

MAC2166 – Introdução à Computação para Engenharia

ESCOLA POLITÉCNICA

Terceira Prova – 17 de junho de 2013

Nome: _____

Assinatura: _____

Nº USP: _____ Turma: _____

Professor: _____

Instruções:

1. Não destaque as folhas deste caderno.
2. A prova consta de 3 questões. Verifique antes de começar a prova se o seu caderno de questões está completo.
3. A prova pode ser feita a lápis. Cuidado com a legibilidade e, principalmente, com a TABULAÇÃO.
4. Qualquer questão pode ser resolvida em qualquer página. Se a questão não está na página correspondente ao enunciado basta indicar isto na página e escrever QUESTÃO X em letras ENORMES antes da solução.
5. Não é necessário apagar rascunhos no caderno de questões.
6. Não é permitido o uso de folhas avulsas para rascunho.
7. Não é permitido o uso de equipamentos eletrônicos.
8. Não é permitido a consulta a livros, apontamentos ou colegas.

DURAÇÃO DA PROVA: 2 horas



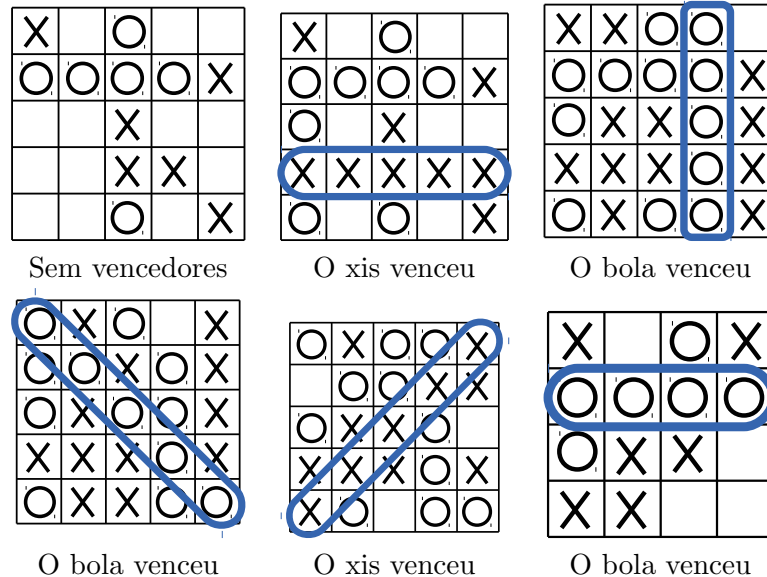
Questão	Valor	Nota
1	3,0	
2	3,0	
3	4,0	
Total	10,0	

Questão 1 (vale 3 pontos)

Considere um Jogo da Velha entre dois jogadores em um tabuleiro $n \times n$. Cada jogador é identificado por um símbolo, uma bola (O) ou um xis (X). Os jogadores jogam alternadamente. Em cada rodada, o jogador marca uma posição livre do tabuleiro com o seu símbolo. Vence o jogo quem conseguir preencher primeiro uma sequência de n símbolos, seja em linha (horizontal), coluna (vertical), ou em uma das duas diagonais.

O objetivo dessa questão é verificar quem foi o vencedor para uma dada configuração do tabuleiro, ou indicar se não há vencedores.

Exemplos:



Considere uma matriz quadrada $A_{n \times n}$ para representar o tabuleiro, onde cada posição armazena um dentre os seguintes caracteres (strings), 'O', 'X', ' ' (letra 'O' maiúscula para bola, letra 'X' maiúscula para xis, e caractere de espaço em branco para posição vazia).

Escreva uma função com cabeçalho:

```
def verifica_tabuleiro(Tab):
```

que recebe um tabuleiro Tab do Jogo da Velha $n \times n$ e retorna 'O', caso o jogador bola seja o vencedor, ou 'X' caso o jogador xis seja o vencedor ou 'N' caso não haja vencedor. Você pode assumir que exista no máximo um vencedor em Tab .

