


José Maria Cardoso 99096 

Pergunta 1

def num_cabos_maiores(l, m):
return len(filter(reduce(l , lambda x, y : $x + y$), l), lambda x : $x > m$))



José Maria Cardoso 99096 Jm

Pergunta 2



```
c) def soma_mos_primos(m):  
    ns=0  
    i=1  
    while i <= m:  
        if i not eh_primo(i):  
            ns += i  
        i += 1  
    return ns
```

```
b) def soma_mos_primos(m, i=1, ns=0):  
    if i > m:  
        return ns  
    else:  
        if i not eh_primo(i):  
            ns = soma_mos_primos(m, i+1, ns+i)  
        return soma_mos_primos(m, i+1, ns)
```

```
a) def soma_mos_primos(m):  
    def auxilior(m, i):  
        if i not eh_primo(i):  
        return  
        if i > m:  
            return 0  
        if not eh_primo(i):  
            return i + auxilior(m, i+1)  
        return auxilior(m, i+1)  
  
    return auxilior(m, 1)
```

José Maria Cardoso 99096 Ju

Pergunta 3

a) V

b) F

c) V

d) V

e) F

f) F



José Maria Cardoso 99096 Lu

Pergunta 4

def omigos(t, t2):

if type(t) != str or type(t2) != str or len(t) != len(t2):

raise ValueError('argumentos inválidos')

def conta_diferencas(t, t2):

neg = 0
for i in range(len(t)):

if t[i] != t2[i]:

neg += 1

return neg

if conta_diferencas(t, t2) < len(t) / 10:

return True

return False



Pergunta 5

c) Com a implementação do `push` de `data` abaixo, escreva a implementação da `pop` de `data`.

e) # Construção

```
def novo_filha_filho():  
    return []
```

Seleção

```
def primeira_filha():  
    return filhas[0]
```

```
def ultima_filha():  
    return filhas[-1]
```

```
def comprimento_filha():  
    return len(filha)
```

Modificação

```
def entre_inicio_filha(filha, m):  
    filha.insert(0, m)
```

return filha

```
def push_entre_filha(filha, m):  
    filha.append(m)  
    return filha
```

```
def sai_inicio_filha(filha):  
    filha.pop(0)  
    return filha
```

```
def sai_fim_filha(filha):  
    filha.pop(-1)  
    return filha
```



Pergunta 5

c) Com a implementação os ficheiros de dados armazenados virão a ser

e) # Construtores

```
def novo_ficheiro_dados():  
    return []
```

```
# Seletores
```

```
def primeiro(ficheiro):  
    return ficheiro[0]
```

```
def ultimo(ficheiro):  
    return ficheiro[-1]
```

```
def comprimento(ficheiro):  
    return len(ficheiro)
```

```
# Modificadores
```

```
def entra_inicio_ficheiro(ficheiro, m):  
    ficheiro.insert(0, m)
```

```
    return ficheiro
```

```
def fica_entrada_fim_ficheiro(ficheiro, m):  
    ficheiro.append(m)  
    return ficheiro
```

```
def sai_inicio_ficheiro(ficheiro):  
    ficheiro.pop(0)  
    return ficheiro
```

```
def sai_fim_ficheiro(ficheiro):  
    ficheiro.pop(-1)  
    return ficheiro
```



José Maria Cardoso 99096 (Ju)

Pergunta 5 - Continuação

Recompreções

def eh_fila_duplo(orig):

if type(orig) == int

return True

return False

def eh_fila_vazia(orig):

~~if type(orig) == int~~

~~return True~~

return len(orig) == 0

c)

def junta_filas_duplos(fl1, fl2):

fl2 = fl2[-1]

for i in range(len(fl2)):

fl1.append(fl2[i])

return fl1



Jose Mario Cardoso 99096 (ju)



