Primeiro control de programación en Python de 2025

Escribe un programa en Python que faga o seguinte:

1. Lea por teclado un número enteiro n maior que 10, verificando que o valor lido é correcto. Crea un vector $\mathbf{x}$ con n elementos, onde $x_0 = 2$, $x_1 = 1$, $x_2 = 1$ e $x_i = 2x_{i-3} - x_{i-2} - x_{i-1}$ para $i \geq 3$.

2. Calcula o número k de elementos do vector $\mathbf{x}$ que hai que sumar para que se cumpra a condición:

$$\sum_{i=0}^{k} x_i > \sum_{j=k}^{n-1} x_j \tag{1}$$

   e visualiza na pantalla o valor de k e dos dous sumatorios da formula anterior.

3. Se n é múltiplo de 3, transforma o vector $\mathbf{x}$ nunha matriz con 3 filas e as columnas que necesite. En caso contrario, calcula a media e a varianza do vector $\mathbf{x}$. Visualiza os resultados na pantalla.

```
from numpy import *
n=0
while n < 11:
    n=int(input("n= "))
x=[2,1,1]
for i in range(3,n):
    x.append(2*x[i-3]-x[i-2]+x[i-1])
print("x= ", x)
si=x[0]; sf=sum(x); k=1
while si < sf:
    si +=x[k]
    sf -=x[k-1]
    k=k+1
    print("k= ", k, "si= ", si, "sf= ", sf)
x=array(x, 'float')
if n%3:
    print('Media= ', mean(x), "Var= ", var(x))
else:
    a=x.reshape([3,n//3])
    print(a)
```

Primeiro control de programación en Python de 2025

Escribe un programa en Python que realice as seguintes operacións:

1. Lea por teclado un número enteiro n no intervalo [5, 30] que sexa múltiplo de 4, verificando que o número n é correcto.

2. Xera un vector $\mathbf{x}$ con n números enteiros aleatorios no intervalo [0, 10]. Crea un vector $\mathbf{y}$ con todos os elementos de $\mathbf{x}$, sen incluir as repeticións de elementos (se o elemento 2 está 3 veces en $\mathbf{x}$, só o incluímos unha vez en $\mathbf{y}$). Visualiza o vector $\mathbf{x}$ na pantalla, e o vector $\mathbf{y}$ ordeado de menor a maior.

3. Transforma o vector $\mathbf{x}$ nunha matriz a con 4 columnas. Se a é unha matriz cadrada, calcula o seu determinante. Se o número de filas é maior que o de columnas, suma os elementos de cada fila da matriz a e visualiza as sumas na pantalla. No caso de que o número de columnas sexa maior que o número de filas, calcula o número k de elementos do vector $\mathbf{x}$ que hai que sumar para superar a suma dos elementos do vector $\mathbf{y}$, é dicir:

$$\sum_{i=0}^{k} x_i > \sum_{i=0}^{m} y_i \tag{2}$$

   onde m é a lonxitude de $\mathbf{y}$. Mostra por pantalla k e as dúas sumas da ecuación anterior.

```
from numpy import *
from numpy.random import *
from numpy.linalg import *
n=0
while n < 5 or n>30 or n%4!=0:
    n=int(input("n= "))
x=randint(0,11, n)
print("x= ", x)
y=[]
for i in range(n):
    if x[i] not in x[i+1:n]:
        y.append(x[i])
print("y= ", sort(y))
# y=unique(x) # alternativa
nc=4; nf=n//nc
a=x.reshape([nf,nc])
if nf>nc:
    print('Suma filas: ', sum(a,1))
elif nf < nc:
    s=0;i=0;m=sum(y)
    while s<=m and i<n:
        s+=x[i]; i+=1
    print(' k=%d sx=%d sy=%d'%(i,s,m))
else:
    print('Determinante de a= ', det(a))
```

Primeiro control de programación en Python de 2025

Escribe un programa en Python que realice as seguintes operacións:

1. Lea por teclado números e os almacene nun vector $\mathbf{x}$. Sexa n a lonxitude de $\mathbf{x}$, e k a posición do valor mínimo de $\mathbf{x}$. Crea un vector $\mathbf{y}$ cos elementos x_i para $i = 0, \ldots, k$ e un vector $\mathbf{z}$ cos elementos x_i para $i = k, \ldots, n-1$. Visualiza na pantalla tódolos vectores.

