Quarta Lista de Exercícios - Computational Thinking

1. Dada uma sequência de números inteiros onde o último elemento é o 0, escreva um algoritmo que calcula a soma dos números pares da sequência.

```
1
2  somaPar = 0
3  num = int(input("Informe num da sequencia"))
4
5  while num != 0:
6     if num % 2 == 0:
7          somaPar = somaPar + num
8     num = int(input("Informe num da sequencia"))
9
10 print("A soma dos pares e {}".format(somaPar))
```

2. Dados o número n de alunos de uma turma de Algoritmos e suas notas da primeira prova, determinar a média das notas dessa turma. Considere que o usuário digite as informações corretamente.

3. Altere o algoritmo anterior para, além da média, contar os alunos que tiraram entre 0 e $5,0$ ($0 \leq nota < 5,0$) e acima de $5,0$ ($nota \geq 5,0$).

```
1  #Correcao dos exercicios 2 e 3
2  soma = 0
3
4  n = int(input("Informe a qtd de alunos"))
5  conta = 1
6
7  contaMenos5 = 0
8  contaMais5 = 0
9
10 while conta <= n:
11    nota = float(input("Informe a nota do aluno"))
12    soma = soma + nota
13    if nota < 5:
14        contaMenos5 = contaMenos5 + 1
15    else:
16        contaMais5 = contaMais5 + 1
17
18 media = soma / n
19 print("A media da turma e {}".format(media))
20 print("{} foi a qtd de alunos que tiraram menos que 5.".format(contaMenos5))
21 print("{} foi a qtd de alunos que tiraram mais ou igual a 5.".format(contaMais5))
```

4. Dados n um inteiro positivo e uma sequência de n números reais, escreva um algoritmo que conta e imprime a quantidade de números positivos e a quantidade de números negativos.

```
1  n = int(input("Informe a qtd de numeros da sequencia: "))
2  conta = 1
3
4  contaPos = 0
5  contaNeg = 0
6
7  while conta <= n:
8     num = float(input("Informe um numero da sequencia: "))
9     if num < 0:
10        contaNeg = contaNeg + 1
11    elif num > 0:
12        contaPos = contaPos + 1
13
14 print("{} foi a qtd de numeros negativos.".format(contaNeg))
15 print("{} foi a qtd de numeros positivos.".format(contaPos))
```

5. Escreva um algoritmo que, dados um número inteiro positivo n, imprime na tela a contagem de todos os divisores positivos de n.

```
n = int(input("Informe numero que deseja contar os divisores: "))
conta = 0

div = 1

while div <= n:
   if n % div == 0:
       conta = conta + 1

print("{} e a qtd de divisores de {}".format(conta, n))
```

6. Em uma prova de concurso com 70 questões haviam 20 pessoas concorrendo. Sabendo que cada questão vale 1 ponto, escreva um algoritmo que lê a pontuação da prova obtida de cada um dos candidatos e calcula:

   a) a maior e a menor nota

   b) o percentual de candidatos que acertaram até 20 questões, o percentual que acertaram de 21 a 50 e o percentual que acertou acima de 50 questões

```
nota = int(input("Cand: "))
maior = nota
menor = nota

conta = 2

contaAte20 = 0
conta20a50 = 0
contaAcima50 = 0

while conta <= 70:
    nota = int(input("Cand: "))
    if nota > maior:
        maior = nota
    if nota < menor:
        menor = nota

    if nota <= 20:
        contaAte20 = contaAte20 + 1
    elif nota <= 50:
        conta20a50 = conta20a50 + 1
    else:
        contaAcima50 = contaAcima50 + 1

    conta = conta + 1

print("A maior nota eh", maior)
print("A menor nota eh", menor)

print("Ate 20 questoes {:.2f}".format(contaAte20 * 100 / 70))
print("Entre 20 e 50 questoes {:.2f}".format(conta20a50 * 100 / 70))
print("Acima de 50 questoes {:.2f}".format(contaAcima50 * 100 / 70))
```

7. A conversão de graus Fahrenheit para centígrados é obtida pela fórmula $C = \frac{5}{9}(F - 32)$. Escreva um algoritmo que calcule e escreva uma tabela de graus centígrados em função de graus Fahrenheit que variem de 50 a 150 Fahrenheit de 1 em 1.

```

fahr = 50
while fahr <= 150:
   celsius = (fahr - 32) * 5 / 9
   print("F: {}  Celsius: {}".format(fahr, celsius))
   fahr = fahr + 1
```

8. Um número inteiro positivo n é denominado primo se existirem apenas dois divisores inteiros positivos dele: o 1 e o próprio n. Escreva um algoritmo que recebe um inteiro n e verifica se n é primo ou não.

```
n = int(input("Digite n: "))

div = 1
contador = 0

while div <= n:
    if n % div == 0:
        contador = contador + 1
    div = div + 1

if contador == 2:
    print(n, " eh primo")
else:
    print(n, " nao eh primo")
```

