

**Examen domiciliario de Computación Aplicada y PLC – 20 de noviembre 2020**  
**Nombre: Brian Ivan Casaña**

**Ejercicio 1 – en octave**

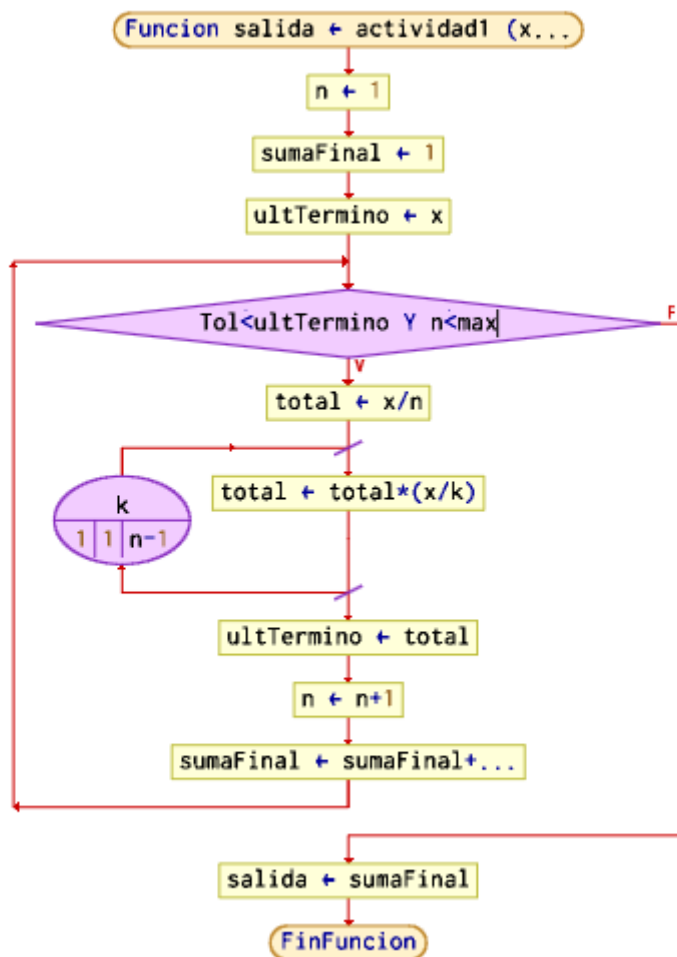
Se desea calcular la suma de los primeros términos de la siguiente expresión

$$1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$$

para un valor de x, hasta que el último término sumado sea menor que un determinado valor denominado Tol o hasta que el número de términos sumados sea mayor a Max

Realizar una función que tenga como variables de entrada x, Tol y Max y como salida el valor de suma

**Diagrama de flujo:**



### Variables utilizadas:

x : double

Tol : double

max : double

sumaFinal : double

ultTermino : double

total : double

k : double

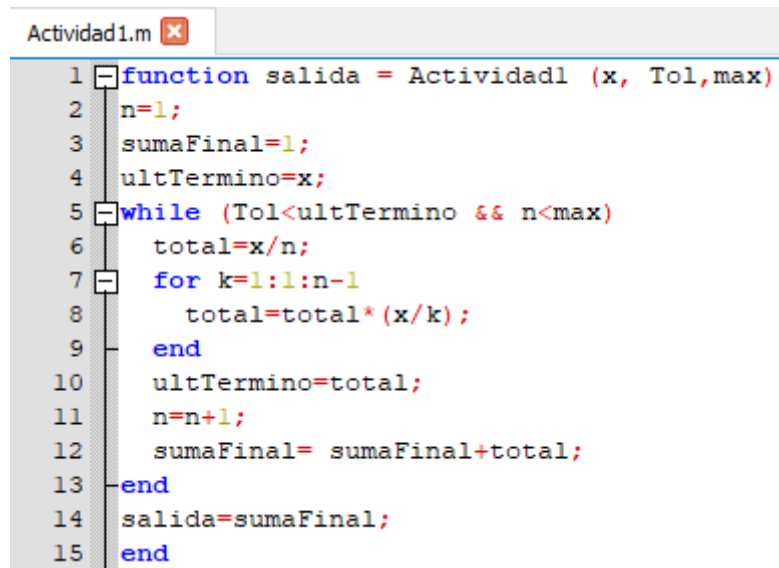
salida : double

### Funciones utilizadas:

While

For

### Programa desarrollado:



```
Actividad1.m x
1 function salida = Actividad1 (x, Tol,max)
2     n=1;
3     sumaFinal=1;
4     ultTermino=x;
5     while (Tol<ultTermino && n<max)
6         total=x/n;
7         for k=1:1:n-1
8             total=total*(x/k);
9         end
10        ultTermino=total;
11        n=n+1;
12        sumaFinal= sumaFinal+total;
13    end
14    salida=sumaFinal;
15 end
```

**Resultado:**

```
>> ans=Actividad1(5,0.1,10)
ans = 143.69
>>
```

<				
Workspace				
Filter <input type="checkbox"/>				
Name	Class	Dimension	Value	Attribute
ans	double	1x1	143.69	

**Ejercicio 2 – en octave**

Construya una función que la entrada sea una matriz de 9x9, el número de fila (i) y columna(j) y que tenga como salida tres vectores

Vector 1:

formada por los nueve elementos correspondiente a la matriz de 3x3 inscripta.

La matriz la podemos pensar dividida en 9 matrices de 3x3 de acuerdo al siguiente grafico.

M1	M1	M1	M2	M2	M2	M3	M3	M3
M1	M1	M1	M2	M2	M2	M3	M3	M3
M1	M1	M1	M2	M2	M2	M3	M3	M3
M4	M4	M4	M5	M5	M5	M6	M6	M6
M4	M4	M4	M5	M5	M5	M6	M6	M6
M4	M4	M4	M5	M5	M5	M6	M6	M6
M7	M7	M7	M8	M8	M8	M9	M9	M9
M7	M7	M7	M8	M8	M8	M9	M9	M9
M7	M7	M7	M8	M8	M8	M9	M9	M9

vector 2:

formada por los elementos de la fila i de la matriz

vector 3:

formada por los elementos de la columna j de la matriz

Por ejemplo si tenemos la siguiente matriz de entrada, el número de fila 3, columna 4

9	6	3	1	7	4	2	5	8
1	7	8	3	2	5	6	4	9
2	5	4	6	8	9	7	3	1
8	2	1	4	3	7	5	9	6
4	9	6	8	5	2	3	1	7
7	3	5	9	6	1	8	2	4
5	8	9	7	1	3	4	6	2
3	1	7	2	4	6	9	8	5
6	4	2	5	9	8	1	7	3

