

**MAC2166 – Introdução à Computação**  
**Grande Áreas Civil, Mecânica, Petróleo e Química**  
ESCOLA POLITÉCNICA  
Terceira Prova – 22 de junho de 2016

Nome: \_\_\_\_\_

Assinatura: \_\_\_\_\_

Nº USP: \_\_\_\_\_ Turma: \_\_\_\_\_

Professor: \_\_\_\_\_

**Instruções:**

1. Não destaque as folhas deste caderno. A prova pode ser feita a lápis.
2. A prova consta de 3 questões. Verifique antes de começar a prova se o seu caderno está completo.
3. As questões podem ser resolvidas em qualquer página. Ao escrever uma solução (ou parte dela) em página diferente do enunciado, escreva QUESTÃO X em letras ENORMES junto da solução.
4. As soluções devem ser em Python. **Você pode usar apenas recursos de Python vistos em aula.** Você pode definir funções auxiliares e usá-las à vontade. Cuidado com a legibilidade e, principalmente, com a TABULAÇÃO.
5. As soluções não precisam verificar consistência de dados.
6. Não é permitido o uso de folhas avulsas para rascunho, a consulta a livros, apontamentos, colegas ou equipamentos eletrônicos. Desligue o seu celular e qualquer equipamento que possa perturbar o andamento da prova.

**DURAÇÃO DA PROVA: 2 horas**



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

### QUESTÃO 1 (vale 3,0 pontos)

Podemos calcular uma aproximação da raiz quadrada de um número através de uma série. A ideia deste método é devida a Isaac Newton, que propôs um método geral para encontrar zeros de funções, ou seja, números que quando aplicados na função o resultado obtido é zero.

O primeiro termo da série para achar a raiz quadrada de um número  $x$  é o próprio  $x$ . Dado um termo  $x_i$  qualquer, podemos calcular o próximo através da fórmula abaixo:

$$x_{i+1} = \frac{x_i + \frac{x}{x_i}}{2}.$$

Por exemplo, se aplicarmos o método para  $x = 4$ , vamos obter os seguintes termos:

$$\begin{aligned}x_0 &= 4 \\x_1 &= 2.5 \\x_2 &= 2.05 \\x_3 &= 2.000609756097561 \\x_4 &= 2.0000000929222947 \dots\end{aligned}$$

Escreva uma função, cujo protótipo está definido na página seguinte, que recebe um número real  $x$  e um número inteiro  $n \geq 0$  e retorna como aproximação da raiz quadrada de  $x$  o número real  $x_n$  obtido pelo método definido acima.

```
def raiz_quadrada(x, n):  
    '''  
        (float, int) -> float  
        Recebe um real x e um inteiro n e retorna o valor  
        de x_n como aproximacao para a raiz quadrada de x.  
    '''
```

## QUESTÃO 2 (vale 4,0 pontos)

Nesta questão vamos implementar uma classe para **números complexos**. Um número complexo é representado por um par  $(a, b)$  de números reais: sua parte real e sua parte imaginária. O número complexo representado é  $a + bi$  onde  $i = \sqrt{-1}$ . Considere abaixo a descrição da classe em Python:

```
class Complexo:
    """
    Classe que representa números complexos. Cada objeto da classe tem dois
    atributos p_real e p_imag, onde p_real tem a parte real e p_imag a parte
    imaginária do número.
    """

    def __init__(self, p_real = 0, p_imag = 0):
        """
        Construtor da classe. Recebe dois números reais e constroi um
        objeto da classe Complexo, com o par de números como atributos.
        """
        self.p_real = p_real
        self.p_imag = p_imag

    def __str__(self):
        """
        Retorna um string para impressão de um número complexo.
        """
        if self.p_imag > 0:
            texto = "%.2f + %.2fi" %(self.p_real, self.p_imag)
        elif self.p_imag < 0:
            texto = "%.2f - %.2fi" %(self.p_real, -self.p_imag)
        else:
            texto = "%.2f" %(self.p_real)
        return texto
```

```

def __add__(self, other):
    '''
        (Complexo, Complexo) -> Complexo
        Retorna a soma dos números complexos self e other.
        Exemplos:
        (4 - 3i) + (2 + 4i) = 6 + i
        (12 - 5i) + (0 + 3i) = 12 - 2i
        (2 - 1i) + (-2 + 0i) = 0 - 1i
    '''
    #####
    ##### Parte a. Implemente este metodo #####
    #####

```

```

def __mul__(self, other):
    '''
        (Complexo, Complexo) -> Complexo
        Retorna o produto dos números complexos self e other.
        Exemplos:
        (3 - 2i) * (5 + 4i) = 23 + 2i
        (12i) * (3 - i) = 12 + 36i
    '''
    #####
    ##### Parte b. Implemente este metodo #####
    #####

```

```

def modulo(self):
    """
    (Complexo) -> float
    Retorna o módulo do número complexo self.
    Exemplos:
    O módulo de  $4 - 3i$  é 5.
    O módulo de  $3i$  é 3.
    O módulo de  $1 + i$  é 1.4142...

    Voce pode usar a funcao da questao 1 mesmo que nao a
    tenha feito
    """
    #####
    ##### Parte c. Implemente este metodo #####
    #####

```

```

def prod_escalar (self, c):
    """
    (Complexo, float) -> Complexo
    Retorna o produto do complexo self com o escalar c.
    Exemplos:
    Para o complexo  $(2 + 3i)$  e  $c = -2$ , deve retornar  $-4 - 6i$ .
    Para o complexo  $(10 + 2i)$  e  $c = 0$ , deve retornar  $0 + 0i$ .
    """
    #####
    ##### Parte d. Implemente este metodo #####
    #####

```

```

def exponencial(self):
    '''
        (Complexo) -> Complexo
        Retorna uma aproximação para e^self usando a expansão de Taylor:
        e^self = 1 + self + self^2/2! + self^3/3! + self^4/4! + ... + self^n/n! + ...
        Inclua na aproximação todos os termos até o primeiro que seja,
        em módulo, menor ou igual a 10^{-6}.
    '''
    #####
    ##### Parte e. Implemente este metodo #####
    #####

```

**QUESTÃO 3** (vale 3,0 pontos)

Dizemos que uma matriz  $A$  com  $m$  linhas e  $n$  colunas tem banda nula  $k \leq m$  se as  $k$  diagonais de  $A$ , começando a partir do canto inferior esquerdo, são nulas. Veja os exemplos abaixo:

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 3 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \text{ tem banda nula } 3.$$

$$\begin{pmatrix} 4 & 2 & 1 & 5 \\ 0 & 3 & 2 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \text{ tem banda nula } 2.$$

$$\begin{pmatrix} 2 & 1 \\ 0 & 2 \\ 2 & 1 \\ 0 & 0 \end{pmatrix} \text{ tem banda nula } 1.$$

$$\begin{pmatrix} 2 & 1 \\ 0 & 2 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \text{ tem banda nula } 3.$$

$$\begin{pmatrix} -1 & 2 & 1 \\ 0 & 3 & 2 \\ 0 & 4 & 1 \\ 3 & 0 & 0 \end{pmatrix} \text{ tem banda nula } 0.$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \text{ tem banda nula } 2.$$

Escreva um programa que lê inteiros positivos  $m$  e  $n$  e uma matriz  $A_{m \times n}$ , e imprime o maior  $k$  tal que  $0 \leq k \leq m$  e a matriz  $A$  tem banda nula  $k$ .