9. Dados um montante em dinheiro inicial d, uma taxa de juros mensal j e um período de tempo em meses t, escreva um algoritmo que calcula o valor final em dinheiro se d ficar aplicado a taxa de juros j durante t meses.

```
dinheiro = float(input("Montante: "))
juros = float(input("Juros: "))
tempo = int(input("Tempo aplicacao: "))

while t > 0:
   dinheiro = dinheiro * (1 + juros / 100)
   t = t - 1

print("O valor final aplicado e de {:.2f}".format(dinheiro))
```

10. Escreva um algoritmo que recebe um inteiro positivo n e calcula $n! = 1 \cdot 2 \cdot 3 \ldots \cdot (n-1) \cdot n$. Por exemplo, se $n = 6$, então $6! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 = 720$.

```
n = int(input("Informe n: "))

fatorial = 1

while n > 0:
   fatorial = fatorial * n
   n = n - 1

print("O fatorial vale {}".format(fatorial))
```

11. Se F_n é o n-ésimo número da sequência de Fibonacci, podemos calculá-la através da seguinte fórmula de recorrência:

$$F_n = \begin{cases} 1 & \text{se } n = 1 \text{ ou } n = 2; \\ F_{n-1} + F_{n-2} & \text{se } n > 2 \end{cases}$$

Vamos mostrar os 10 primeiros números da sequência de Fibonacci:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55$$

Escreva um algoritmo que dado n, calcula o n-ésimo número da sequência de Fibonacci.

```
n = int(input("Qual numero de Fibonacci voce quer: "))

anterior = 1
atual = 1
```

```
5
6  if n <= 2:
7     print(1)
8  else:
9     i = 3
10    while i <= n:
11       proximo = atual + anterior
12       anterior = atual
13       atual = proximo
14       i = i + 1
15
16    print(atual)
```

12. Dizemos que um número natural n é palíndromo se o 1º algarismo de n é igual ao seu último algarismo, o 2º algarismo de n é igual ao penúltimo algarismo, e assim sucessivamente. Exemplos: 567765 e 32423 são palíndromos. 567675 não é palíndromo.

```
1  num = int(input("Informe o numero: "))
2
3  numOriginal = num
4  resp = 0
5
6  while num != 0:
7      d = num % 10
8      num = num // 10
9      resp = resp * 10 + d
10
11 if numOriginal == resp:
12     print("Eh palindrome")
13 else:
14     print("Nao eh palindrome")
```

13. Vamos escrever um programa que consiste em um Jogo de Adivinhação. O jogo funciona do seguinte modo: sorteia-se um número inteiro aleatório entre 1 e 1000. Sua tarefa é tentar adivinhar o número sorteado através de "chutes". A cada chute o programa deverá informar se o número "sorteado"é maior, menor ou igual ao número "chutado". Quando o usuário acertar o número deverá ser impresso uma mensagem dizendo que ele acertou e a quantidade de chutes dados. Para gerar números aleatórios entre 1 e 1000 use as seguintes instruções dentro do seu programa Python.

```
import random

sorteado = random.randint(1,1001)
```

```
1  import random
2
3  sorteado = random.randint(1, 1001)
4  #print(sorteado)
5  tentativas = 1
6  chute = int(input("Tentativa: "))
7
8  while chute != sorteado:
9      if chute < sorteado:
10         print("Tente um numero maior!")
11     elif chute > sorteado:
12         print("Tente um numero menor!")
13     chute = int(input("Tentativa: "))
14     tentativas = tentativas + 1
15
16 print("Parabens, voce acertou!")
17 print(tentativas, " chutes")
```

14. Dizemos que um inteiro positivo n é perfeito se for igual à soma de seus divisores positivos diferentes de n.

    Exemplo:

    6 é perfeito, pois $1 + 2 + 3 = 6$.

    Sua tarefa será a de escrever um algoritmo em Python que, dado um inteiro positivo n, verificar se n é perfeito.

```
1
2  soma = 0
3  divisor = 1
4
5  num = int(input("Informe o numero: "))
6
7  while divisor < num:
8      if num % divisor == 0:
9          soma = soma + divisor
10     divisor = divisor + 1
11
12 if soma == num:
13     print(num, 'e perfeito')
14 else:
15     print(num, 'nao e perfeito')
```

15. Dados um inteiro n e uma seqüência de n números inteiros, determinar o comprimento de um segmento crescente de comprimento máximo.

    Exemplos:

    Na seqüência 5, 10, 3, 2, 4, 7, 9, 8, 5 o comprimento do segmento crescente máximo é 4.