---

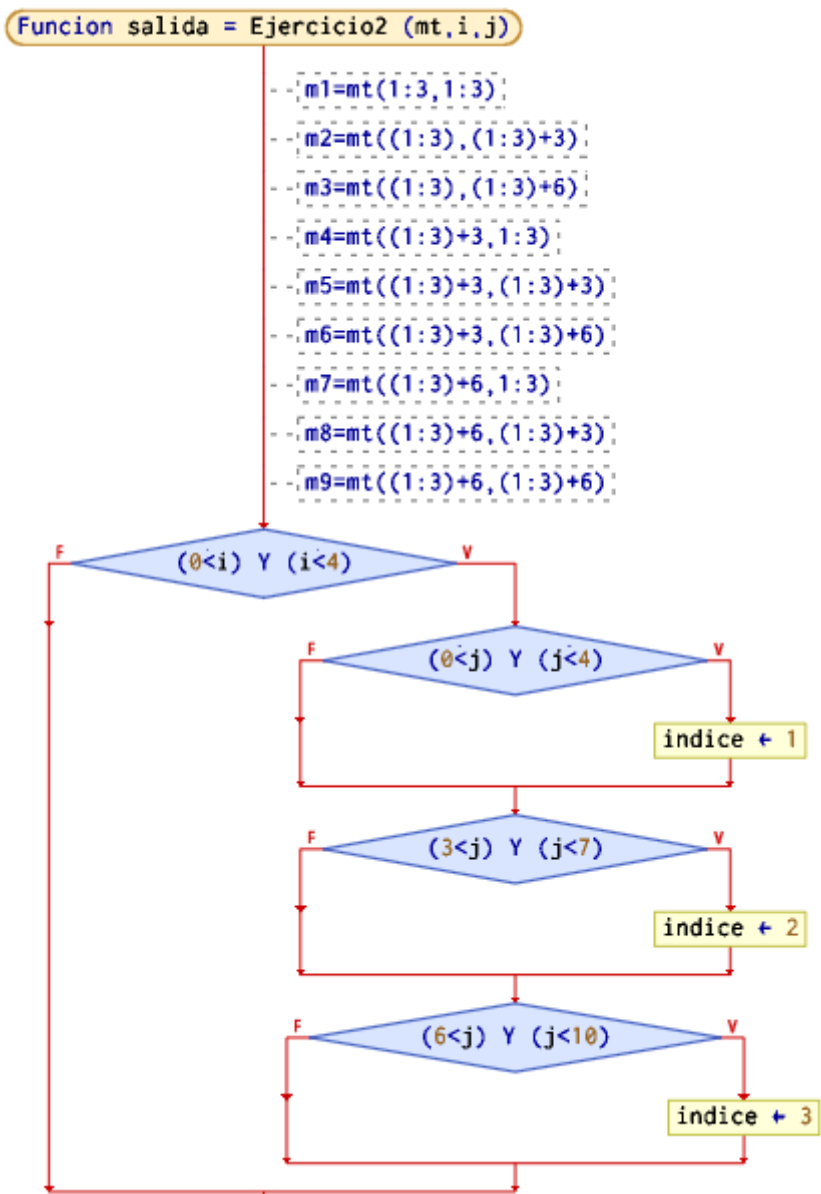
El programa deberá entregar:

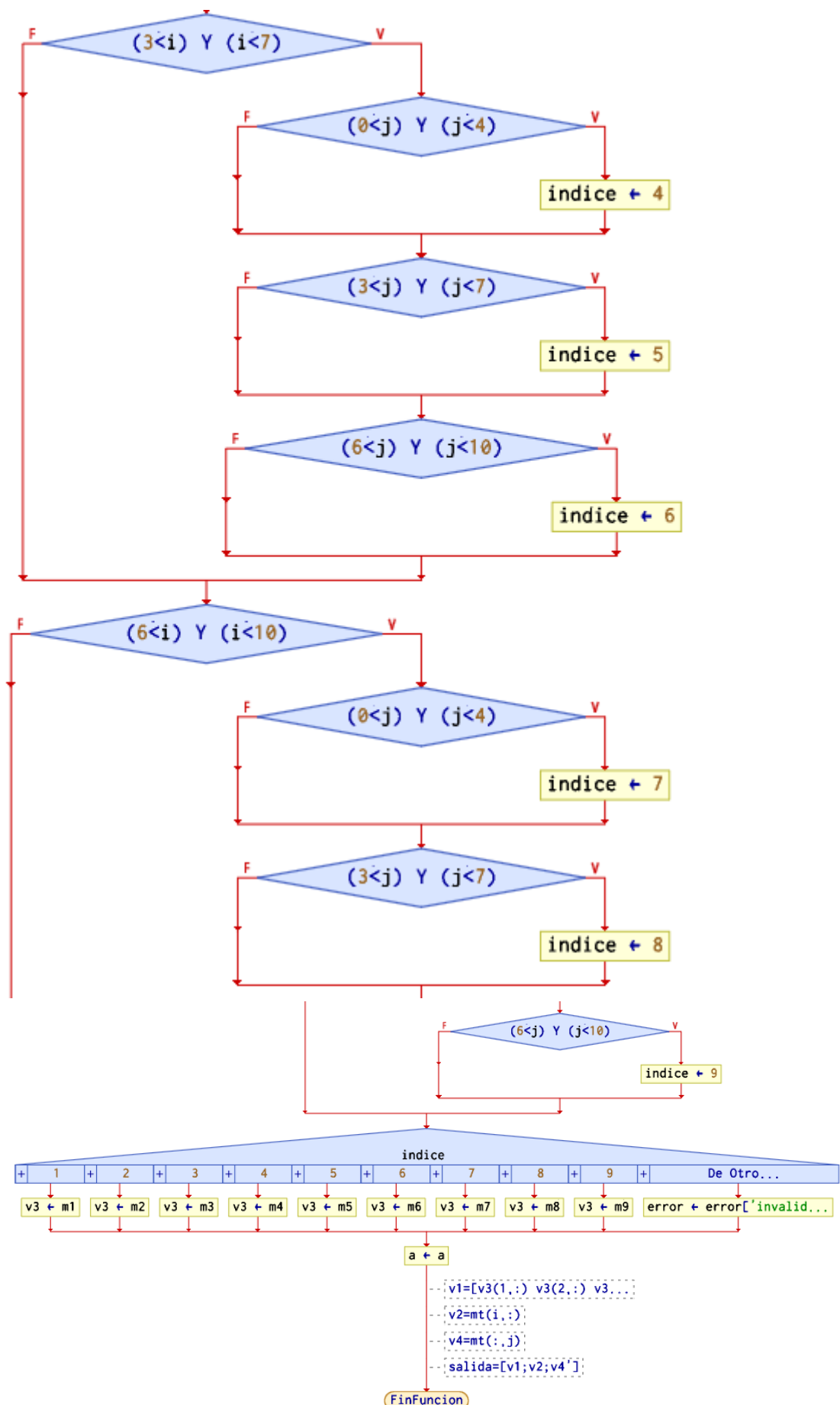
V1=[2 5 4 6 8 9 7 3 1]

V2=[1 3 6 4 8 9 7 2 5]

V3=[1 7 4 3 2 5 6 8 9]

Diagrama de flujo:





Variables usadas:

mt : 9x9 double  
m1: 3x3 double  
m2: 3x3 double  
m3: 3x3 double  
m4: 3x3 double  
m5: 3x3 double  
m6: 3x3 double  
m7: 3x3 double  
m8: 3x3 double  
m9: 3x3 double  
i: double  
j: double  
índice: double  
v3 :3x3 double  
v1: 3x9 double  
v2: 3x9 double  
v4: 9x3 double  
salida: 3x9 double  
ans: 3x9 double

Funciones utilizadas:

if()

switch()

