Segundo control de programación en Python

Escribe un programa en Python que realice as seguintes operacións:

1. Lea por teclado valores enteiros positivos e almacéneos no vector **v**. Remata de ler cando o usuario introduza un valor negativo (non incluido no vector), ou o usurario introduza o décimo valor.

2. Sexa n o número de valores introducidos, define unha función co nome `matriz(...)`, cos argumentos axeitados, que calcule a matriz **a** de orde $n \times n$, onde os elementos da matriz a_{ij} veñen dados por:

$$a_{ij} = \sum_{k=0}^{n-1} v_{\text{mín}(i+k,n-1)} v_{\text{máx}(j-k,0)}, \quad i,j = 0,\ldots,n-1 \tag{1}$$

   onde: $\text{mín}(x,y) = x$ se $x \leq y$ e $\text{mín}(x,y) = y$ en caso contrario; $\text{máx}(x,y) = x$ se $x \geq y$ e $\text{máx}(x,y) = y$ en caso contrario.

3. O programa principal ten que chamar a función `matriz(...)` e representar nunha figura (con dous gráficos na mesma fila) os seguintes gráficos: un mapa de calor da matriz transposta de **a** e un histograma dos valores da matriz **a**. Pon un título a cada gráfico.

4. Finalmente, o programa principal debe almacenar no arquivo `saida.txt` os seguintes datos:

   - O vector **v** e a matriz **a**.
   - O número de valores múltiplos de 4 na matriz **a**.
   - A suma dos elementos da matriz **a** que son divisibles por 3.

Se introduces os números $1, 2, 3, -1$, tes que obter o arquivo `saida.txt` seguinte:

```
v=     1    2    3
a=
    6    7   10
    8   10   15
    9   12   18
Numero multiplos de 4 = 2
Suma numeros divisibles 3 = 95
```

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
from numpy import *
from matplotlib.pyplot import *
v=[]
for i in range(10):
    x=int(raw_input('Introduce numero positivo: '))
    if x<0:
        break
    v.append(x)
n=len(v)
def matriz(x):
    n=len(x)
    b=zeros([n,n], dtype='int')
    for i in range(n):
        for j in range(n):
            b[i,j]=0
            for k in range(n):
                b[i,j]=b[i,j] + x[min(i+k,n-1)]*x[max(j-k,0)]
```

```
    return b
a=matriz(v)
clf(); subplot(121)
imshow(a.T); title('Matriz a transposta')
subplot(122)
hist(a.flatten()); title('Histograma de a')
show(False)
try:
    outf=open("saida.txt", "w")
    outf.write('%s' % 'v= ')
    for i in range(n):
        outf.write('%5d' % v[i])
    outf.write('\n%s\n' % 'a= ')
    for i in range(n):
        for j in range(n):
            outf.write('%5d' % a[i, j])
        outf.write('\n')
    m4=len(where(a%4 ==0)[0])
    sm3=sum(extract(a > 5,a))
    outf.write('Numero multiplos de 4 = %d\n' % m4)
    outf.write('Suma numeros divisibles 3 = %d\n' % sm3)
    outf.close()
except IOError:
    print "Erro escribindo en arquivo saida.txt"
```

Segundo control de programación en Python

Escribe un programa en Python chamado `exame2.py` que:

1. Lea un número enteiro positivo n por teclado e xere duas matrices **a** e **b** de dimensión $n \times n$ con números aleatorios no intervalo [10,20]. Visualiza ambas matrices na pantalla.
2. Define unha función chamada `calculaVector(...)` que, dada unha matriz, calcule un vector que contén os elementos da matriz (percorrendo a matriz por columnas) mentres que a súa suma sexa inferior a suma dos elementos da matriz dividido por 3.
3. Chama a función `calculaVector(...)` para calcular os vectores **v** e **w**, obtidos a partir da matriz **a** e **b** respectivamente.
4. Sea n e m as dimensións do vectores **v** e **w** respectivamente (en xeral $n \neq m$). Engade a mediana do vector de menor dimensión repetidamente, ate que ambos vectores sexan da mesma dimensión. Visualiza ambos vectores na pantalla.
5. Representa nun gráfico os valores do vector **v** en azul e os do vector **w** en verde. Marca en vermello os valores de **v** que son maiores ós valores de **w** na mesma posición. Por lenda ó gráfico, enreixado e fixa o rango de representación do eixo `y` ó intervalo [10,20] e o do eixo `x` que varie entre 1 e o tamaño dos vectores . Garda o gráfico no arquivo `figura.png`.

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
from numpy import *
from matplotlib.pyplot import *
from numpy.random import *

n=-1
while n<1:
    n=int(raw_input('Introduce n:'))
a=10+10*rand(n,n)
print 'a=', a
b=10+10*rand(n,n)
print 'b=', b
#####################################
def calculaVector(c):
    sc=sum(sum(c))/3
    m=size(c,1)
    s=0; x=[]
    for i in range(m):
        for j in range(m):
            s = s+ c[j,i]
            if s < sc:
                x.append(c[j,i])
            else:
                break
        if s > sc:
            break
    return x
#########################################
v=calculaVector(a)
w=calculaVector(b)
nv=len(v); nw=len(w); n=max(nv,nw)
if nv > nw:
```

```
    for i in range(nw, n):
        w.append(median(w))
elif nw > nv:
    for i in range(nv, n):
        v.append(median(v))
print 'v=', v
print 'w=', w
clf();
x=arange(1,n+1)
plot(x, v, 'b*', label=u'Vector v')
plot(x, w, 'go', label=u'Vector w')
v2=array(v); w2=array(w)
p=where(v2 >w2)[0]
plot(p+1, v2[p], 'ro', label=u'v(i) > w(i)')
legend(loc='upper right')
xlim(min(x), max(x))
ylim(10, 20)
grid(True); show(False)
savefig('figura.png')
```

