Exame de xullo de programación en Python de 2019

Crea co editor un arquivo de texto un arquivo `datos1.dat` con n números enteiros nunha única fila. Por exemplo, co seguinte contido: `5 4 3 8 6 13` (supón que os números non están repetidos) n sería 6. Crea co editor un segundo arquivo de texto, co nome que ti queiras, que teña m liñas con n números enteiros cada liña. Por exemplo, podería ser o seguinte arquivo:

```
1 2 3 4 5 6
5 4 3 8 6 13
4 1 3 5 0 2
4 5 4 4 6 3
```

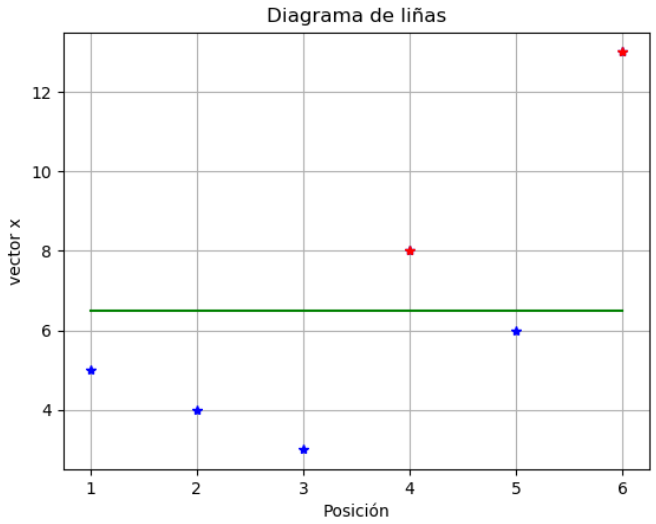
Escribe un programa en Python chamado `xullo.py` que realice as seguintes operacións:

1. Ler o arquivo `datos1.dat` e almacénao no vector **x**. Pedirlle ó usuario o nome do segundo arquivo de texto e le o arquivo á matriz **a**. Se o tamaño do vector **x** e o número de columnas da matriz **a** non coinciden rematar o programa. En caso contrario, o programa debe visualiza na pantalla o valor de n e m.

2. Calcula un vector **y**, de dimensión m, de modo que y_i, con $i = 0, \ldots, m-1$, sexa o número de veces que calquera elemento de **x** aparece na fila i de **a**. Por exemplo, cos datos anteriores o vector **y** tería o contido: `4 6 3 6`.

3. Define unha función chamada `calculos(...)`, cos argumentos axeitados, que a partir dun vector **v** e unha matriz **b**, retorne o número de elementos pares de **b** (percorrendo a matriz por filas) que hai que sumar para superar o valor $sv = \frac{\sum_{i=0}^{n-1} x_i}{n}$. Se a suma dos elementos pares de **b** fose inferior á sv, a función debe devolver o número dos elementos pares de **b**. O programa principal debe chamar á función `calculos(...)` para que se aplique sobre a matriz **a** e o vector **x**, e visualizar o resultado que devolve na pantalla.

4. Realizar un diagrama de liñas representando os elementos do vector **x** no eixo y e no eixo x as posicións dos elementos (vector 1, 2, 3, 4, ...). Representa os puntos en cor azul. Debuxa unha liña verde correspondente a media dos elementos de **x**. Pon en cor vermella os puntos maiores ca media. Pon título e etiquetas nos eixo ó gráfico. Gardao co nome `grafico.png`. Co anterior vector **x** a figura resultante sería:




```python
#-*-coding:utf-8-*-
from numpy import *
```

```
from sys import exit

name=raw_input('Nome arquivo matriz: ')
try:
    x=loadtxt('datos.dat', int)
    a=loadtxt(name, int)
    n=len(x)
    m=size(a, 0)
    na=size(a,1)
    if(n != na):
        exit('Non coinciden no tamanho')
    else:
        print 'n= ', n, ' m= ', m
        print 'x= ', x
        print 'a= ', a
except IOError:
    exit('Erro lendo arquivos')

y=zeros(m);
for i in range(m):
    k=0
    z=a[i]
    for j in range(n):
        k= k + sum(z == x[j])
    y[i]=k
print 'y= ', y

def calculos(v, b):
    sv=mean(v)
    nf=size(b, 0)
    nc=size(b, 1)
    suma=0
    w=extract(b%2==0, b)
    l=len(w)
    for i in range(l):
        suma=suma + w[i]
        if suma > sv:
            break
    return i+1
    # outra posibilidade
    #nelem=0
    #for i in range(nf):
        #for j in range(nc):
            #if b[i,j]%2 == 0:
                #suma = suma + b[i,j]
                #nelem +=1
            #if suma > sv:
                #break
        #if suma > sv:
            #break
    #return nelem

print 'No. elementos sumados: ', calculos(x,a)
from matplotlib.pyplot import *
t=range(1, n+1)
plot(t, x, 'b*')
media=mean(x)
```

```
plot([1, n], [media, media], 'g-')
p=where(x > media)[0]
plot(p+1, x[p], 'r*')
title(u'Diagrama de liñas')
xlabel(u'Posición'); ylabel('vector x')
grid(True)
savefig('grafico.png')
show(False)
```