>>> soma_nao_primos(10)
38
a) Usando recursão com operações adiadas (não pode utilizar a atribuição nem
os ciclos while e for).
b) Usando recursão de cauda (não pode utilizar a atribuição nem os ciclos
while e for).
c) Usando um processo iterativo.
```

Pergunta 3 (val. 1.5)

Responda verdadeiro ou falso cada uma das seguintes afirmações.
Cada resposta correta/errada conta/desconta 0.25 valores (mínimo 0 valores).
Indentifique as perguntas/respostas adequadamente.
3a: A recursão de cauda é igual de eficiente em espaço e em tempo que a recursão de operações adiadas.
3b: Em Python, os atributos de uma classe podem ser privados utilizando a palavra chave "private" antes do nome do atributo.
3c: As chaves de um dicionário em Python podem ser de tipos diferentes.

3d: Em Python, os dicionários estão ordenados pela chave em ordem ascendente.
3e: Um algoritmo de complexidade O(1), realiza o mesmo número de operações independentemente do tamanho da entrada.
3f: Em Python, os tuplos não podem conter listas porque são tipos imutáveis.

Pergunta 4 (val. 1)

```
Duas palavras de igual comprimento dizem-se amigas se o número de posições
em que os respetivos caracteres diferem for inferior a 10%. Escreva a função
amigas que recebe como argumentos duas cadeias de caracteres e devolve
verdadeiro se os seus argumentos corresponderem a palavras amigas e falso em
caso contrário. A função deve validar os dados de entrada. Por exemplo:
>>> amigas('amigas', 'amigas')
True
>>> amigas('amigas', 'asigos')
False
>>> amigas('amigos', 'amigo')
ValueError: argumentos invalidos
```

Pergunta 5 (val. 3.0)

```
Uma fila dupla é um tipo estruturado de dados constituído por uma sequência
de elementos. Os elementos da fila são retirados tanto do início da
sequência como do fim da sequência. Novos elementos podem ser adicionados
tanto no início da sequência como no fim da sequência. Uma fila dupla tem
dois elementos que se distinguem dos restantes, o elemento do início da fila
e o elemento do fim da fila.

Considere que o tipo abstrato fila dupla tem as seguintes operações básicas:

* Construtores:
- nova_fila_dupla (): devolve uma fila dupla sem elementos.

* Seletores:
- primeiro(fila) devolve o primeiro elemento da fila.  Esta operação não
altera a fila. Se a fila não tiver elementos esta operação é indefinida.
ultimo(fila) devolve o último elemento da fila.  Esta operação não altera a
fila. Se a fila não tiver elementos esta operação é indefinida.

- comprimento(fila) devolve o número de elementos da fila.

* Modificadores:
```

- entra_inicio_fila(fila, el) altera destrutivamente a fila que é seu argumento para a fila que resulta da inserção do elemento el no início da fila. Devolve a fila resultante.

- entra_fim_fila(fila, el) altera destrutivamente a fila que é seu argumento para a fila que resulta da inserção do elemento el no fim da fila. Devolve a fila resultante.

- sai_inicio_fila(fila) altera destrutivamente a fila que é seu argumento para a fila que resulta da remoção do elemento no início da fila. Devolve a fila resultante. Se a fila não tiver elementos, esta operação é indefinida.

- sai_fim_fila(fila) altera destrutivamente a fila que é seu argumento para a fila que resulta da remoção do elemento no fim da fila. Devolve a fila resultante. Se a fila não tiver elementos, esta operação é indefinida.

* Reconhecedores:
- eh_fila_dupla(arg) recebe como argumento um elemento de um tipo qualquer e decide se este pertence ou não ao tipo fila dupla.

- eh_fila_vazia(fila) recebe como argumento uma fila e decide se esta corresponde à fila sem elementos (a fila gerada por nova_fila_dupla).

Não consideramos testes nas operações básicas.

a) Escolha uma representação para filas duplas
b) Escreva as operações básicas em termos da sua representação
c) Escreva a função junta_filas_duplas que recebe duas filas duplas e devolve a fila resultante de inserir a fila que é o segundo argumento no final da fila que é o primeiro argumento. A fila é inserida do final da fila para o início.

Pergunta 6 (val. 2)

Escreva a função recursiva conta_duplicados que recebe uma lista de inteiros, lst, e devolve informação sobre os elementos repetidos na forma de uma lista em que cada elemento é uma lista de dois elementos contendo o elemento que está repetido e o número de vezes que este elemento aparece na lista. Se a lista não tiver elementos repetidos, a sua função deve devolver a lista vazia. Na sua função não pode utilizar a atribuição nem os ciclos while e for. Deve validar de modo recursivo o argumento da sua função. Por exemplo:

