Universidade Federal de Viçosa
Centro de Ciências Exatas e Tecnológicas
Departamento de Informática
INF101 – Introdução à Programação II
04/12/2017

Nome: _____ Matr.: _____ Turma Teórica: 2

2ª Prova (Valor: 30%)

1. (12%) Considere as estruturas de dados a seguir, representando, respectivamente, o estoque de uma quitanda e uma lista das vendas de um dia na mesma. No dicionário estoque, cada chave é o nome de um produto e o respectivo valor é uma lista contendo a quantidade do produto no estoque e, a seguir, o preço do produto. Na lista das vendas, para cada produto, estão listadas as quantidades dos mesmos que foram vendidas naquela transação. Em seguida, duas colunas devem ser relacionadas. A coluna da esquerda contém trechos de código que leem dados ou imprimem dados na tela, enquanto a coluna da direita contém descrições gerais do que cada trecho deve exibir na tela.

```
# Um exemplo de estoque da quitanda:
estoque = { 'tomate': [1000, 2.30],
            'alface': [500, 0.45],
            'batata': [2150, 1.20],
            'feijao': [100, 5.50]
          }
 # Um exemplo de vendas de um dia na quitanda:
vendas = [ ('batata', 10), ('tomate', 5), ('alface', 5), ('batata', 8),
           ('tomate', 15)
         ]
```

Observação: Os dados acima são meros exemplos. Para responder à questão, considere que os dados podem ser arbitrariamente outros.

( )
```
soma = 0
for v in vendas:
    prod, quant = v
    preco = estoque[prod][1]
    valor = preco * quant
    estoque[prod][0] -= quant
    soma += valor
print(soma)
```

( )
```
print(len(estoque))
```

( )
```
estoque = {}
arqEst = open('estoque.dat', 'r')
dados = arqEst.readline().split()
while dados:
    produto = dados[0].lower()
    quant = int(dados[1])
    preco = float(dados[2])
    estoque[produto] = [quant, preco]
    dados = arqEst.readline().split()
arqEst.close()
```

( )
```
for p in sorted(estoque.keys()):
    soma = 0
    for p0, q in vendas:
        if p0 == p:
            soma += q
    print(p, soma)
```

( )
```
for p in sorted(estoque.keys()):
    print (p, estoque[p][1], estoque[p][0])
```

( )
```
vendas = []
arqVendas = open('vendas.dat', 'r')
for linha in arqVendas.readlines():
    dados = linha.split()
    produto = dados[0].lower()
    quant = int(dados[1])
    vendas.append((produto, quant))
arqVendas.close()
```

( )
```
print(len(vendas))
```

( )
```
for p in sorted(estoque.keys()):
    soma = 0
    for p0, q in vendas:
        if p0 == p:
            soma += q * estoque[p][1]
    print(p, soma)
```

(A) Leitura dos dados do estoque

(B) Quantidade de vendas realizadas em um dia

(C) Valor total arrecadado em um dia

(D) Número de produtos distintos que a quitanda vende

(E) Leitura dos dados das vendas de um dia

(F) Número de unidades vendidas de cada produto em um dia

(G) Tabela exibindo o estoque atual

(H) Valor arrecadado com a venda de cada produto

2. (2%) Faça um círculo na(s) letra(s) correspondente(s) ao que for CORRETO:

   a) Todo arquivo de saída perde seu conteúdo ao ser aberto, caso ele já exista.
   b) Em média, para listas de valores aleatórios, o método de ordenação *quicksort* é o que apresenta a melhor performance.
   c) O método `readlines` de arquivos textos em Python transfere todo o conteúdo de um arquivo para a memória primária do computador.
   d) O método `write` do Python funciona para arquivos apenas de entrada.
   e) Em Python, um arquivo só pode ser aberto em apenas um modo de cada vez.

3. (2%) Faça um círculo na(s) letra(s) correspondente(s) ao que for INCORRETO:

   a) A estrutura de dados tupla em Python é imutável, assim como *string* também o é.
   b) A estrutura de dados dicionário em Python é mutável, assim como lista também o é.
   c) A estrutura de dados conjunto (*set*) em Python não permite repetição de um mesmo elemento na estrutura.
   d) A estrutura de dados arquivo em Python é permanente, no sentido de que, se a máquina em que ele resida for desligada, seu conteúdo não se apaga.
   e) A estrutura de dados dicionário em Python tem ordem, assim como as listas.

4. (6%) Complete as lacunas nas seguintes frases de modo que elas se tornem afirmativas verdadeiras:

   a) A estrutura de dados *dicionário* em Python é um tipo _____, enquanto que a estrutura _____ não permite que um valor do respectivo tipo possa ser mudado do valor com que ele(a) foi criado.
   b) A estrutura de dados *dicionário* em Python consiste em uma coleção de pares _____ e _____.
   c) A estrutura de dados *arquivo* reside normalmente em memória _____ .
   d) Uma *lista* em Python permite repetição de elementos em sua estrutura, mas a estrutura de dados _____ não permite.
   e) As operações típicas de *arquivo* em Python são: _____, _____, _____ e _____ .
   f) Os nomes de três métodos de ordenação de listas podem ser: _____, _____ e _____ .

5. (8%) Assinale com V (verdadeiro) ou F (falso) nas afirmações a seguir:

a) ( ) Seja uma lista Python armazenada na variável `lista1`. A operação `s = set(lista1)` tem, como efeito, transformar a lista em um conjunto.

b) ( ) O método de ordenação de listas denominado *mergesort* tem o tempo de execução proporcional a $n\log_2 n$, onde n o tamanho da lista.

c) ( ) O operador `in`, quando utilizado sobre um conjunto em Python, é o equivalente ao operador matemático da teoria de conjuntos "pertence a".

d) ( ) O método de ordenação de listas denominado *inserção direta* tem o tempo de execução proporcional a $n\log_2 n$, onde n é o tamanho da lista.

e) ( ) Uma tupla em Python, depois de criada, não pode mudar de tamanho (número de seus elementos).

f) ( ) Dicionários em Python não podem ter chaves numéricas.

g) ( ) Um dicionário em Python, depois de criado, não pode mudar de tamanho (número de seus elementos).

h) ( ) O Python tem embutido na própria linguagem o método *sort* e a função *sorted* para ordenar listas ou outras estruturas que podem ser iteradas.

i) ( ) Para criar um conjunto vazio em Python, podemos fazer a atribuição: `s = {}`

j) ( ) Conjunto (*set*) e dicionário em Python são sinônimos.