Segundo control de programación en Python

Escribe un programa en Python chamado `exame3.py` que:

1. Lee un vector **v** de números enteiros por teclado. Sea n a dimensión do vector **v**.
2. Define unha función, chamada `reducir(...)`, cos argumentos axeitados que transforme o vector **v** de modo que:

$$v_i = \begin{cases} abs(v_i + v_{n-i-1}) & 0 \leq i < m \\ abs(v_{i-m} + v_i) & m \leq i < n \end{cases} \tag{2}$$

   onde $m = \lfloor n/2 \rfloor$ e *abs()* é a función valor absoluto.
3. A chamada ao función repetirase ata que a suma dos elementos do vector **v** sexa maior que 1000. En cada iteración, o programa debe engadir o vector **v** calculado ao final do arquivo `saida.txt` (o vector **v** debe ir nunha liña distinta en cada iteración), e calcular o número de iteracións (visualiza este dato na pantalla).
4. Crea e mostra por pantalla a matriz **a** de orde n cos elementos dados pola expresión:

$$a_{ij} = \begin{cases} v_i & i = j \\ v_i v_j & i > j \\ v_i + v_j & i < j \end{cases} \tag{3}$$

5. Finalmente, representa nun gráfico de barras o valor máximo para cada fila da matriz **a**, poñendo as etiquetas *Filas* e *Valor máximo* nos eixos x e y respectivamente.
6. Proba con $n = 5$ e **v**=(1,2,3,4,5). Tes que obter o seguinte arquivo `saida.txt`.

```
    1     2     3     4     5
    6     6     9    10    14
   20    16    29    26    43
   63    42    92    68   135
  198   110   290   178   425
```

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
from numpy import *
from matplotlib.pyplot import *
from sys import *

v=array(input('Introduce vector: '), dtype='int')
n=len(v)
################################
def reducir(w):
    n=len(w); m=n/2
    for i in range(m):
        w[i]=abs(w[i]+w[n-i-1])
    for i in range(m,n):
        w[i]=abs(w[i-m]+w[i])
    return w
################################
out=v.copy()
m=0
```

```
while sum(v)<1000:
    v=reducir(v)
    out=vstack([out, v])
    m = m+1
try:
    savetxt("saida.txt", out, fmt='%5d')
except IOError:
    exit('Erro escribindo no arquivo saida.txt')
[nf, nc]=out.shape
print 'Numero de repeticions: ', nf-1
a=zeros([n,n])
for i in range(n):
    for j in range(n):
        if i == j:
            a[i,j]=v[i]
        elif i > j:
            a[i,j]=v[i]*v[j]
        else:
            a[i,j]=v[i]+v[j]
print 'Matriz a: ', a
clf();bar(range(n), amax(a,1))
xlabel('Filas'); ylabel('Valor maximo')
show(False)
```

Segundo control de programación en Python

Co editor de texto crea un arquivo que conteña na primeira liña un vector de números e nas seguintes liñas unha matriz (polo tanto, todas as liñas terán o mesmo número de elementos). Podes probar co seguinte arquivo:

```
1 2 3 4 5
9 4 2
1 2 7
```

Escribe un programa en Python chamado `exame4.py` que:

1. Pida ó usuario o nome dun arquivo (do tipo descrito) e lea a primeira liña do arquivo a un vector **v**, e as liñas restantes a unha matriz **a**.

2. Define unha función `porcentaxe(...)`, cos argumentos axeitados, que sume os elementos do vector **v** ata que a súa suma sexa superior a 7 ou o vector **v** non teña mais elementos. Chamando m ao número de elementos sumados, a función debe calcular a porcentaxe de elementos da matriz **a** maiores que m.

3. O programa ten que chamar a función `porcentaxe(...)` e mostrar na pantalla a porcentaxe con 6 díxitos e 2 cifras decimais. Co arquivo anterior, debes obter unha porcentaxe do 33.33 %.

4. Convirte a matriz **a** nun vector **y** por columnas (primeiro a primeira columna, logo a segunda, etc.). Sea n a dimensión do vector **y**, crea un vector **x** con valores $x_i = i,\ i = 0, \ldots, n-1$. Representa nun gráfico os puntos $(x_i, y_i),\ i = 0, \ldots, n-1$ con asteriscos azuis e a lenda *Puntos reais*. Calcula os valores interpolados linealmente de 20 puntos equidistantes no rango $[minimo(x), maximo(x)]$ e representaos no mesmo gráfico con círculos verdes e a lenda *Puntos interpolados*. Pon enreixado ó gráfico.

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
from numpy import *
from matplotlib.pyplot import *
from sys import exit
arquivo=raw_input('Nome do arquivo: ')
try:
    f=open(arquivo, 'r')
    l=f.readline()
    aux=l.rsplit()
    v=array(aux, dtype='float')
    l=f.readline()
    a=array(l.rsplit(), dtype='float')
    for l in f:
        a=vstack([a,array(l.rsplit(),dtype='float')])
except IOError:
    print 'Erro lendo arquivo entrada.txt'
    exit()
########################################
def porcentaxe(w, b):
    n=len(w); suma=0
    for i in range(n):
        suma = suma + w[i]
        if suma > 7:
            m=i+1
            break
```

```
    if i == n-1:
        m=n
    else:
        m=i+1
    (nf, nc)=a.shape
    nm=sum(sum(b > m))
    return float(nm)*100/(nf*nc)
########################################
p=porcentaxe(v,a)
print 'Porcentaxe= %6.2f' % p
y=ravel(a.T)
n=len(y)
x=arange(0,n,1)
clf(); plot(x,y,'b*', label='Ptos reais')
xi=linspace(min(x), max(x),20)
yi=interp(xi, x, y)
plot(xi, yi, 'go', label='Ptos interpolados')
grid(True); legend(loc='upper right')
show(False)
```

Segundo control de programación en Python

Escribe un programa en Python chamado `exame5.py` que:

1. Lee un núumero enteiro positivo n polo teclado.
2. Lea números enteiros por teclado e almacenaos no vector $\mathbf{x}$ de dimensión n só se son números positivos e xa non están no vector $\mathbf{x}$.
3. Define unha función, chamada `transforma(...)`, cos argumentos axeitados, que transforme un vector $\mathbf{v}$ noutro vector $\mathbf{w}$ da mesma dimensión onde:

$$w_i = v_i + v_{i-1} \quad i = 1, \ldots, n-1 \quad w_0 = 5 \tag{4}$$

   sendo n a dimensión do vector $\mathbf{v}$.
4. Chama a función `transforma()` para transformar o vector $\mathbf{x}$ noutro vector $\mathbf{y}$. Garda no arquivo `vectores.txt` ambos vectores, un en cada liña, como números enteiros que ocupen 4 espacios cada un.
5. Ordea os vectores $\mathbf{x}$ e $\mathbf{y}$ en orde crecente, calcula o valor da superficie:

$$z = \sin(x_i y_j) \quad i, j = 0, 1, \ldots, n-1 \tag{5}$$

   e representaa nun gráfico en 3D, poñendo etiquetas ós eixos.
6. Proba con $n = 4$ e introducindo os números -5, 1, 2, 1, -2, 3, 4, 8. Tes que obter o seguinte arquivo `vectores.txt` :

```
   1    2    3    4
   5    3    5    7
```

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
from numpy import *
from matplotlib.pyplot import *
from mpl_toolkits.mplot3d import Axes3D

n=int(raw_input('Numero de elementos: '))
i=0; x=[]
while len(x) < n:
    m=int(raw_input('Introduce numero: '))
    if m > 0 and m not in x:
        x.append(m)
#######################################
def transforma(v):
    m=len(x)
    w=ones(m)*5
    for i in range(1,m):
        w[i]=v[i]+v[i-1]
    return w
#################################
y=transforma(x)
x=array(x, dtype='float')
z=x.copy()
z=vstack((z,y))
savetxt('vectores.txt', z, '%4d ')
```

```
x=sort(x)
y=array(y); y=sort(y)
fig = figure()
ax = Axes3D(fig)
X,Y=meshgrid(x,y)
Z=sin(X*Y)
ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap='hot')
ax.set_xlabel('x'); ax.set_ylabel('y')
ax.set_zlabel('Z')
show(False)
```