Primeiro control de programación en Python de 2019

Escribe un programa en Python que faga o seguinte:

1. Lea do teclado un número n que sexa par e maior que 10, comprobando que o número lido sexa correcto.
2. Define un vector **v** de dimensión n con valores enteiros aleatorios no intervalo [0,n].
3. Define dous vectores **x** e **y** de dimensión $m = n/2$ co seguinte contido: o vector **x** contén os valores de **v** situados nas posicións pares e o vector **y** contén os valores de **v** nas posicións impares.
4. Calcula unha matriz **a** de orde m con elementos $a_{ij} = x_i y_j$, para $i, j = 1, \ldots, m$.
5. Calcula unha matriz **b** de orde m onde:

$$b_{ij} = \begin{cases} 1 & \text{se } (i+i)(j+1) \text{ é múltiplo de 2} \\ 2 & \text{se } (i+i)(j+1) \text{ é múltiplo de 3} \\ 3 & \text{se } (i+i)(j+1) \text{ é múltiplo de 2 e 3} \\ 4 & \text{resto dos casos} \end{cases}$$

```
#!/usr/bin/python3
from numpy import *
from numpy.random import *

n=0
while n<10 or n%2:
    n=int(input('n= '))
print('n= ', n)
v=randint(0,n, n)
x=v[range(0,n,2)]
y=v[range(1,n,2)]
m=int(n/2)
a=x.reshape([m,1])*y
print(a)
# alternativa sen vectorizar
a=zeros([m,m])
for i in range(m):
    for j in range(m):
        a[i,j]=x[i]*y[j]
b=zeros([m,m])
for i in range(m):
    for j in range(m):
        l=(i+1)*(j+1)
        if l%2 == 0 and l%3==0:
            b[i,j]= 3
        elif l%2 == 0:
            b[i,j]= 1
        elif l%3 == 0:
            b[i,j] = 2
        else:
            b[i,j] = 4
print('b= ')
print(b)
```

Primeiro control de programación en Python de 2019

Escribe un programa en Python que realice as seguintes operacións:

1. Pida por teclado un número enteiro n no intervalo [10, 30] que sexa múltiplo de 4, comprobando que o dato sexa correcto.
2. Crea un vector $\mathbf{x}$ de dimensión n con elementos x_i, con $i = 1, \ldots, n$, definidos pola expresión:

$$x_i = \begin{cases} \mathrm{e}^{-i^2} \operatorname{sen} i x_{i-1} & \text{se } n/4 < i < n/2 \\ \sqrt{10(i+1)^3} & \text{resto dos casos} \end{cases}$$

   onde $i = 1, \ldots, n$. Visualiza o vector $\mathbf{x}$ na pantalla.
3. Sexa z a media do vector $\mathbf{x}$. Calcula e visualiza na pantalla o número dos elementos de $\mathbf{x}$ que son superiores que a media e súa suma.

```
#!/usr/bin/python3
from numpy import *
n=0
while n<10 or n>30 or n%4:
    n=int(input('n (en [10,30] e multiplo de 4) = '))
print('n= ', n)
x=zeros(n)
inicio = n/4
fin = n/2
for i in range(n):
    if i > inicio and i<fin:
        x[i]= exp(-i*i)*sin(i*x[i-1])
    else:
        x[i]=sqrt(10*(i+1)**3)
print('x= ', x)
z=mean(x)
ind = where(x > z)[0]
print('No.elementos maior media de x: ', len(ind))
print('Suma elementos maior a media: ', sum(x[ind]))
```

Primeiro control de programación en Python de 2019

Escribe no editor de texto unha matriz de números, cun número de filas e columnas maior que 2, e gárdaa no arquivo `datos1.txt`. Por exemplo, co seguinte contido:

```
2 3 4 -5 7 2 5 9
4 1 8 4 2 6.2 7 1
8 1 5 3 2 2 4 2
10 1 1 1 1 1 1 1
8 7 4 3 2 1 0 -2
```

Escribe un programa en Python que realice as seguintes operacións:

1. Lea o arquivo `datos1.txt` a unha matriz **a** (non tes que comprobar os erros na lectura do arquivo), e visualiza na pantalla o número de filas nf e número de columnas nc. Se o número de filas ou columnas é menor que dous remata o programa.

