UNIVERSIDADE FEDERAL DO CARIRI

EXERCÍCIO DE DEPURAÇÃO - LABORATÓRIO DE PROGRAMAÇÃO

ARIANE KEVINNY MUNIZ RIBEIRO - 2019002827 - CIÊNCIAS DA COMPUTAÇÃO

Primeira

Bug: Ao inserir o primeiro valor, ao invés de imprimir, ele pedia novamente para inserir um valor e só a partir daí os valores eram impressos.

Código

```
def double_input():
    x = int(input("Digite um número: "))
    while (x > -1):
        x = int(input("Digite um número: "))
        print(x * 2)

double_input()
```

Solução: Ao inverter a os comandos dentro laço de repetição faz com que o primeiro valor de x válido já seja impresso, e só depois o programa pede um novo valor.

Código solucionado:

```
def double_input():
    x = int(input("Digite um número: "))
    while (x > -1):
        print (x * 2)
        x = int(input("Digite um número: "))

double_input()
```

Segunda

Bug:

Ao inserir valor menor ou igual a 50 a mensagem impressa é "Menor que 50" ao invés de "Menor ou igual a 50"

Código:

```
def maior_50():
    x = int(input("Digite um inteiro: "))
    if x > 50:
        print("Maior que 50")
    else:
        print("Menor que 50")
```

```
maior_50()
```

Solução:
Mudar a mensagem impressa na linha 6, segundo print.

Código solucionado:
```
def maior_50():
    x = int(input("Digite um inteiro: "))
    if x > 50:
        print("Maior que 50")
    else:
        print("Menor ou igual a 50")

maior_50()
```

Terceira
Bug: O usuário só consegue inserir um valor, pois o programa entra em loop.

Código:
```
def double_input():
    num = float(input("Digite um número: "))
    while num > 0:
        print(num*2)


double_input()
```

Solução:
A primeira mudança foi criar uma lista de valores a ser impressos depois que o usuário inserir todos os valores desejados, e a segunda mudança foi criar um laço infinito porém com uma condição de parada.

Código solucionado:
```
def double_input():
    lista = []
    while True:
        num = float(input("Digite um número:"))
        if (num > 0):
            lista.append(num*2)
        else:
            break
    print (lista)

double_input()
```

Quarta
Bug:
1 - Pede apenas 10 valores ou invés de 100
2 - Calcula erroneamente o valor da média

Código:
```
def media_100():
    media = 0
    for i in range(0, 10):
        numero = float(input("Digite um número: "))
        media = numero + numero/10
    print("A média é {}".format(media))

media_100()
```

Solução:
1 - Trocar o valor de 10 para 100 na linha 3.
2 - Incrementar os valores inseridos dentro da variável média, divididos por 100.

Código solucionado:
```
def media_100():
    media = 0
    for i in range(0, 100):
        numero = float(input("Digite um número: "))
        media += numero/100
    print("A média é {}".format(media))

media_100()
```

Quinta
Bug:
1 - Verifica somente as 5 primeiras idades inseridas.
2 - Não considera a idade = 18 na condição de impressão.

Código:
```
def maior_idade():
    idades = []
    for i in range(0,50):
        idades.append(int(input("Digite sua idade: ")))
    for i in range(0,5):
        if idades[i] > 18:
            print(idades[i])
```

maior_idade()

Solução:
1 - Na linha 6, alterar range(0,5) para range(0,50) para verificar toda a lista de idade.
2 - Na linha 7, alterar > para >= na condicional.

Código solucionado:
```
def maior_idade():
    idades = []
    for i in range(0,50):
        idades.append(int(input("Digite sua idade: ")))
    for i in range(0,50):
        if idades[i] >= 18:
            print(idades[i])

maior_idade()
```

Sexta
Bug:
A lista passada como parâmetro na função com **n** valores, possui índices de 0 a n-1, assim quando o valor da variável **i** for 0, o valor de j vária de 0 a n, assim se tenta manipular a lista num índice que está fora do intervalo

Código:
```
def bubbleSort(lista):
    n = len(lista)
    for i in range(n - 1):
        for j in range(0, n - i):
            if lista[j] > lista[j + 1]:
                lista[j], lista[j + 1] = lista[j + 1], lista[j]

lista = [64, 34, 25, 12, 22, 11, 90]
bubbleSort(lista)
print("A lista ordenada é: {}".format(lista))
```

