MAC 110 - Introdução à Computação
Primeiro Semestre de 2017– BMAC – IMEUSP - Prof.Marcilio
Prova 2 – 28/Junho/2017

Questão	1	2	3	4	5	6	Total
Valor	2	2	2	2	1	1	10
Nota							

Nome:_____ **NUSP:**_____

Questão 1 – 2 Pontos:

As funções abaixo devem comparar os primeiros n elementos de 2 listas **a** e **b**, devolvendo o índice do primeiro elemento diferente encontrado. Se todos forem iguais devolver **n**. Supor **n > 0** e que ambas as listas possuem pelo menos **n** elementos.
Exemplos:

1) Se n = 3, a = [4, 6, 3, 6, 7...] b = [4, 6, 5, 6, 7...] – devolver 2
2) Se n = 3, a = [4, 6, 3, 6, 7...] b = [4, 6, 3, 6, 7...] – devolver 3
3) Se n = 5, a = [4, 6, 3, 6, 7...] b = [9, 6, 3, 6, 7...] – devolver 0

Qual dos trechos abaixo está correto? Escreva acima de cada um deles **CERTO** ou **ERRADO**.

a) CERTO

```
def compara(a, b, n):
    for i in range(n):
        if a[i] != b[i]: return i
    return n
```

b) ERRADO

```
def compara(a, b, n):
    for i in range(n):
        if a[i] != b[i]: break
    return i
```

c) ERRADO

```
def compara(a, b, n):
    i = 0
    while a[i] == b[i]: i = i + 1
    return i
```

d) CERTO

```
def compara(a, b, n):
    i = 0
    while True:
        if i == n or a[i] != b[i]: return i
        i = i + 1
```

e) CERTO

```
def compara(a, b, n):
    i = 0
    while i < n and a[i] == b[i]: i = i + 1
    return i
```

f) **ERRADO**

```
def compara(a, b, n):
    i = 0
    while i < n:
        if a[i] == b[i]: i = i + 1
    return i
```

Questão 2 – 2 Pontos:

a) Escreva uma função **`TestaLinha (A, L, M)`** que recebe uma matriz **`A`** e verifica se a linha **`L`** de **`M`** elementos (**`A[L][0] .. A[L][M-1]`**) está em ordem crescente (onde cada elemento é menor ou igual ao seguinte) devolvendo **`True`** ou **`False`**.

```
def TestaLinha(A, L, M):
    # verifica se a linha L está em ordem crescente
    for k in range(M - 1):
        if A[L][k] > A[L][k + 1]: return False
    return True
```

b) Escreva uma função **`TestaColuna(A, C, N)`** que recebe uma matriz A e verifica se a coluna C com N elementos (A[0][C]..A[N-1][C]) está em ordem crescente (onde cada elemento é menor ou igual ao seguinte) devolvendo **`True`** ou **`False`**.

```
def TestaColuna(A, C, N):
    # verifica se a coluna C está em ordem crescente
    for k in range(N - 1):
        if A[k][C] > A[k + 1][C]: return False
    return True
```

c) Escreva a função **`TestaMatriz(A, N, M)`** que recebe uma matriz A de N linhas e M colunas e verifica se todas as linhas e todas as colunas estão em ordem crescente devolvendo **`True ou False.`** Use obrigatoriamente as funções **`TestaLinha`** e **`TestaColuna`**

```
def TestaMatriz(A, N, M):
    # verifica se as linhas e colunas de A estão em ordem
crescente

    # testa as linhas
    for k in range(N):
        # linha k
        if not TestaLinha(A, k, M): return False

    # testa as colunas
    for k in range(M):
        # coluna k
        if not TestaColuna(A, k, N): return False

    # todas as linhas ou colunas são crescentes
    return True
```

Questão 3 – 2 Pontos

Escreva uma função **`Testa_String(st)`** que verifica se a string **`st`** é composta apenas por letras maiúsculas, minúsculas e espaços, devolvendo **`True`** ou **`False`**. Exemplos:

```
sx = "Exemplo De String"
Testa_String(sx) # devolve True

sx = "Exemplo 2 De String"
Testa_String(sy) # devolve False
```

Solução 1:

```
def Testa_String(st):
    for x in st:
      if x not in " abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ":
            return False

    # se chegou aqui é porque todos são letras ou espaços
    return True
```

Solução 2:

```
def Testa_String1(st):
    for k in range(len(st)):
        if st[k] not in " abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ":
            return False

    # se chegou aqui é porque todos são letras ou espaços
    return True
```

Solução 3:

```
def Testa_String2(st):
    for k in range(len(st)):
        if not 'A' <= st[k] <= 'Z' and not 'a' <= st[k] <= 'z' and st[k] != ' ':
return False

    # se chegou aqui é porque todos são letras ou espaços
    return True
```

Questão 4 – 2 Pontos

Considere a seguinte função recursiva:

```
def FF(L):
    if len(L) == 0: return 0
    return L[0] + FF(L[1:])
```

a) Diga o que será impresso pelos comandos abaixo:

```
print(FF([1, 2, 3])) : 6
s = []
print(FF(s)) : 0
t = [-1, -2]
print(FF(t)) : -3
```

b) Escreva uma função **não recursiva** equivalente a FF acima.

Essa função devolve a soma dos elementos da lista L

```
def FF(L):
    soma  = 0
    for i in range(len(L)): soma = soma + L[i]
    return soma
```

Questão 5 – 1 Ponto

Considere a tabela abaixo com 10 elementos que está em ordem crescente

0	1	2	3	4	5	6	7	8	9

Diga quantas comparações são necessárias para procurar nesta tabela cada dígito diferente do seu NUSP, usando o algoritmo de busca binária. Exemplo: se seu NUSP = **4903423:**

Dígito 4 – xxx comparações
Dígito 9 – xxx comparações
Dígito 0 – xxx comparações
Dígito 3 – xxx comparações
Dígito 2 – xxx comparações

Digito	Comparações
0	3
1	2
2	3
3	4
4	1
5	3
6	4
7	2
8	3
9	4

Questão 6 – 1 Ponto:

Considere o método da bolha para classificar uma sequência. Este método pega cada um dos elementos a partir do segundo e vai trocando com o anterior até encontrar o seu lugar na sequência.

a) Dê uma sequência com os números 1,2,3,4,5 que para classificá-la sejam necessárias exatamente 4 trocas.

Exemplo: 3 2 1 5 4

b) Idem para uma sequência com 6 trocas

Exemplo: 4 3 2 1 5

c) Quantas trocas serão necessárias se a sequência estiver na ordem invertida: 5, 4, 3, 2, 1?

10 trocas

d) E se for uma sequência com n elementos: n, n-1, n-2, ..., 3, 2 ,1?

n * (n – 1) / 2