Universidade de Brasília
Departamento de Ciência da Computação

Introdução à Ciência da Computação - 113913

Gabarito da Lista 4

Funções Frutíferas

Observações:

- As listas de exercícios serão corrigidas por um **corretor automático,** portanto é necessário que as entradas e saídas do seu programa estejam conforme o padrão especificado em cada questão (exemplo de entrada e saída). Por exemplo, a não ser que seja pedido na questão, não use mensagens escritas durante o desenvolvimento do seu código como "Informe a primeira entrada". Estas mensagens não são tratadas pelo corretor, portanto a correção irá resultar em resposta errada, mesmo que seu código esteja correto.
- As Instâncias de Entrada serão as usadas pelo corretor e suas saídas devem estar **iguais** às apresentadas em Instâncias de Saída.

Questão A.

```
def compare(x, y):
    if(x > y):
        return 1
    elif(x == y):
        return 0
    return -1

x = int(input())
y = int(input())
retorno = compare(x, y)
if(retorno == 1):
    print("x e maior que y")
elif(retorno == 0):
    print("x e igual a y")
else:
    print("x e menor que y")
```

Instâncias de Entrada	Instâncias de Saída
2 2	x e igual a y
0 0	x e igual a y
5 4	x e maior que y
-5 -2	x e menor que y
-1 -1	x e igual a y
100 99	x e maior que y
99 100	x e menor que y
49 490	x e menor que y
-1500 -2000	x e maior que y
-10 -10	x e igual a y

Questão B.

```
def entrada_dados(maior, numero):
    if(numero > maior):
        maior = numero
    return maior

numero = int(input())
maior = numero
x = numero
#Atribuimos maior como o primeiro numero lido e chamamos a função 9 vezes
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
maior = entrada_dados(maior, int(input()))
print(maior)
if(maior % x == 0):
    print(x)
```

Instâncias de Entrada	Instâncias de Saída
4 1 2 3 4 5 6 7 8 80	80 4
8 32 1 2 3 4 5 6 7 16	32 8
1 2 3 4 5 6 7 8	10 1

9 10	
-1 -2 -3 -4 -5 -6 -7 -8 -9 -10	-1 -1
-2 -4 -8 -16 -32 -64 -128 -256 -512 -1024	-2 -2
-2 -4 -5 -6 -7 -8 -9 -10 -11 -1	-1
5 0 0 0 0 0 0 0 0 0	5 5
4 4 4 4 4	4 4

4 4 4 4 4	
-10 5 4 3 100 2 3 4 -100 20	100 -10
-2 -40 -1 -80 -60 -50 -40 -30 -20 -10	-1

Questão C.

```
""" Se o número for par imprimimos na tela e chamamos a função recursivamente,
passando como parâmetro n-2(o próximo número par menor que o atual n). Caso o
número passado inicialmente como parâmetro seja ímpar, então apenas chamamos a
função de forma recursiva, passando n-1 como parâmetro. Quando n for negativo,
finalizamos a função """
def imprime_pares(n):
    if(n >= 2):
        if(n%2 == 0):
            print(n)
            imprime_pares(n-2)
        else:
            imprime_pares(n-1)
    else:
        return

n = int(input())
imprime_pares(n)
```

Instâncias de Entrada	Instâncias de Saída
0	
12	12 10 8 6 4 2
18	18 16 14 12 10 8 6 4 2
11	10 8 6 4 2
9	8 6 4 2
7	6 4 2
5	4 2
4	4

	2
3	2
1	

Questão D.

```
""" Com raciocínio semelhante à questão 3, se o número digitado for par então
retornaremos a soma de n-2 e chamamos a função novamente passando n-2. Caso o
número digitado for ímpar, retornaremos o próximo par p que seja igual a n - 2e
chamamos a função passando p.
Note que depois que verificarmos se o n digitado pelo usuário é par, todas as
chamadas da função terão como parâmetro um número par.
Continuaremos essas chamadas até que n seja 0, onde terminamos o somatório """
def somatorio(n):
    if(n > 0):
        if(n % 2 == 0): #Caso n seja par
            return (n-2) + somatorio(n-2)
        else: #Caso seja ímpar
            return (n-3) + somatorio(n-3)
    else:
        return 0

n = int(input())
if(n < 0):
    print("-1")
else:
    print(somatorio(n))
```

Instâncias de Entrada	Instâncias de Saída
-2	-1
0	0
-4	-1
21	90
30	210
14	42
25	132
10	20
100	2450
18	72