2. Sexa x a suma dos elementos das dúas primeiras filas de **a**. Percorre a matriz por columnas sumando os seus elementos ata se supere o valor de x, é dicir, ata que:

$$\sum_{i=0}^{nc-1} \sum_{j=0}^{nf-1} a_{ji} > x$$

   Visualiza na pantalla a fila e columna na que se cumpre a condición e se deixa de sumar elementos.

3. Divide repetidamente todos os elementos da matriz **a** polo seu valor máximo +1 en cada iteracción ate que a suma dos elementos da matriz sexa menor ou igual que 1. Visualiza na pantalla o número de veces que se repetiu o proceso.

```
#!/usr/bin/python3
from numpy import *
from sys import *
a=loadtxt('datos3.txt')
print('a= ')
print(a)
[nf, nc]=a.shape
print('No. Filas a= %i e columnas= %i' % (nf, nc))
if nf<3 or nc < 3:
    exit()
x=sum(a[0:2, :])
suma=0
for i in range(nc):
    for j in range(nf):
        suma = suma + a[j, i]
        if suma > x:
            print('suma > %f en fila= %i e columna=%i' % (x,i, j))
            break
    if suma > x:
        break

n=0
while sum(a) > 1:
    a = a/(a.max()+1)
    n = n + 1
print('No. de veces: ', n)
```

Primeiro control de programación en Python de 2019

Escribe un programa en Python que realice as seguintes operacións:

1. Lea do teclado un número enteiro n que sexa múltiplo de 3 no intervalo [5,20]. O programa non pode avanzar até que n sexa correcto.
2. Xere un vector $\mathbf{x}$ con n valores enteiros aleatorios no intervalo [20, 50] e visualízao na pantalla.
3. Calcule un vector $\mathbf{y}$ que sexa o vector $\mathbf{x}$ barallado e calcula o número de elementos iguais nas mesma posición de ambos vectores.
4. Calcula un vector $\mathbf{z}$ de dimensión n, onde cada elemento z_i, con $i = 0, \ldots, n-1$ é;

$$z_i = \begin{cases} x_i y_{n-i-1} & \text{se } \sum_{k=0}^{i} x_k < \sum_{k=0}^{i} y_k \\ \dfrac{x_i}{y_i^2 + 1} & \text{resto dos casos} \end{cases}$$

5. Garda no arquivo `saida.txt` os tres vectores $\mathbf{x}$,$\mathbf{y}$ e $\mathbf{z}$ (un en cada liña) utilizando a función `savetxt()` do módulo `numpy`.

```
#!/usr/bin/python3
from numpy import *
from numpy.random import *
n=0
while n % 3  or n<5 or n>20:
    n=int(input('n= '))
x=randint(20, 50, n)
print('x= ', x)
y=x.copy()
shuffle(y)
print('y= ', y)
ind=extract(x == y, x)
print('No. elementos igual posicion: ', len(ind))
z=zeros(n)
for i in range(n):
    sx=sum(x[:i+1])
    sy=sum(y[:i+1])
    if sx < sy:
        z[i]=x[i]*y[n-i-1]
    else:
        z[i]=float(x[i])/(y[i]**2 + 1)
a=vstack((x, y, z))
savetxt('saida.txt', a, '%10.2f')
```

Primeiro control de programación en Python de 2019

Escribe un programa en Python que realice as seguintes operacións:

1. Lea do teclado un número enteiro n que sexa impar no intervalo [10,15]. O programa non pode avanzar ate que n sexa correcto.