2. Crea un vector $\mathbf{v}$ que conteña os elementos comúns a ambos vectores $\mathbf{z}$ e $\mathbf{y}$, e visualízao na pantalla.

3. Se $\mathbf{z}$ e $\mathbf{y}$ son de igual tamaño, crea e visualiza unha matriz $\mathbf{a}$ con dúas filas, unha para cada vector. Se a lonxitude de $\mathbf{z}$ é maior que a de $\mathbf{y}$, calcula o número q de elementos de $\mathbf{z}$ que hai que sumar para superar a suma de elementos de $\mathbf{y}$ (se chegamos ó final de $\mathbf{z}$ sen superar a suma, volve a comezar no principio outra vez). En caso de que $\mathbf{z}$ sexa de menor lonxitude que $\mathbf{y}$, visualiza na pantalla os elementos de $\mathbf{y}$ que son menores ó seu valor medio.

```
from numpy import *
x=float_(array(input('x= ').split()))
n=len(x)
k=argmin(x)
y=x[:k+1]
z=x[k:n]
print("x= ", x)
print("y= ", y)
print("z= ", z)
v=[]
for i in y:
    if i in z:
        v.append(i)
print('v= ', v)
ny=len(y); nz=len(z)
if ny == nz:
    a=vstack((y,z))
    print('a= ', a)
elif nz > ny:
    sy=sum(y); q=0; s=0; i=0
    while s < sy:
        s = s + z[i]
        q = q+1; i=i+1
        if i == nz:
            i=0
        # alternativa s= s + z[i%nz] que non necesita
        # comprobar se nos saimos do vector z
    print('s= ', s, 'q= ',q, 'sy= ', sy)
else:
    print(extract(y<mean(y), y))
```

Primeiro control de programación en Python de 2025

Escribe un programa en Python que realice as seguintes operacións:

1. Lea por teclado un número n enteiro par e maior que 7, verificando que é correcto. Xera unha matriz **a** cadrada de orde n con números reais aleatorios no intervalo [2, 10).

2. Crea unha matriz **b** cos elementos de **a** en filas e columnas pares. Crea outra matriz **c** cos elementos de **a** en filas e columnas impares. Visualiza ambas matrices.

3. Sexan s_b e s_c as suma dos elementos das matrices **b** e **c** respectivamente. Se $s_b > s_c$, suma os elementos da matriz **b** (percorrendo a matriz por filas) ate que a súa suma sexa maior que s_c, visualizando na pantalla a fila, columna e valor da matriz **b** onde se acada a condición. En caso contrario, divide todos os elementos da matriz **b** por 2, e visualiza na pantalla o número de veces que hai que repetir este proceso para que a suma dos elementos da matriz **b** sexa menor que 1.

```
from numpy import *
import numpy.random as rd
n=0
while n<8 or n%2==1:
    n=int(input('n= '))
a=2+8*rd.rand(n,n)
b=a[1:n:2,1:n:2]
c=a[0:n:2,0:n:2]
sb=sum(b); sc=sum(c)
print('sb= ', sb, 'sc= ', sc)
if sb > sc:
    s=0; m=b.shape[0]
    for i in range(m):
        for j in range(m):
            s = s + b[i,j]
            if s > sc:
                print('Valor= ', b[i,j], ' fila= ', i, ' columna= ',j)
                break
        if s > sc:
            break
else:
    veces=0
    while sum(b) > 1:
        b = b/2
        veces = veces + 1
    print('No. veces= ', veces)
```

Primeiro control de programación en Python de 2025

Escribe no editor de texto unha matriz con números enteiros como por exemplo:

```
2 4 7 3 8
1 0 4 9 3
4 4 1 2 3
```

e gárdao o arquivo `datos5.txt`. Escribe un programa en Python que realice as seguintes operacións:

1. Pida por teclado o nome do arquivo e almacene o seu contido na matriz **a** (sen comprobar erros). Visualiza **a** na pantalla. Sexa $n \times m$ a dimensión de **a**.