Questão E.

```
""" Semelhante as questões 3 e 4, porém na função quadrado_pares quando temos
todos os quadrados dos pares calculados e chegamos ao valor 1 nós chamamos
a função entrada novamente para ler o próximo valor """

def entrada():
    n = int(input())
    if(n == 0): #Se n for 0 então paramos de ler valores do teclado
        return
    else:
        quadrado_pares(n)

def quadrado_pares(n):
    if(n > 1):
        if(n % 2 == 0):
            print("%d^2 = %d"%(n,n**2))
            quadrado_pares(n-2)
        else:
            quadrado_pares(n-1)
    else: #Aqui precisamos ler o próximo valor
        entrada()

entrada()
```

Instâncias de Entrada	Instâncias de Saída
8 4 3 0	8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4 4^2 = 16 2^2 = 4 2^2 = 4
0	
9 0	8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4
15 3 1 2 0	14^2 = 196 12^2 = 144 10^2 = 100 8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4 2^2 = 4 2^2 = 4
1 1 1 4 5 0	4^2 = 16 2^2 = 4 4^2 = 16 2^2 = 4

3 9 27 1 0	2^2 = 4 8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4 26^2 = 676 24^2 = 576 22^2 = 484 20^2 = 400 18^2 = 324 16^2 = 256 14^2 = 196 12^2 = 144 10^2 = 100 8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4
4 1 17 0	4^2 = 16 2^2 = 4 16^2 = 256 14^2 = 196 12^2 = 144 10^2 = 100 8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4
5 4 3 2 1 0	4^2 = 16 2^2 = 4 4^2 = 16 2^2 = 4 2^2 = 4 2^2 = 4
1 2 4 8 1 0	2^2 = 4 4^2 = 16 2^2 = 4 8^2 = 64 6^2 = 36 4^2 = 16 2^2 = 4
2 3 5 5 1 0	2^2 = 4 2^2 = 4 4^2 = 16 2^2 = 4 4^2 = 16 2^2 = 4

Questão F.

```
def mdc(a, b):
    if(b == 0):
        return a
    else:
        return mdc(b, a%b)

def mmc(a,b):
    if(a == 0 or b == 0):
        return 0
    else:
        minimo_multiplo = (a*b)//mdc(a,b)
    return minimo_multiplo
#O mmc(a,b) * mdc(a,b) = a*b

def entrada():#Vamos ler valores enquanto os dois forem maiores ou iguais a zero
    num1, num2 = input().split()
    num1, num2= [int(num1), int(num2)]
    if(num1 < 0 or num2 < 0):
        return
    else:#Escrevemos na tela o mmc e lemos os próximos números
        print(mmc(num1, num2))
        entrada()

entrada()
```

Instâncias de Entrada	Instâncias de Saída
8 16 4 10 3 5 -3 29	16 20 15
0 5 5 0 4 8 -1 -1	0 0 8
1 1 3 7 22 11 10 25 -4 25	1 21 22 50
4 2 3 -3	4
10 10 0 4 7 147 2 25 900 800 -3543 1	10 0 147 50 7200
-1000 0	
2 2 15 44 15 45 1001 0	2 660 45 0

-5 -1001	
4 240 3 2016 6 -1	240 2016
2017 2016 2 1 -1 0	4066272 2
20 24 30 35 0 40 800 650 397 311 5 0 4 1 100 23 -15 9	120 210 0 10400 123467 0 4 2300

Questão G.

```
def concat(a, b):
    if not a:
        return b
    else:
        return a[0:1] + concat(a[1:], b)
def prefix(a, b):
    if (not a) and b:
        return True
    elif (a and b):
        return prefix(a[1:], b[1:])
    else:
        return False
def rev(l):
    if not l:
        return l
    else:
        return concat(rev(l[1:]), l[0:1])

a = input()
b = input()
print(concat(a, b))
print(rev(a))
print(prefix(a, b))
```

Amostras de Entrada	**Amostras de Saída**
raphael	raphael leahpar False
jon jonerys	jonjonerys noj True
seila	seila alies False
 raphael	raphael True
valarmorghulis valardohaeris	valarmorghulisvalardohaeris siluhgromralav False
valar valarh	valarvalarh ralav True
aaaa bbbb	aaaabbbb aaaa False
arara ararab	araraararab arara True
xh hx	xhhx hx

	False
oie	oie eio False