2. Crea un vector $\mathbf{x}$ de con n elementos equidistantes entre 0 e n. Crea un vector $\mathbf{y}$, onde y_i, $i = 0, \ldots, n-1$ é $y_i = \text{sen}(2\pi i) + \cos(\pi(n-i)/4)$

3. Crea unha matriz $\mathbf{a}$ de orde n con valores consecutivos do vector $\mathbf{x}$ cando o índice da fila máis o índice da columna sexa par, e con valores consecutivos de $\mathbf{y}$ en caso contrario, é dicir, a_{ij}, con $i, j = 0, \ldots, n-1$ dado por:

$$a_{ij} = \begin{cases} x_k & k = 0, 1, \ldots \quad \text{se } i+j \text{ par} \\ y_l & l = 0, 1, \ldots \quad \text{resto dos casos} \end{cases}$$

   de xeito que se se chega ó final dos vectores $\mathbf{x}$ ou $\mathbf{y}$, volvamos a empezar desde o inicio.

4. Visualiza na pantalla os vectores $\mathbf{x}$ e $\mathbf{y}$, e a matriz $\mathbf{a}$.

```
#!/usr/bin/python3
from sys import *
from numpy import *
n=0
while n%2==0 or n<10 or n>15:
    n=int(input('n impar en [10,15]= '))
print('n= %i',n)
x=linspace(0,n, n)
y=zeros(n)
for i in range(n):
    y[i]=sin(2*pi*i)+ 2*cos(pi*(n-i)/4)
print('y= ', y)
a=zeros([n,n])
k=0; l=0
for i in range(n):
    for j in range(n):
        if (i+j) % 2: #impar
            a[i,j]=y[l]
            l= l+1
            if l==n:
                l=0
        else: # par
            a[i,j]=x[k]
            k=k+1
            if k == n:
                k=0
print('a= '); print(a)
```

Primeiro control de programación en Python de 2019

Escribe no editor de texto un vector de números enteiros e gárdao no arquivo `datosvector.txt`. Por exemplo, co seguinte contido:

```
2 7 3 4 5 9 12
```

Escribe un programa en Python que realice as seguintes operacións:

1. Lea o contido do arquivo `datosvector.txt` no vector **x**. Calcula ma (valor máximo do vector **x**), mi (valor mínimo do vector **x**) e n (número de elementos de **x**), e visualízaos na pantalla.

2. Define un vector **y** cos números enteiros no intervalo $[mi, ma]$ que **non** estean contidos no vector **x**. Calcula a dimensión de **y** e almacénaa na variable m.

3. Amplía ambos vectores (**x** e **y**) ate que a súa dimensión sexa $k = \max(n, m)$. Cando sexa necesario engadir elementos ó vector, engade elementos do principio do vector.

4. Calcula unha matriz **a** de orde na, onde $na = \lceil (1 + \sqrt{1 + 8k})/2 \rceil$ da seguinte maneira: almacena o vector **x** no triángulo superior da matriz **a**, o vector **y** no seu triángulo inferior e escribe na diagonal principal o valor 0. Visualiza na pantalla a matrix **a**.

```
#!/usr/bin/python3
from numpy import *
x=loadtxt('datosvector.txt', int)
ma=max(x); mi=min(x); n=len(x)
print('ma=%i mi=%i n=%i' % (ma, mi, n))
y=[]
for i in range(mi, ma+1):
    if not any(i == x):
        y.append(i)
m=len(y); k=max(n, m)
if k==n:
    for i in range(k-m):
        y.append(y[i])
else:
    for i in range(k-n):
        x=append(x, x[i])
print('x= ', x); print('y= ', y)
na=int(ceil((1+sqrt(1+8*k))/2))
a=zeros([na, na]); lx=0; ly=0
for i in range(na):
    for j in range(i+1,na):
        a[i, j]=x[lx]; lx=lx+1
        if lx ==k:
            break
    if lx == k:
        break
for i in range(na):
    for j in range(i):
        a[i,j]=y[ly]; ly=ly+1
        if ly==k:
            break
    if ly==k:
        break
print('a= '); print(a)
# codigo alternativo
```

```
#b=zeros([na, na]); fil=0; col=1
#for i in range(k):
    #b[fil, col]=x[i]
    #col = col +1
    #if col==na:
        #fil=fil+1; col = fil+1
#fil=1; col=0
#for i in range(k):
    #b[fil,col]=y[i]
    #col=col+1
    #if col == fil:
        #fil =fil+1; col=0
#print('b= '); print(b)
```