Programa desarrollado:

```
Ejercicio2.m ✕  
1 function salida = Ejercicio2 (mt,i, j)  
2     m1=mt (1:3,1:3);  
3     m2=mt ((1:3), (1:3)+3);  
4     m3=mt ((1:3), (1:3)+6);  
5     m4=mt ((1:3)+3,1:3);  
6     m5=mt ((1:3)+3, (1:3)+3);  
7     m6=mt ((1:3)+3, (1:3)+6);  
8     m7=mt ((1:3)+6,1:3);  
9     m8=mt ((1:3)+6, (1:3)+3);  
10    m9=mt ((1:3)+6, (1:3)+6);  
11    if ((0<i) && (i<4) )  
12        if((0<j) && (j<4) )  
13            indice=1;  
14        end  
15    elseif((3<j) && (j<7) )  
16        indice=2;  
17    end  
18    elseif((6<j) && (j<10) )  
19        indice=3;  
20    end
```

```

21 end
22 if ((3<i) && (i<7) )
23     if((0<j) && (j<4) )
24         indice=4;
25     end
26     if((3<j) && (j<7) )
27         indice=5;
28     end
29     if((6<j) && (j<10) )
30         indice=6;
31     end
32 end
33 if ((6<i) && (i<10) )
34     if((0<j) && (j<4) )
35         indice=7;
36     end
37     if((3<j) && (j<7) )
38         indice=8;
39     end
40     if((6<j) && (j<10) )
41         indice=9;
42     end
43 end
44 switch (indice)
45     case {1}
46         v3 = m1;
47     case {2}
48         v3 = m2;
49     case {3}
50         v3 = m3;
51     case {4}
52         v3 = m4;
53     case {5}
54         v3 = m5;
55     case {6}
56         v3 = m6;
57     case {7}
58         v3 = m7;
59     case {8}
60         v3 = m8;
61     case {9}
62         v3 = m9;
63     otherwise
64         error ("invalid value");
65 end
66 v1=[v3(1,:) v3(2,:) v3(3,:)] ;
67 v2=mt(i,:);
68 v4=mt(:,j);
69 salida=[v1;v2;v4'];
70 end
71

```



Resultado:

```
>> ans=Ejercicio2(mt,4,4)
```

ans =

31	32	33	40	41	42	49	50	51
28	29	30	31	32	33	34	35	36
4	13	22	31	40	49	58	67	76

```
>> mt
```

mt =

1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54
55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81

---

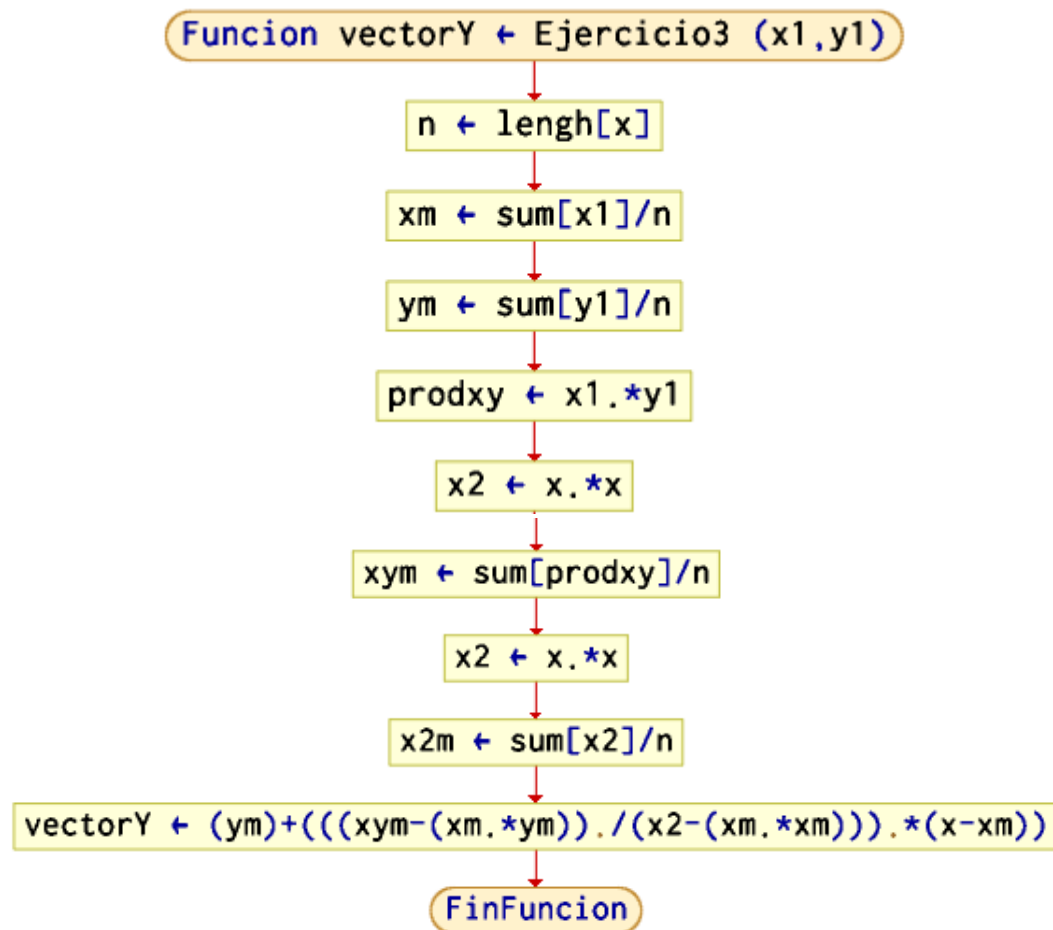
### Ejercicio 3 – en octave

Construir una función que tenga como parámetros de entrada dos vectores (x) e (y) del mismo tamaño. Como parámetro de salida entregue un vector Y

$$\bar{x} = \frac{\sum_{i=1}^n x(i)}{n} \quad \bar{y} = \frac{\sum_{i=1}^n y(i)}{n} \quad \bar{xy} = \frac{\sum_{i=1}^n x(i)y(i)}{n} \quad \bar{x^2} = \frac{\sum_{i=1}^n x(i)x(i)}{n}$$

$$Y(i) = \bar{y} + \frac{\bar{xy} - \bar{x}\bar{y}}{\bar{x^2} - (\bar{x})^2} (x(i) - \bar{x})$$

Diagrama de flujo:



Variables utilizadas:

x: 1x2 double

y: 1x2 double

n: double

xm: double

ym: double

prodxy: 1x2

xym: double

x2: 1x2 double

x2m: double

vectorY: 1x2 double

Funciones utilizadas:

length()

sum()

Programa desarrollado:

```
Ejercicio3.m x
1 function vectorY = Ejercicio3 (x,y)
2   n=length(x);
3
4   xm=sum(x)/n; %xmmedio
5   ym=sum(y)/n; %ymedio
6   prodxy=x.*y;
7   xym=sum(prodxy)/n; %xy medio
8   x2=x.*x;
9   x2m=sum(x2)/n; %x^2 medio
10  vectorY=(ym)+((xym-(xm.*ym))./(x2-(xm.*xm)).*(x-xm));
11
12
13 end
```