Questão 2 (vale 3 pontos)

Esta questão é composta por 2 itens.

item (a) (vale 1.5 ponto) Nesse item vamos simular o comportamento da função `split()` do Python, e portanto você NÃO deve utilizar a função `split()` para resolver essa questão.

Escreva uma função de cabeçalho

```
def recorta(texto_base, texto_buscado):
```

que recebe os strings `texto_base` e `texto_buscado` e retorna uma lista com os pedaços (strings) do `texto_base` “recortados” nas ocorrências de `texto_buscado` (as ocorrências do `texto_buscado` são eliminadas). Você pode considerar que o `texto_buscado` não é vazio.

Exemplos:

para `texto_base = 'a'` e `texto_buscado = 'b'`, a função deve retornar a lista: `['a']`

para `texto_base = 'a'` e `texto_buscado = 'a'`, a função deve retornar a lista: `['', '']` (string vazio antes e depois de 'a').

para `texto_base = 'mac'` e `texto_buscado = 'a'`, a função deve retornar a lista: `['m', 'c']`

para `texto_base = 'a,b,,d,'` e `texto_buscado = ','` (vírgula), a função deve retornar a lista: `['a', 'b', '', 'd', '']`

e para `texto_base = 'ana e banana'` e `texto_buscado = 'ana'`, a função deve retornar a lista: `['', ' e b', 'na']`

item (b) (vale 1.5 ponto)

Nesse item você deve utilizar a função do item anterior mesmo que não a tenha feito. Você não precisa reescrever a função, basta escrever o programa.

Escreva um programa que leia três strings: `frase`, `texto_velho` e `texto_novo`, e imprima um string onde todas as ocorrências do string `texto_velho` foram substituídos pelo string `texto_novo`.

Exemplos considerando a `frase = 'as rugas das tartarugas ninjas.'`:

para:

`texto_velho = 'rugas ninj'` e `texto_novo = 'ninjas rug'`, o programa deve imprimir
'as rugas das tartaninjas rugas.'

para: `texto_velho = 'rugas'` e `texto_novo = 'ninjas'`, o programa deve imprimir
'as ninjas das tartaninjas ninjas.'

para: `texto_velho = 's'` e `texto_novo = ''` (string vazio), o programa deve imprimir:
'a ruga da tartaruga ninja.'

Questão 3 (vale 4 pontos)

Essa questão é baseada no exercício programa 4.

Escreva um programa que leia o nome de dois arquivos. Você deve assumir que os dois arquivos contém matrizes no mesmo formato utilizado no EP4 (ou seja, matrizes quadradas $n \times n$). O primeiro nome especifica o nome de uma matriz `turtledorm` `tdorm` e o segundo nome especifica o nome de uma matriz `sol` candidata a solução do `tdorm`.

O seu programa deve carregar dos arquivos as duas matrizes `tdorm` e `sol` e verificar se a matriz `sol` é realmente uma solução de `tdorm`.

Exemplo: considere o conteúdo dos seguintes 3 arquivos com matrizes 3×3 :

1 1 0	1 0 0	0 0 0
1 0 0	0 0 0	0 0 0
0 0 0	0 0 0	0 0 1
<code>tdorm.txt</code>	<code>sol1.txt</code>	<code>sol2.txt</code>

O programa deve imprimir 'SIM' para a matriz no arquivo `tdorm.txt` e a matriz candidata à solução no arquivo `sol1.txt`, e imprimir 'NÃO' considerando o mesmo `tdorm.txt` para a matriz candidata à solução em `sol2.txt`.

Para resolver essa questão você pode utilizar a função `leia_turtledorm` sem implementá-la. A função recebe o nome de um arquivo e retorna uma matriz (o `turtledorm` definido no arquivo), tendo o cabeçalho:

```
def leia_turtledorm(nome_do_arquivo):
```

Caso ache conveniente, escreva e use outras funções adicionais.

