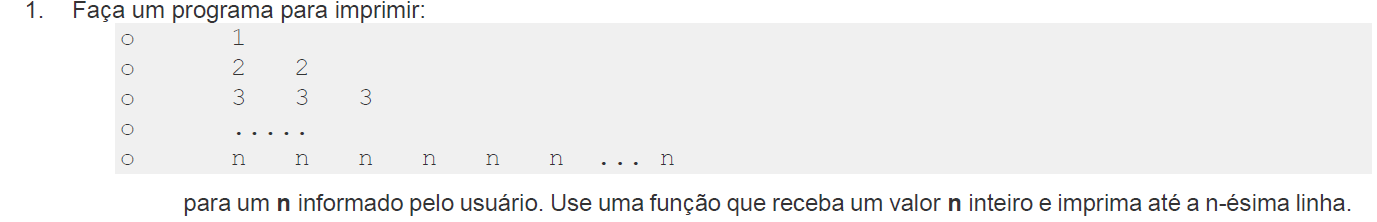
BEATRIZ HOLANDA, EMANUEL, RAFAEL E SARA

SOMATIVA PARTE 1



x = int(input("Digite um número:"))

def piramide():

  res = ""

  aux = 1

  for y  in range(x):

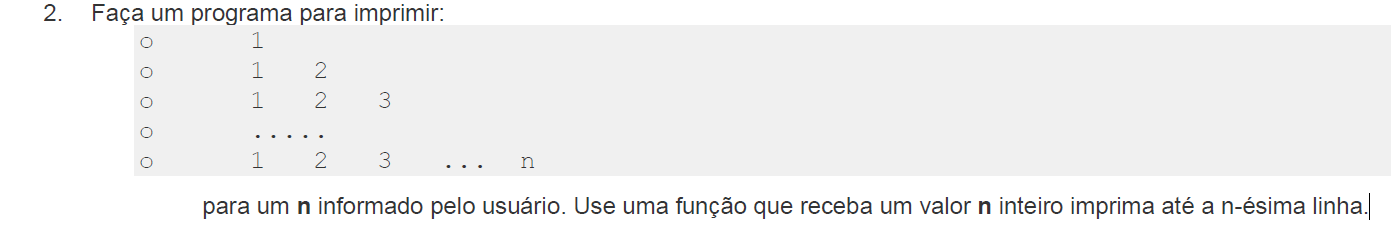
      for y in range(aux):

        res = f"{aux} "\*(aux)

      print(res)

      aux = aux + 1

piramide()



x = int(input("Digite um número:"))

def piramide():

  res = ""

  aux = 1

3. Faça um programa, com uma função que necessite de três argumentos, e que forneça a soma desses três argumentos.

def valores (a,b, c):

    print(a+b+c)

valores(15,30,55)

4. Faça um programa, com uma função que necessite de um argumento. A função retorna o valor de caractere ‘P’, se seu argumento for positivo, e ‘N’, se seu argumento for zero ou negativo.

5. Faça um programa com uma função chamada somaImposto. A função possui dois parâmetros formais: taxaImposto, que é a quantia de imposto sobre vendas expressa em porcentagem e custo, que é o custo de um item antes do imposto. A função “altera” o valor de custo para incluir o imposto sobre vendas.

def somaimposto(taxaimposto, custo):

    total=0

    imposto=0

    imposto= taxaimposto \* custo

    total= custo + imposto

    print(f"O valor com imposto é R${total}")

somaimposto(0.15, 500)

somaimposto(0.42, 80)

6. Faça um programa que converta da notação de 24 horas para a notação de 12 horas. Por exemplo, o programa deve converter 14:25 em 2:25 P.M. A entrada é dada em dois inteiros. Deve haver pelo menos duas funções: uma para fazer a conversão e uma para a saída. Registre a informação A.M./P.M. como um valor ‘A’ para A.M. e ‘P’ para P.M. Assim, a função para efetuar as conversões terá um parâmetro formal para registrar se é A.M. ou P.M. Inclua um loop que permita que o usuário repita esse cálculo para novos valores de entrada todas as vezes que desejar