Resultado:

```
>> x=[1 2]
>> y=[3 4]
>> Ejercicio3(x,y)
ans =

    3.6000    3.5714

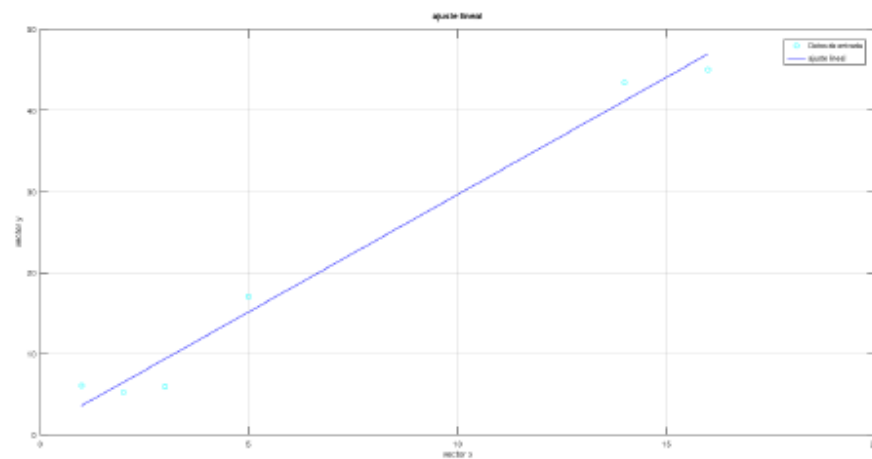
>>
```

---

#### Ejercicio 4 – en octave

Construir una script que solicite el nombre de archivo de entrada para procesar los datos utilizando la función del ejercicio anterior.

Posteriormente deberá graficar los puntos del vector de entrada (x,y) y el vector (x,Y) como se indica a continuación (deberá tener título, nombre en el eje x, nombre en el eje y, leyenda y grilla)



Los nombres de los archivos adjuntos son los siguientes.

- datosejercicio4\_x.txt (donde x es un número entero)

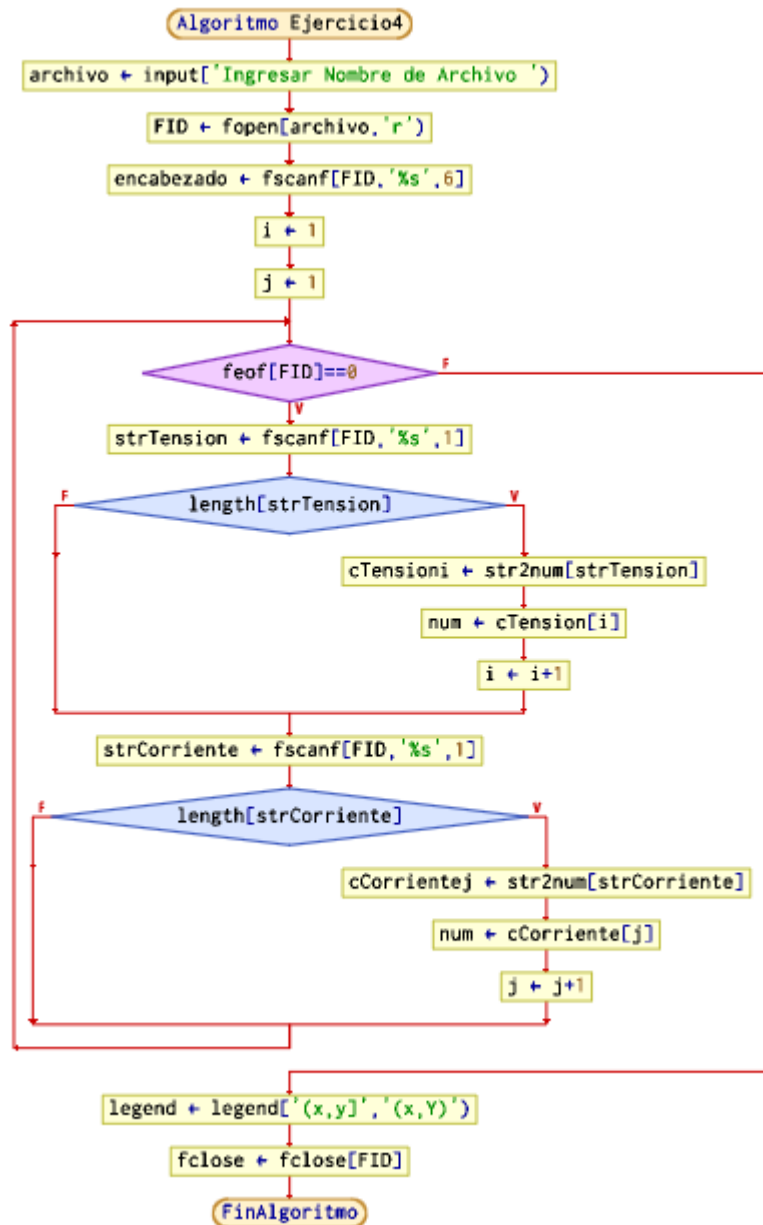


Diagrama de flujo:

Variables utilizadas:

Name	Class	Dimension
FID	double	1x1
Y	double	1x5581
ans	double	1x1
archivo	char	1x20
cCorriente	double	1x5581
cTension	double	1x5581
encabezado	char	1x50
i	double	1x1
j	double	1x1
num	double	1x1
strCorriente	char	1x0
strTension	char	1x0

Funciones utilizadas:

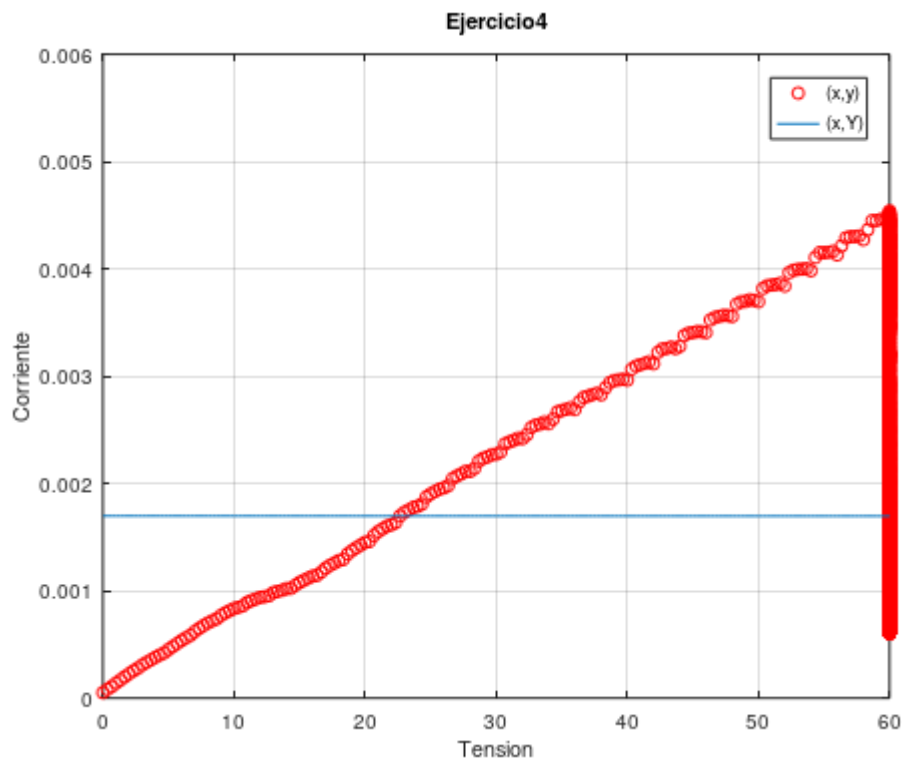
input()  
fopen()  
fscanf()  
while()  
feof()  
length()  
str2num()  
plot()  
grid on  
title()  
xlabel()  
ylabel()  
legend()  
fclose()