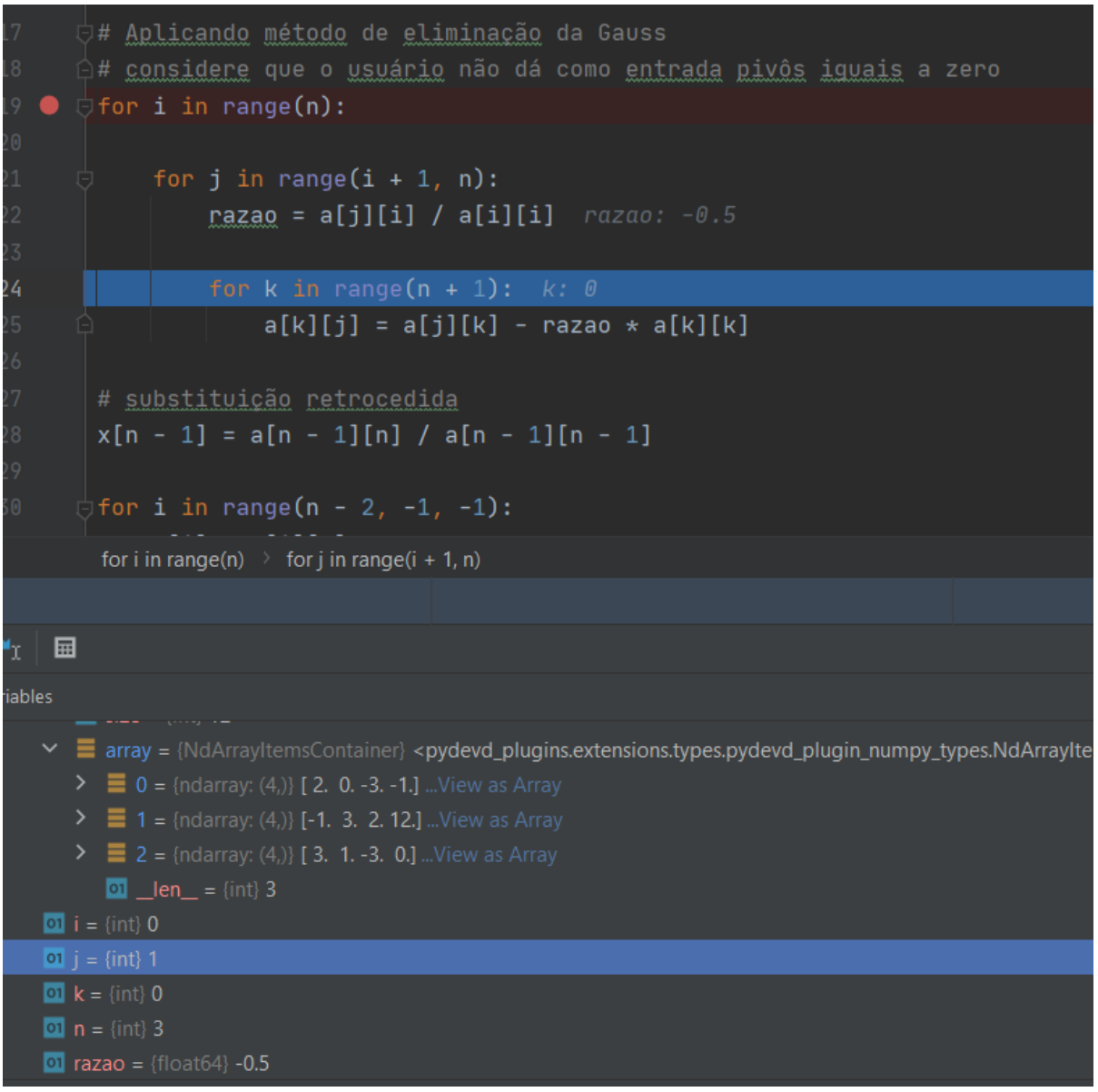
Solução:
Na linha 4, no segundo for, o valor de n é trocado para n - 1.