Pergunta 6

```
def conta_duplicados(e):
```

```
    def verificacao(e,i):
```

```
        if type(i) != int:
```

```
            raise ValueError('argumentos inválidos')
```

```
        return
```

```
        if i == len(e)-1:
```

```
            return True
```

```
    else:
```

```
        if type(e[i]) != int:
```

```
            return False
```

```
            raise ValueError('argumentos inválidos')
```

```
        return verificacao(e, i+1)
```

```
def
```

```
def contagem(e,m,i):
```

```
    if i == len(e)-1:
```

```
        return 0
```

```
    if e[i] == m:
```

```
        return 1 + contagem(e,m,i+1)
```

```
    return contagem(e,m,i+1)
```

```
def novo_lista(e, newE, i):
```

```
    * x = contagem(e, e[i], 0)
```

```
    if x > 1:
```

```
        raise ValueError
```

```
        newE = newE + [e[i]]
```

```
        return novo_lista(e, newE + [e[i]], i+1)
```

```
    return novo_lista(e, newE, i+1)
```

```
return
```

```
if verificacao(e, 0)
```

```
    return novo_lista(e, [], 0)
```

```
else:
```

```
    raise ValueError('argumentos inválidos')
```

```
* if i == len(e)
    return newE
```


Jose Maria Cardoso 99096 fu

Pergunta 7

import random

class slot_machine():

def __init__(self, m, m2):

self.c_Morrima = m2

self.c_Dezaguel = m

self.c = m

def jogar (self):

~~self.c = m~~

ans =

if self.c < self.c_Dezaguel or self.c == self.c_Morrima:

return 'Slot Machine não operacional'

self.c += 1

m = random.randint(1,50)

m2 = random.randint(1,50)

m3 = random.randint(1,50)

ns = 0

if m in (m2, m3):

if m == m2 and m == m3:

ns += 100

else:

ns += 10

elif 2 m2 and m2 == m3

~~if m2 == m3 and m2 == m3~~

ns += 10

if ns == 100 and m == 7:

ns = self.c

if ns > 0:

return 'Resultado: (ns, m, m2, m3) em Sorte'. format(m, m2, m3)

return 'Resultado: (ns, m, m2, m3) em Azar'. format(m, m2, m3)



José Maria Cardoso 99096 (ju)

Pergunta 7 - Continuação

def estado (self):

if self.c < self.cDesejavel or self.c == self.cMaxima:

~~return~~

return 'Agência Montemoreau em Depsito com {} Euros em Copoidade Máxima

~~{}~~ ~~Euros~~ ~~format~~

em Copoidade Desejavel {} Euros em Copoidade Máxima {} Euros ~~format~~

.format (self.c, self.cDesejavel, self.cMaxima)

return 'Operacional em Depsito com {} Euros em Copoidade Desejavel {} Euros em

Copoidade Máxima {} Euros'.format (self.c, self.cDesejavel, self.cMaxima)

def abatece (self, m):

~~self.c += m~~

if type(m) != int or m < 0:

raise ValueError ('argumento invalido')

self.c += m



José Maria Cardoso 99096 Jm

Pergunta 8

def primoioe(m):

~~def primoioe(m):~~

~~if m == 1:~~

~~if m == 2:~~

~~if m % 2 == 0 and m > 2:~~

~~return True~~

~~return False~~

~~for i in range(2, m):~~
~~if m % i == 0:~~
~~return False~~
~~return True~~

res = 1

if

for i in range(1, m+1):

if primoioe eh_primo(i)

res *= i

return res

def eh_primo(m):

e = []

~~for i in range~~

while i <= m:

if m % i == 0:

e.append(i)

if e == [1, m]:

return True

return False



José Mario Cardoso 99096 JW

Pergunta 9

```
def soma_expres (e1, e2):
```

```
    da
```

```
    e3 = dict(e1, **e2)
```

```
    for k in e1:
```

```
        if k in e3:
```

```
            for i in e1[k]:
```

```
                e3[k] += v
```

```
    return e3
```



José Maria Cardoso 99096 Ju

Pergunta 10

a) $\langle \text{expression} \rangle ::= n \mid \langle \text{random} \rangle c$

$\langle \text{random} \rangle ::= a \mid a \langle \text{random} \rangle \mid d \mid d \langle \text{random} \rangle$

b)

def predicado(t):

if type(t) != str:

raise ValueError('argumento inválido')

else: n = 0

if t[0] == 'n' and t[-1] == 'c':

n += 2

for i in range(1, len(t)-1):

if t[i] in ('a', 'd'):

n += 1

if n == len(t):

return True

return False



José Maria Cardoso 99096 *JM*

Pergunta 11

def soma_cumulativo(e):

~~new~~
newe=[]

for i in range(len(e)):

new = 0

for a in range(i+1):

new += e[a]

newe.append(new)

return newe