Programa desarrollado:

Ejercicio4.m

```
1 archivo=input("Ingresar Nombre de Archivo ('dataejercicio4_1.txt'): ");
2 FID=fopen(archivo,'r');%abrir archivo
3 encabezado=fscanf(FID,'%s',6);%lectura encabezado
4 i=1;
5 j=1;
6 while (feof(FID)==0)
7     strTension=fscanf(FID,'%s',1);
8     if length(strTension)
9         cTension(i)=str2num (strTension);
10        num=cTension(i);
11        i=i+1;
12    end
13    strCorriente=fscanf(FID,'%s',1);
14    if length(strTension)
15        cCorriente(j)=str2num (strCorriente);
16        num=cCorriente(j);
17        j=j+1;
18    end
19 end
20 Y=Ejercicio3(cTension,cCorriente);
21 plot(cTension,cCorriente,'ro',cTension,Y)
22 grid on;
23 title("Ejercicio4");
24 xlabel("Tension");
25 ylabel("Corriente");
26 legend("(x,y)", "(x,Y)")
27 fclose(FID)
```

Resultado:



---

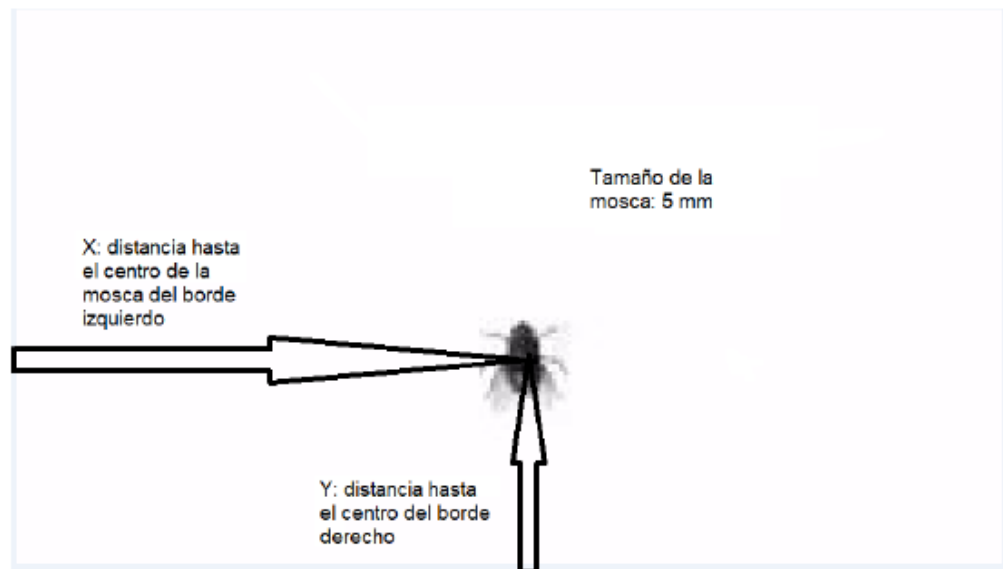
### Ejercicio 5 – en octave

Construir un script que solicite el nombre de archivo de entrada para procesar las imagen de una mosca moviéndose con los siguiente nombres.

- fotoejercicio5\_x.jpg (donde x es un número entero)

la variable de salida tiene que ser dos variables x,y . Que corresponde a la ubicación de la mosca en mm.

Utilice el paquete de procesamiento de imágenes de octave image (recuerde que para utilizarlo tiene que utilizar el comando pkg load image).



No es necesario realizar el diagrama de flujo.

---



Variables utilizadas:

Name	Class	Dimension
A	uint8	144x256x3
Abin	logical	144x256
Abin_dilate	logical	144x256
Abin_rode	logical	144x256
Agris	uint8	144x256
Ainv	uint8	144x256x3
Box	double	1x4
D	struct	2x1
L	double	144x256
ans	char	1x21
center	double	1x2
dx	double	1x1
dy	double	1x1
imagen	char	1x21
n	double	1x1
relacionPixe...	double	1x1
se	strel	1x1
t	double	1x1
tamanoMos...	double	1x1
tamanoMos...	double	1x1
x	double	1x1
y	double	1x1

Funciones utilizadas:

input()  
imread()  
imadjust()  
rgb2gray()  
graythresh()  
im2bw()  
strel()  
imdilate()  
imerode()  
bwlablel()  
regionprops()

### Programa desarrollado:

```
Ejercicio5.m x
1 pkg load image;
2 tamanoMoscammm=5;
3 imagen=input("Ingrese Nombre de la Imagen(\"fotoejercicio5_1.jpg\"): ");
4 A=imread(imagen);
5 Ainv=imadjust(A,[0 1],[1 0]); #Ajustes de imagen
6 Agris=rgb2gray(Ainv);
7 t=graythresh(Agris);
8 Abin=im2bw(Agris,t);
9 se=strel('rectangle',[3 3]);
10 Abin_dilate=imdilate(Abin,se);
11 Abin_rode=imerode(Abin_dilate,se);
12 [L,n]=bwlabel(Abin_rode);
13 D=regionprops(L);
14 Box=D(1).BoundingBox;
15 tamanoMoscaPixel=Box(4);#relacion de pixel a mm
16 relacionPixelmm=tamanoMoscaPixel/tamanoMoscammm; #lo que equivale un mm en pixeles
17 center=D(1).Centroid;
18 dx=Box(1);#distancia hasta el centro de la mosca del borde izquierdo
19 dy= center(2);#distancia hasta el centro del borde derecho
20 x=dx/relacionPixelmm
21 y=dy/relacionPixelmm
22
```

### Resultado:

```
>> Ejercicio5
```

```
Ingrese Nombre de la Imagen("fotoejercicio5_1.jpg"): "fotoejercicio5_2.jpeg"
x = 36.618
y = 20.975
```

### Ejercicio 6 – en octave

Construir una script que solicite el número de fotos que se quieren procesar de las imágenes del ejercicio anterior deberá graficar los puntos del vector de entrada (x,y) y el vector (x,Y) de forma similar al ejercicio 4

No es necesario realizar el diagrama de flujo.

Variables utilizadas:

A	uint8	144x256x3
Abin	logical	144x256
Abin_dilate	logical	144x256
Abin_rode	logical	144x256
Agris	uint8	144x256
Ainv	uint8	144x256x3
Box	double	1x4
D	struct	1x1
L	double	144x256
Y	double	1x5
ans	char	1x21
cant	double	1x1
center	double	1x2
dx	double	1x1
dy	double	1x1
imagen	char	1x21
k	double	1x1
n	double	1x1
relacionPixe...	double	1x1
se	strel	1x1
t	double	1x1
tamanoMos...	double	1x1
tamanoMos...	double	1x1
x	double	1x1
xVec	double	1x5
y	double	1x1
yVec	double	1x5

Funciones utilizadas:

input()  
for()  
length()  
sum()  
Ejercicio3()  
imread()

imadjust()  
rgb2gray()  
graythresh()  
im2bw()  
strel()  
imdilate()  
imerode()  
bwhlabel()  
regionprops()