2. Calcula o valor de s dado pola expresión:

$$s = \sum_{i=0}^{n-1} \sum_{j=i}^{m-1} a_{ij}^2 \tag{3}$$

3. Se $s > 100$, visualiza a media m de **a** e os elementos de **a** que son superiores a m. En caso contrario, almacena no vector **x** os elementos de **a** que están repetidos. Cada elemento repetido debe aparecer unha soa vez en **x**. Neste exemplo serían o 1,2,3 e 4.

```python
from numpy import *
nome=input('Nome arquivo: ')
a=loadtxt(nome)
print('a= ', a)
[n,m]=a.shape
s=0
for i in range(n):
    for j in range(i, m):
        s = s +a[i,j]**2
print('s= ', s)
if s > 100:
    m=mean(a)
    print('medio= ', m)
    print('a > me= ', extract(a>m, a))
else:
    z=a.flatten()
    x=[]
    for i in z:
        if sum(z == i)>1 and t not in x:
            x.append(i)
    print('x= ', x)
```

Primeiro control de programación en Python de 2025

Escribe un programa en Python que realice as seguintes operacións:

1. Lee un número enteiro positivo n por teclado, verificando que é un número correcto (por exemplo $n = 10$). Lee tamén un número real q. Crea un vector $\mathbf{x}$ de lonxitude n con elementos x_i dados pola expresión:

$$x_i = \mathrm{e}^q \sin\left(\frac{q\pi i}{n}\right) \qquad i = 0, 1, \ldots, n-1 \tag{4}$$

   Crea outro vector $\mathbf{y}$ da mesma lonxitude con $y_0 = x_0$ e $y_i = y_{i-1} + x_i$ para $i = 1, \ldots, n-1$. Visualiza ambos vectores.

2. Se $q < 0$, calcula e visualiza unha matriz $\mathbf{a}$ de orde n, onde $a_{ij} = x_i y_j$ para $i, j = 0, 1, \ldots, n-1$.

3. Se $0 \leq q < 1$, xera un número enteiro aleatorio m no intervalo $[0, n)$, e comezando no índice m suma elementos do vector $\mathbf{x}$ ate que a súa suma sexa superior a $s_y = \sum_{i=0}^{n-1} y_i$. Se chegas ó final do vector $\mathbf{x}$ sen superar s_y, continúa polo principio de $\mathbf{x}$. Visualiza o número de elementos sumados, o índice do vector $\mathbf{x}$ no que paramos e o seu valor.

4. Se $q \geq 1$, calcula e visualiza un vector $\mathbf{z}$ de lonxitude n, onde z_i para $i = 0, 1, \ldots, n-1$ é:

$$z_i = \sum_{j=0}^{i} x_j + \sum_{k=i}^{n-1} y_k \tag{5}$$

```
from numpy import *
import random
n=0
while n<1:
    n=int(input('n= '))
q=float(input('q= '))
t=arange(n)
x=exp(q)*sin(q*pi*t/n)
y=zeros(n); y[0]=x[0]
for i in range(1,n):
    y[i]=y[i-1]*x[i]
print('x= ', x); print('y= ', y)
if q<0:
    a=x.reshape([n,1])*y
    print('a= ', a)
elif q>= 0 and q<1:
    m=random.randint(0,n-1); print('m inicial= ', m)
    s=x[m]; sy=sum(y); k=1
    while s<sy:
        m += 1
        if m == n:
            m=0
        s = s + x[m]; k += 1
    print('Indice= ', m, ' Valor= ', x[m], 'No. elem.= ', k)
else:
    z=zeros(n)
    for i in range(n):
        z[i]=sum(x[:i+1])+sum(y[i:n])
    print('z= ', z)
```