```
>>> conta_duplicados([2, 3, 5, 3, 2, 2, 4])
```

```
[[2, 3], [3, 2]]
>>> conta_duplicados([2, 3, 'a', 3, 2, 2, 4])
ValueError: argumentos inválidos
```

SUGESTÃO: A contagem do número de ocorrências de um certo elemento numa lista pode-se realizar definindo uma função recursiva auxiliar para este efeito. Também pode ser útil a definição de outras funções auxiliares, por exemplo para retirar certos elementos de uma lista. Todas as suas funções auxiliares devem ser recursivas.

Pergunta 7 (val. 2.5)

Uma "slot-machine" é uma máquina de jogar que existe nos casinos. A slot-machine tem um depósito com moedas de um Euro, depósito esse que tem uma capacidade desejável e uma capacidade máxima. Quando a slot-machine é ligada, o depósito tem a capacidade desejável. Para além disso, a slot-machine mostra três rodas, cada uma delas com uma sequência de inteiros entre 1 e 50. Para se jogar, insere-se uma moeda de um Euro, a qual é adicionada ao depósito, e a slot-machine movimenta as suas rodas, escolhendo aleatoriamente um número de cada uma delas. Se entre os três números escolhidos aparecerem dois números iguais, o jogador recebe 10 Euros, que são removidos do depósito; se os três números escolhidos forem iguais, o jogador recebe 100 Euros, que são removidos do depósito; se os três números forem sete, o jogador recebe todo o dinheiro que está no depósito. Se o deposito ficar com uma quantidade de dinheiro inferior à capacidade desejável, a slot-machine fica inoperável até que o seu depósito seja preenchido com a capacidade desejável.

Para a geração de números aleatórios, pode utilizar a função de dois argumentos randint(a, b) do módulo random, que devolve um valor inteiro no intervalo [a,b].

Defina a classe \slot-machine" com os seguintes métodos:

- __init__: recebe dois inteiros correspondentes à capacidade desejável e à capacidade máxima e, verificando os argumentos, cria uma instância de \slot-machine";

- joga: método sem argumentos que efetua uma jogada;

- estado: método sem argumentos que mostra o estado interno da instância;

- abastece: método que recebe um inteiro positivo (negativo) e coloca (

retira) esse valor no depósito, verificando as restrições.

Por exemplo,

```
>>> s = slot_machine(100, 104)
>>> s.joga()
Resultado: (42, 30, 38)
Azar
>>> s.estado()
Operacional
Depósito com 101 Euros
Capacidade desejável 100 Euros
Capacidade máxima 104 Euros
>>> s.joga()
Resultado: (37, 44, 9)
Azar
>>> s.joga()
Resultado: (24, 21, 34)
Azar
>>> s.joga()
Resultado: (12, 44, 50)
Azar
>>> s.joga()
Slot machine não operacional
>>> s.estado()
Aguarda manutenção
Depósito com 104 Euros
Capacidade desejável 100 Euros
Capacidade máxima 104 Euros
>>> s.abastece(-9)
>>> s.estado()
Aguarda manutenção
Depósito com 95 Euros
Capacidade desejável 100 Euros
Capacidade máxima 104 Euros
```

Pergunta 8 (val. 1.5)

O primorial de um número natural n maior do que 1, representado por n#, é definido como o produto de todos os números primos menores ou iguais a n. Um número primo é um natural superior a 1 que apenas é divisível por si próprio e por 1. Escreva a função primorial que recebe como argumento um número inteiro maior que 1, n, e devolve o primorial de n. Não é necessário validar os dados de entrada. Por exemplo,

```
>>> primorial(2)
2
>>> primorial(4)
6
>>> primorial(5)
30
```

Pergunta 9 (val. 1.5)

Uma matriz é dita esparsa (ou rarefeita) quando a maior parte dos seus elementos é zero. As matrizes esparsas aparecem em grande número de aplicações em engenharia. Uma matriz esparsa pode ser representada por um dicionário cujas chaves correspondem a tuplos que indicam a posição de um elemento na matriz (linha e coluna) e cujo valor é o elemento nessa posição da matriz. Por exemplo, {(3, 2): 20, (150, 2): 6, (300, 10): 20} corresponde a uma matriz esparsa com apenas três elementos diferentes de zero.
Escreva uma função que recebe duas matrizes esparsas e devolve a sua soma. Por exemplo,

```
>>> e1 = {(1,5): 4, (2, 3): 9, (4, 1): 1}
>>> e2 = {(1, 6): 2, (4, 1): 2, (5,4): 2}
>>> soma_esparsas(e1, e2)
{(1, 5): 4, (1, 6): 2, (2, 3): 9, (4, 1): 3, (5, 4): 2}
```

Pergunta 10 (val. 1.5)

Considere a linguagem cujas frases são constituídas pelos símbolos ‘c’, ‘a’, ‘r’, ‘d’. As frases da linguagem começam pelo símbolo ‘r’, o qual é seguido por uma ou mais ocorrências dos símbolos ‘a’ e ‘d’, e terminam no símbolo ‘c’. Por exemplo ‘raaddaarc’ e ‘rdrc’ são frases da linguagem, ‘dc’ e ‘rdrr’ não o são.

a) Escreva uma gramática em notação BNF para a linguagem apresentada.

b) Escreva o predicado reconhece que recebe como argumento uma cadeia de caracteres e devolve verdadeiro apenas se a cadeia de caracteres pertence à linguagem. O predicado gera um erro se o seu argumento não for uma cadeia de caracteres.

Pergunta 11 (val. 1.5)

Escreva a função, soma_cumulativa, que recebe uma lista de números e que devolve uma lista que contém a soma cumulativa da lista recebida, ou seja, o elemento na posição i da lista devolvida contém a soma de todos os elementos da lista original nas posições de 0 a i. Não é necessário validar os argumentos. Por exemplo,

```
>>> soma_cumulativa([1, 2, 3, 4, 5])
[1, 3, 6, 10, 15]
```