Programa desarrollado:

```
Ejercicio6.m
1 cant=input("Ingresar Cantidad de imagenes que desea procesar: ");
2 for k=1:1:cant
3     Ejercicio5
4     xVec(k)=x;
5     yVec(k)=y;
6 endfor
7 Y=Ejercicio3(xVec,yVec);
8 plot(xVec,yVec,'ro',xVec,Y)
9 grid on;
10 title("Ejercicio6");
11 xlabel("X");
12 ylabel("Y");
13 legend("(x,y)", "(x,Y)");
```

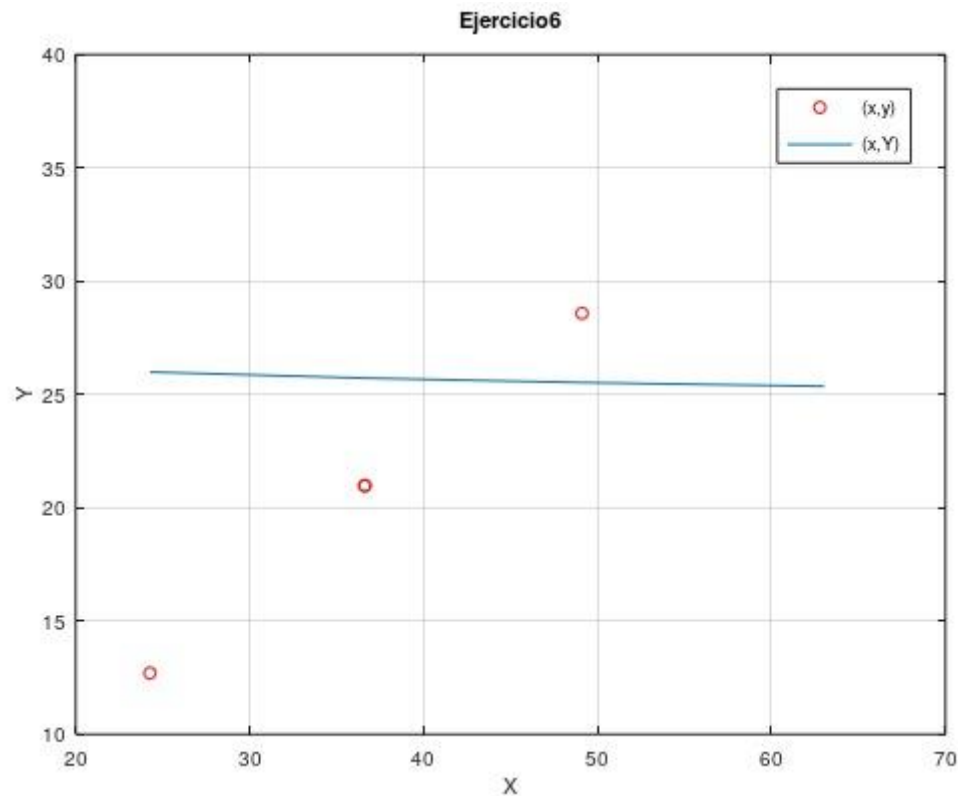
Resultado:

```
Ingresar Cantidad de imagenes que desea pro
cesar: 5
Ingrese Nombre de la Imagen("fotoejercicio5
_1.jpg"): "fotoejercicio5_1.jpeg"
x = 24.265
y = 12.700
Ingrese Nombre de la Imagen("fotoejercicio5
_1.jpg"): "fotoejercicio5_2.jpeg"
x = 36.618
y = 20.975
Ingrese Nombre de la Imagen("fotoejercicio5
_1.jpg"): "fotoejercicio5_3.jpeg"
x = 36.618
y = 20.975
Ingrese Nombre de la Imagen("fotoejercicio5
_1.jpg"): "fotoejercicio5_4.jpeg"
x = 49.107
y = 28.581
```

```

Ingrese Nombre de la Imagen("fotoejercicio5
_1.jpg"): "fotoejercicio5_5.jpeg"
x = 63.036
y = 38.186
>> |

```



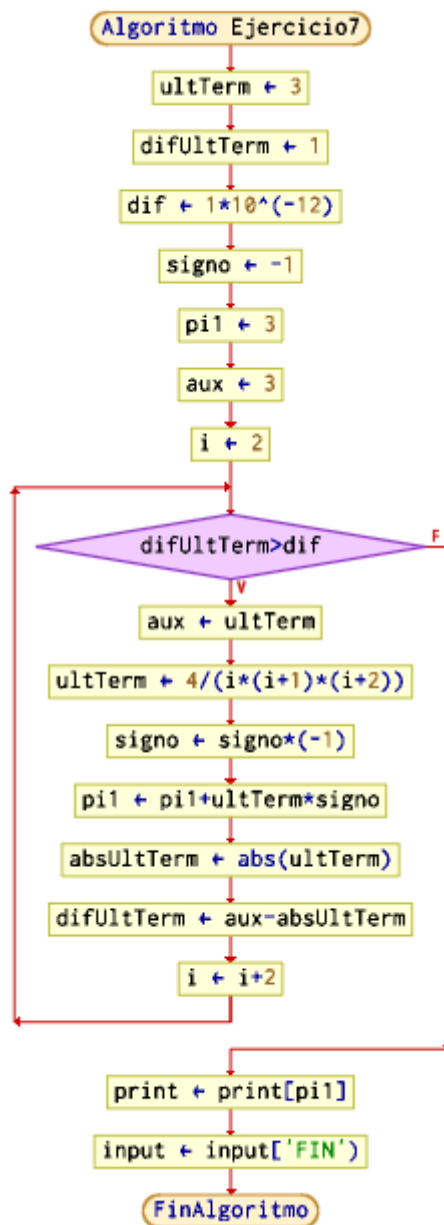
### Ejercicio 7 – en python

El valor de  $\pi$  se puede aproximar realizando la suma parcial de la siguiente serie infinita:

$$\pi = 3 + \frac{4}{2 \times 3 \times 4} - \frac{4}{4 \times 5 \times 6} + \frac{4}{6 \times 7 \times 8} - \frac{4}{8 \times 9 \times 10} + \dots$$

realizar un programa que estime el valor  $\pi$  para ello realice estimaciones aumentando el número de términos de la suma parcial de la serie hasta lograr que la diferencia entre dos términos consecutivos sea menor a  $10^{-12}$ .

Diagrama de flujo:



Variables utilizadas:

ultTerm float  
difUltTerm float  
dif float  
signo int  
aux float  
pi float  
i int  
absUltTerm float

Funciones utilizadas:

while:

Abs()  
print()

Programa desarrollado:

```
ultTerm=3
difUltTerm=1
dif=1*10**(-12)
signo=-1
pi=3
aux=3
i=2
while difUltTerm>dif:
    aux=ultTerm
    ultTerm = 4 / (i*(i+1)*(i+2))
    signo=signo*(-1)
    pi = pi + ultTerm*signo
    absUltTerm=abs(ultTerm)
    difUltTerm=aux-absUltTerm
    i=i+2

print(pi)
```