Código solucionado:
```
def bubbleSort(lista):
    n = len(lista)
    for i in range(n - 1):
        for j in range(0, (n -1)- i):
            if lista[j] > lista[j + 1]:
```

```
                lista[j], lista[j + 1] = lista[j + 1], lista[j]

lista = [64, 34, 25, 12, 22, 11, 90]
bubbleSort(lista)
print("A lista ordenada é: {}".format(lista))
```

Sétima
Bug:
Depois de instalar o numpy através do comando 'pip install numpy'



Nesse momento em que se aplica pela primeira vez o método de eliminação de Gauss o elemento da posição a[0][1] (k = 0, e j = 1) é zerado, porém o esperado é que o elemento a[1][0] é que seja, já que esse método consiste em: Transformar a matriz aumentada [A | b] em uma matriz aumentada na forma [A^{-} | b^{-}] que é uma matriz triangular superior.[1].

[1] Informação retirada do link https://pt.wikipedia.org/wiki/Elimina%C3%A7%C3%A3o_de_Gauss

Depois de zerar o elemento triangular superior, o programa segue alterando os valores da coluna, onde ocorre o segundo erro, já que pelo método os elementos são modificados em relação a razão no sentido das linhas.
O terceiro fator é que a matriz é 3x4, assim, quando chega na linha 25, a[k][j] = a[j][k] - razao * a[k][k], sendo k = [0, 3] e j = [1, 3]. e o k = 3 e j = 2, gera a[3][2] = a[2][3] - razao * a[3][3]. Porém os elementos a[3][2], a[3][3] não fazem parte da matriz.

Código:

```
import numpy as np

n = int(input('Digite o número de incógnitas: '))

#cria matriz de zeros
a = np.zeros((n, n + 1))

#cria vetor solução com zeros
x = np.zeros(n)

# Lê coeficientes da matriz aumentada
print('Entre com os coeficientes da matriz aumentada:')
for i in range(n):
    for j in range(n + 1):
        a[i][j] = float(input('a[' + str(i) + '][' + str(j) + ']='))

# Aplicando método de eliminação da Gauss
# considere que o usuário não dá como entrada pivôs iguais a zero
for i in range(n):

    for j in range(i + 1, n):
        razao = a[j][i] / a[i][i]

        for k in range(n + 1):
            a[k][j] = a[j][k] - razao * a[k][k]

# substituição retrocedida
x[n - 1] = a[n - 1][n] / a[n - 1][n - 1]

for i in range(n - 2, -1, -1):
    x[i] = a[i][n]

    for j in range(i + 1, n):
        x[i] = x[i] - a[i][j] * x[j]

    x[i] = x[i] / a[i][i]
```

```
# mostrando solução
print('O vetor solução é: ')
for i in range(n):
    print('X[{}] = {:.2f}'.format(i, x[i]), end='\t')
```

Solução:
Me baseei no passo a passo do método de Gauss e assumindo os elementos **i j k** para caminhar pela matriz de 3 formas diferentes, sendo **j** a linha dos elementos a serem modificados e **k** a sua respectiva coluna, e utilizando a variável **i** como a linha do elemento da diagonal utilizada na linha 22 (razao = a[j][i] / a[i][i]).

Código Solucionado:
```
for i in range(n):

    for j in range(i + 1, n):
        razao = a[j][i] / a[i][i]

        for k in range(n + 1):
            a[j][k] = a[j][k] - razao * a[i][k]
```

Código Solucionado:
```
import numpy as np

n = int(input('Digite o número de incógnitas: '))

#cria matriz de zeros
a = np.zeros((n, n + 1))

#cria vetor solução com zeros
x = np.zeros(n)

# Lê coeficientes da matriz aumentada
print('Entre com os coeficientes da matriz aumentada:')
for i in range(n):
    for j in range(n + 1):
        a[i][j] = float(input('a[' + str(i) + '][' + str(j) + ']='))

# Aplicando método de eliminação da Gauss
# considere que o usuário não dá como entrada pivôs iguais a zero
for i in range(n):

    for j in range(i + 1, n):
        razao = a[j][i] / a[i][i]
```

```
        for k in range(n + 1):
            a[j][k] = a[j][k] - razao * a[i][k]

# substituição retrocedida
x[n - 1] = a[n - 1][n] / a[n - 1][n - 1]

for i in range(n - 2, -1, -1):
    x[i] = a[i][n]

    for j in range(i + 1, n):
        x[i] = x[i] - a[i][j] * x[j]

    x[i] = x[i] / a[i][i]

# mostrando solução
print('O vetor solução é: ')
for i in range(n):
    print('X[{}] = {:.2f}'.format(i, x[i]), end='\t')
```

Terminal - Sétimo Problema

```
O vetor solução é:
X[0] = 1.00    X[1] = 3.00    X[2] = 2.00
Process finished with exit code 0
```