7. Faça um programa que use a função valorPagamento para determinar o valor a ser pago por uma prestação de uma conta. O programa deverá solicitar ao usuário o valor da prestação e o número de dias em atraso e passar estes valores para a função valorPagamento, que calculará o valor a ser pago e devolverá este valor ao programa que a chamou. O programa deverá então exibir o valor a ser pago na tela. Após a execução o programa deverá voltar a pedir outro valor de prestação e assim continuar até que seja informado um valor igual a zero para a prestação. Neste momento o programa deverá ser encerrado, exibindo o relatório do dia, que conterá a quantidade e o valor total de prestações pagas no dia. O cálculo do valor a ser pago é feito da seguinte forma. Para pagamentos

# arrumar o calculo

def valorPagamento():

    vlr\_prest = 1

    vlr\_tot = 0

    while(vlr\_prest!= 0):

        vlr\_pag = 0

        if vlr\_prest == 0:

            break

        vlr\_prest = float(input("Valor prestação: "))

        d\_atr = int(input("Dias em atraso: "))

        if d\_atr == 0:

            vlr\_pag = vlr\_prest

            vlr\_tot = vlr\_tot + vlr\_pag

        elif d\_atr!= 0:

            vlr\_pag = (vlr\_prest + (vlr\_prest \* 0.03)) + (d\_atr \* 0.001)

            vlr\_tot = vlr\_tot + vlr\_pag

    print(f"\t Relatório do dia:\n",

    f"Valor total das prestações pagas = {vlr\_tot:.2f}")

valorPagamento()

sem atraso, cobrar o valor da prestação. Quando houver atraso, cobrar 3% de multa, mais 0,1% de juros por dia de atraso.

8. Faça uma função que informe a quantidade de dígitos de um determinado número inteiro informado.

def numInt():

    num=str(input("Digite um numero: "))

    print(f"A quantidade de digitos desse número é {num.\_\_len\_\_()}")

numInt()

9. Reverso do número. Faça uma função que retorne o reverso de um número inteiro informado. Por exemplo: 127 -> 721.

num = str(input("Digite um número:"))

x = num.\_\_len\_\_() - 1

res = ""

while x != -1:

    res = res + f"{num[x]}"

    x = x - 1

print(res)

10. Jogo de Craps. Faça um programa de implemente um jogo de Craps. O jogador lança um par de dados, obtendo um valor entre 2 e 12. Se, na primeira jogada, você tirar 7 ou 11, você é um "natural" e ganhou. Se você tirar 2, 3 ou 12 na primeira jogada, isto é chamado de "craps" e você perdeu. Se, na primeira jogada, você fez um 4, 5, 6, 8, 9 ou 10,este é seu "Ponto". Seu objetivo agora é continuar jogando os dados até tirar este número novamente. Você perde, no entanto, se tirar um 7 antes de tirar este Ponto novamente.

import random

def function():

  ponto = 0

  primeira\_jogada = random.randrange(2, 12)

  def jogarDado():

    dado = random.randrange(2, 12)

    return dado

  if primeira\_jogada == 7 or primeira\_jogada == 11:

    print(f"Caiu {primeira\_jogada} na primeira rodada")

    print("Você é um natural\nGanhou, parabéns!!!")

    return

  elif primeira\_jogada == 2 or primeira\_jogada == 3 or primeira\_jogada == 12:

    print(f"Caiu {primeira\_jogada}\nCRAPS!!!\nVocê perdeu")

    return

  else:

    outra\_rod = jogarDado()

    ponto = primeira\_jogada

    jogarDado()

    while outra\_rod != ponto:

      d2 = jogarDado()

      outra\_rod = d2

      print(f"{ponto} é seu ponto da primeira jogada.\n{outra\_rod} É O seu ponto nesta rodada\nSeu objetivo é continuar jogando O dado até tirar o número da primeira jogada novamente")

      if outra\_rod ==7:

        print(f"Você tirou 7 antes de tirar o mesmo número. Você perdeu")

        jogarDado()

        return

function()

11. Data com mês por extenso. Construa uma função que receba uma data no formato DD/MM/AAAA e devolva uma string no formato D de mesPorExtenso de AAAA. Opcionalmente, valide a data e retorne NULL caso a data seja inválida.