    Na seqüência 10, 8, 7, 5, 2 o comprimento de um segmento crescente máximo é 1.

```
1  n = int(input("Informe tamanho da sequencia: "))
2
3  #leio o primeiro numero da sequencia fora do while
4  #estou supondo que n > 0, ou seja, existe pelo menos 1 numero na sequencia
5
6  num = int(input("Num: "))
7
8  maximo = 1
9  contadorMaximo = 1
10 contadorSequencia = 1
11 anterior = num
12
13 while contadorSequencia < n:
14     num = int(input("Num: "))
15     if num > anterior:
16         contadorMaximo = contadorMaximo + 1
17     else:
18         if maximo < contadorMaximo:
19             maximo = contadorMaximo
20         contadorMaximo = 1
21     anterior = num
22     contadorSequencia = contadorSequencia + 1
23
24 print("O segmento crescente maximo tem tamanho", maximo)
```

16. Uma das maneiras de evitar erros na digitação de números como conta corrente, CPF, boleto bancário é a utilização de um ou mais dígitos de controle. Um dos métodos de cálculo é a utilização do método módulo 10. Segue a descrição do algoritmo: Dado um número inteiro n devemos pegar cada dígito desse número começando pela casa das unidades e multiplicar, alternadamente, por 2 e por 1. Caso o resultado da multiplicação seja um

número maior ou igual a 10 devemos simplificar esse valor somando os dois dígitos. Após feitas as multiplicações e as simplificações devemos somar todos os valores e calcular o resto da divisão dessa soma por 10. Se o resto for 0 o dígito de controle é zero, caso contrário o dígito de controle será 10 menos o resto.

A Figura 1 pode servir de exemplo para o algoritmo móduto 10.

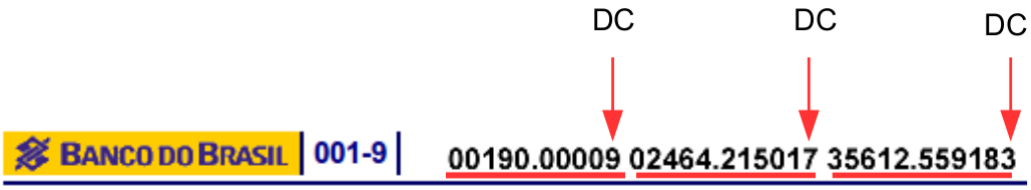

Figura 1: Trecho de boleto bancário do Banco do Brasil

Por exemplo vamos pegar o número do meio do boleto, o número corresponde a $n = 246421501$ e o dígito de controle será 7. Vamos efetuar os seguintes cálculos:

$$\begin{aligned} 1 * 2 &= 2 \\ 0 * 1 &= 0 \\ 5 * 2 &= 10 \\ 1 * 1 &= 1 \\ 2 * 2 &= 4 \\ 4 * 1 &= 4 \\ 6 * 2 &= 12 \\ 4 * 1 &= 4 \\ 2 * 2 &= 4 \end{aligned}$$

Daí somamos $2 + 0 + 1(1 + 0) + 1 + 4 + 4 + 3(1 + 2) + 4 + 4 = 23$. O resto da divisão de 23 por 10 é 3 e como ele é diferente de zero o dígito de controle de 246421501 será 7 $(10 - 3)$.

Escreva um algoritmo que lê um número inteiro positivo e calcula o seu dígito de controle usando o método do módulo 10.

```
numero = 2464215017

dcDigitado = numero % 10
numero = numero // 10
soma = 0
fator = 2
while numero != 0:
    dig = numero % 10
    dig = dig * fator
    valor = (dig // 10) + (dig % 10)
    soma = soma + valor
    numero = numero // 10
    if fator == 2:
        fator = 1
    else:
        fator = 2
```

```
17
18 resto = soma % 10
19 dcCalculado = 0
20 if resto > 0:
21     dcCalculado = 10 - resto
22
23 if dcDigitado == dcCalculado:
24     print("Parabens, tudo certo!")
25 else:
26     print("Algo deu errado!")
```

17. Em quase todas as linguagens existe um método/função que calcula a raiz quadrada de um número. Contudo, vamos supor que na linguagem Python não exista tal método. Sua tarefa é tentar encontrar a raiz quadrada de um número por tentativa e erro, ou seja, seu algoritmo recebe um número real $n > 0$ e tenta encontrar sua raiz quadrada. Dica: pegue um número e eleve ao quadrado, caso ele fique "próximo"ao número n, significa que você encontrou a raiz quadrada.

```
1 num = float(input("Informe o numero: "))
2
3 aproximacao = 0
4
5 while aproximacao * aproximacao < num:
6     aproximacao = aproximacao + 0.001
7
8 print("A raiz quadrada de ", num, "e", aproximacao)
```

Boa sorte!