12. Embaralha palavra. Construa uma função que receba uma string como parâmetro e devolva outra string com os carateres embaralhados. Por exemplo: se função receber a palavra python, pode retornar npthyo, ophtyn ou qualquer outra combinação possível, de forma aleatória. Padronize em sua função que todos os caracteres serão devolvidos em caixa alta ou caixa baixa, independentemente de como foram digitados.

import random as r

from xml.etree.ElementTree import tostring

letras = []

def embaralhar():

    palavra = str(input("Digite a palavra que deseja embaralhar:"))

    letras = list(palavra)

    r.shuffle(letras)

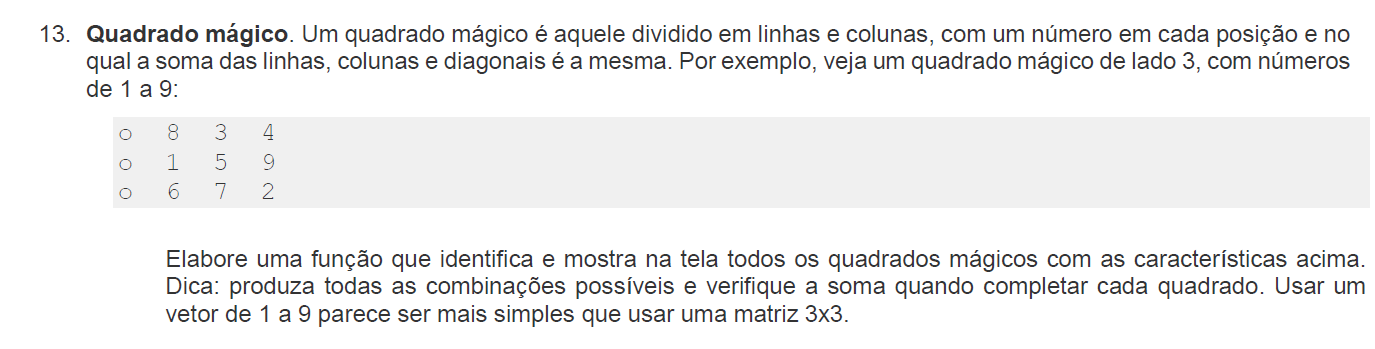
    palavra = ""

    for i in range(letras.\_\_len\_\_()):

        palavra = palavra + f"{letras[i]}"

    print(palavra)

embaralhar()



import random as r

resultado = []

global x

def quadradoPerfeito():

    x = 0

    while x < 1:

        numeros = [1,2,3,4,5,6,7,8,9]

        p\_trinca = []

        s\_trinca = []

        t\_trinca = []

        for i in range(3):

            numero\_p = r.choice(numeros)

            numeros.remove(numero\_p)

            p\_trinca.append(numero\_p)

        for i in range(3):

            numero\_s = r.choice(numeros)

            numeros.remove(numero\_s)

            s\_trinca.append(numero\_s)

        for i in range(3):

            numero\_t = r.choice(numeros)

            numeros.remove(numero\_t)

            t\_trinca.append(numero\_t)

        if(sum(p\_trinca) == sum(s\_trinca) == sum(t\_trinca)):

            if(sum(p\_trinca) == (p\_trinca[0] + s\_trinca[1] + t\_trinca[2])):

                if(sum(p\_trinca) == (p\_trinca[2] + s\_trinca[1] + t\_trinca[0])):

                    if((p\_trinca[1]+ s\_trinca[1]+ t\_trinca[1]) == (p\_trinca[0]+ s\_trinca[0]+ t\_trinca[0]) == (p\_trinca[2]+ s\_trinca[2]+ t\_trinca[2])):

                        resultado.append(p\_trinca)

                        resultado.append(s\_trinca)

                        resultado.append(t\_trinca)

                        x = x + 1

    print("RESULTADO:")

    for i in range(3):

        print(f"{resultado[i]} ")

    print("\n")

    print("SOMA DAS LINHAS")

    print(sum(p\_trinca))

    print(sum(s\_trinca))

    print(sum(t\_trinca))

    print("\n")

    print("SOMA DAS COLUNAS")

    print(p\_trinca[0]+ s\_trinca[0]+ t\_trinca[0])

    print(p\_trinca[2]+ s\_trinca[2]+ t\_trinca[2])

    print(p\_trinca[1]+ s\_trinca[1]+ t\_trinca[1])

    print("\n")

    print("SOMA DAS COLUNAS")

    print(p\_trinca[0] + s\_trinca[1] + t\_trinca[2])

    print(p\_trinca[0] + s\_trinca[1] + t\_trinca[2])

quadradoPerfeito()