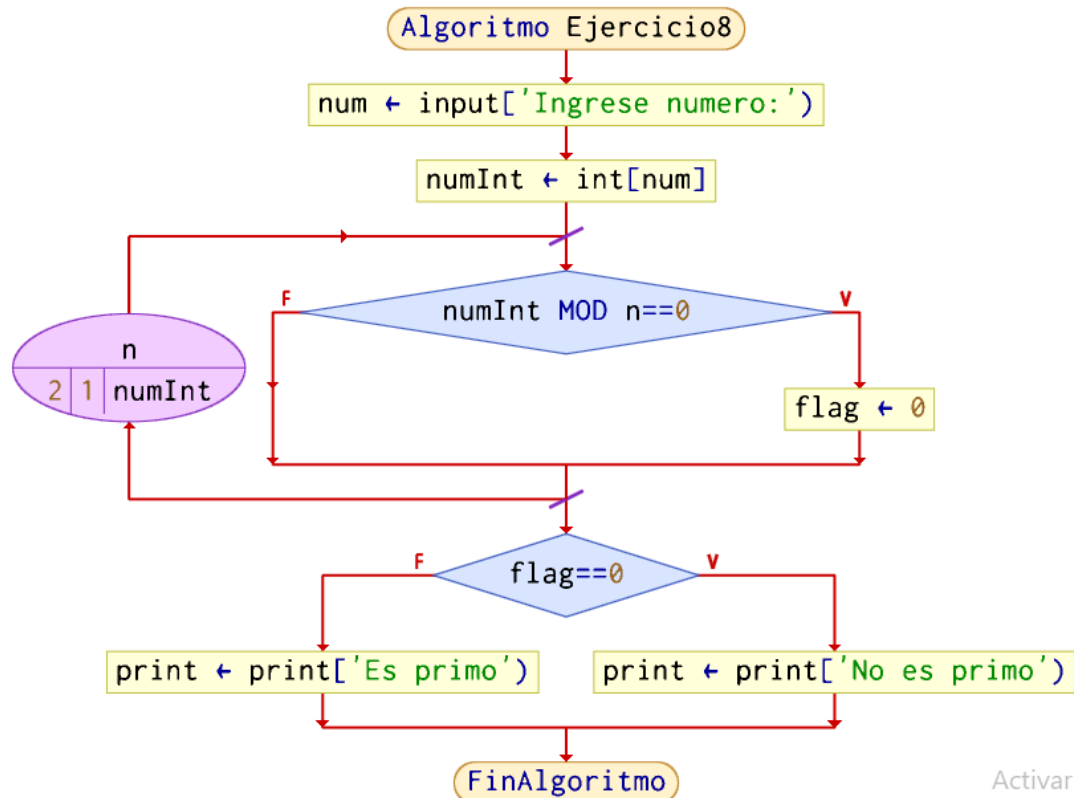
Resultado:

```
= RESTART: C:\Users\Ivan PC\Desktop\Examen Domiciliario\Resolucion Examen Domici
liario\Ejercicios Resueltos\Ejercicio7.py
3.1415926537735834
>>> |
```

## Ejercicio 8 – en python

Escriba un programa que lea un número entero positivo y el programa indique si el número es primo o no.

Diagrama de flujo:



Activar Wir

Variables utilizadas:

num string  
numInt int  
flag int  
n int

Funciones utilizadas:

input()  
for:  
if:

Programa desarrollado:

```
num = input("Ingrese numero:")
numInt=int(num)
flag=1
for n in range(2, numInt):
    if numInt%n == 0:
        flag=0

if flag==0:
    print("No es primo")
else:
    print("Es primo")
```



Resultado:

```
= RESTART: C:/Users/Ivan PC/Desktop/Examen Domiciliario/Resolucion Examen Domiciliario/Ejercicios Resueltos/Ejercicio8.py
Ingrese numero:101
Es primo
>>>
= RESTART: C:/Users/Ivan PC/Desktop/Examen Domiciliario/Resolucion Examen Domiciliario/Ejercicios Resueltos/Ejercicio8.py
Ingrese numero:12
No es primo
```

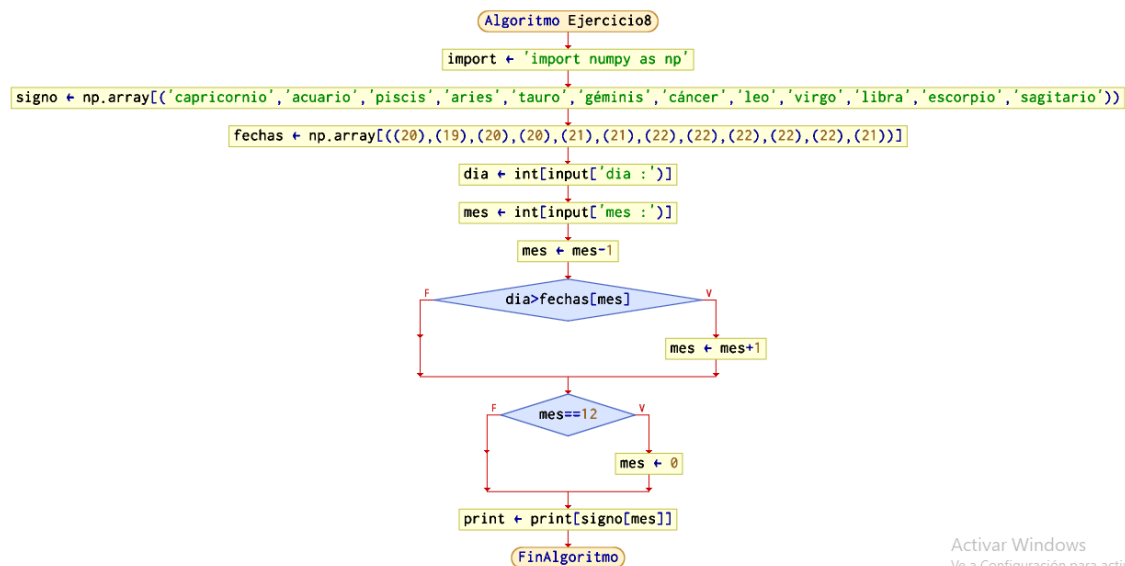
### Ejercicio 9 – en python - Utilice la librería numpy

La astrología divide el año en doce signos del zodiaco (para mas detalles puede consultar

<https://es.wikipedia.org/wiki/Zodiaco>)

Escriba un programa que lea el día de cumpleaños del usuario. Su programa debe mostrar el signo del zodiaco del usuario.

Diagrama de flujo:



Activar Windows  
Ve a Configuración para activar

Variables utilizadas:

signo numpy.ndarray

fechas numpy.ndarray

dia int

mes int

Funciones utilizadas:

array()

input()

if:

print()



potencia numpy.ndarray

Funciones utilizadas:

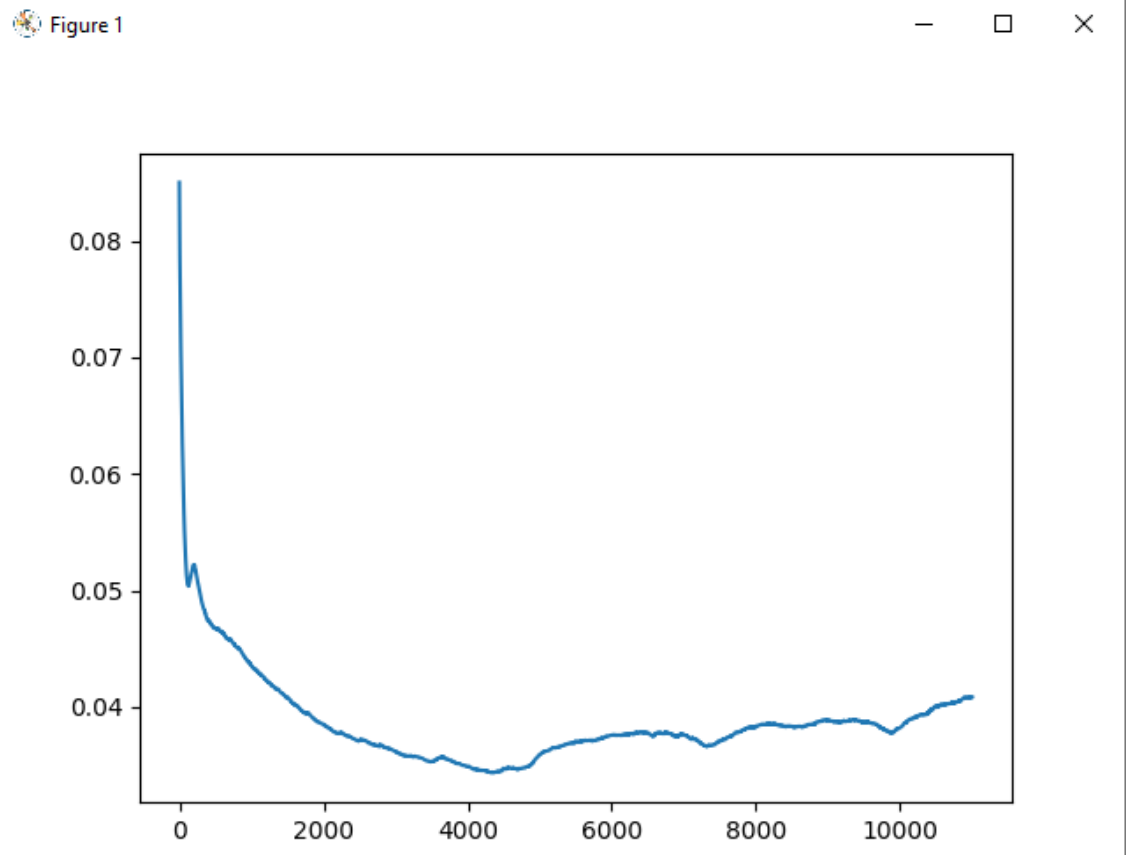
input()  
read()  
replace()  
split()  
len()  
int()  
empty()  
for:  
float()  
if:  
elif:  
plot()  
show()

Programa desarrollado:

```
import numpy as np
from matplotlib import pyplot as plt
k1=0;
k2=0;
flag=1
archivo=input("Ingrese Nombre de archivo(datoejercicio10_1.txt): ")
f=open(archivo)
encabezado=f.read(56)
texto=f.read()
texto2=texto.replace("\t", "\n")
splitTexto=texto2.split("\n")
size=int(len(splitTexto))
size=int((size/2))
tension=np.empty(size)
corriente=np.empty(size)
potencia=np.empty(size)
for i in splitTexto:
    if len(i):
        if flag==1:
            tension[k1]=float(i)
            k1=k1+1
        elif flag==-1:
            corriente[k2]=float(i)
            k2=k2+1
        flag=-flag
potencia=tension*corriente
plt.plot(potencia)
plt.show()
input("FIN")
```

Resultado:

```
>>>  
= RESTART: C:/Users/Ivan PC/Desktop/Examen Domiciliario/Resolucion Examen Domici  
liario/Ejercicios Resueltos/Ejercicio10.py  
Ingrese Nombre de archivo(datoejercicio10_1.txt): datoejercicio10_1.txt
```



## Ejercicio 11- en python – utilice la librería numpy y matplotlib.

Se dispone de un archivo con el siguiente nombre.

- estacionmetereologica\_smn.txt

la primer columna corresponde a la fecha, la segunda la temperatura máxima, la tercera la temperatura mínima y la cuarta es el nombre de la estación meteorológica. Ver la siguiente figura.



FECHA	TMAX	TMIN	NOMBRE
01112020	21.3	15.3	AEROPARQUE AERO
01112020	25.5	5.6	AZUL AERO
01112020	28.2	5.8	BAHIA BLANCA AERO
01112020	28.1	4.8	BARILLOCHE AERO
01112020		-14.0	BASE BELGRANO II
01112020	3.9	-4.8	BASE CARLINI (EX JUBANY)
01112020	8.7	-3.6	BASE ESPERANZA
01112020	10.8	-2.9	BASE MARAMBIO
01112020	1.9	-6.6	BASE ORCADAS
01112020	7.6	-3.2	BASE SAN MARTIN
01112020			SENITO JUAREZ AERO

El número que corresponde a cada estación esta en el archivo.

- estaciones\_metereologicas.txt



1	AEROPARQUE AERO
2	AZUL AERO
3	BAHIA BLANCA AERO
4	BARILLOCHE AERO
5	BASE BELGRANO II
6	BASE CARLINI (EX JUBANY)
7	BASE ESPERANZA
8	BASE MARAMBIO
9	BASE ORCADAS

El usuario deberá ingresar el número de la estación y el programa deberá graficar la temperatura máxima y mínima para esa estación. En el título del grafico deberá indicarse el nombre de la estación meteorológica .

- No es necesario realizar el diagrama de flujo.

Variables utilizadas:

i int

indice int

Estaciones type

nombre list

fecha list

tMin list

Tmax list

nombre2 list

tempMax list

tempMin list

index int

archivo str

f io.TextIOWrapper'

textostr

texto1str

texto2str

splitTexto list

x list

dato list

encabezado str

D list

tM float  
tm float  
ran int  
min1 float  
max1 float

Funciones utilizadas:

int()  
input  
open()  
read()

Strip()  
replace  
Split  
append  
close()  
len  
float  
print()  
plot  
show

Programa desarrollado:

```
import numpy as np
from matplotlib import pyplot as plt
i=0
class Estaciones:
    indice=[]
    nombre=[]
class Temp:
    fecha=[]
    tMin=[]
    Tmax=[]
    nombre2=[]
class TempMaxMin:
    tempMax=[]
    tempMin=[]

index=int(input("Ingrese numero de la estacion: ")) -1
archivo="estaciones_meteorologicas.txt"
f=open(archivo)
texto=f.read()
textol = texto.strip()
texto2=textol.replace(" ", "")
splitTexto=texto2.split("\n")
for x in splitTexto:
    dato=x.split()
    Estaciones.indice.append(dato[0])
    Estaciones.nombre.append(dato[1:])
f.close()
archivo2="estacionmeteorologica_smn.txt"
f=open(archivo2)
encabezado=f.read(202)
texto=f.read()
textol = texto.strip()
splitTexto=textol.split("\n")
for x in splitTexto:
    dato=x.strip()
    dato=x.split()
    if len(dato)>3:
        Temp.fecha.append(dato[0])
        Temp.Tmax.append(dato[1])
        Temp.tMin.append(dato[2])
        Temp.nombre2.append(dato[3:])
```

Activar Windows  
Ve a Configuración

```

f.close()
D=Estaciones.nombre[index]
print(D)
for x in Temp.nombre2:
    i=i+1
    if x==D:
        tM=float(Temp.Tmax[i])
        TempMaxMin.tempMax.append(tM)
        tm=float(Temp.tMin[i])
        TempMaxMin.tempMin.append(tm)
        ran=len(TempMaxMin.tempMin)
min1=min(TempMaxMin.tempMin)
max1=max(TempMaxMin.tempMax)
print(min1)
print(max1)

plt.plot(1,min1,'o','b')
plt.plot(1,max1,'o','r')
plt.show()

```

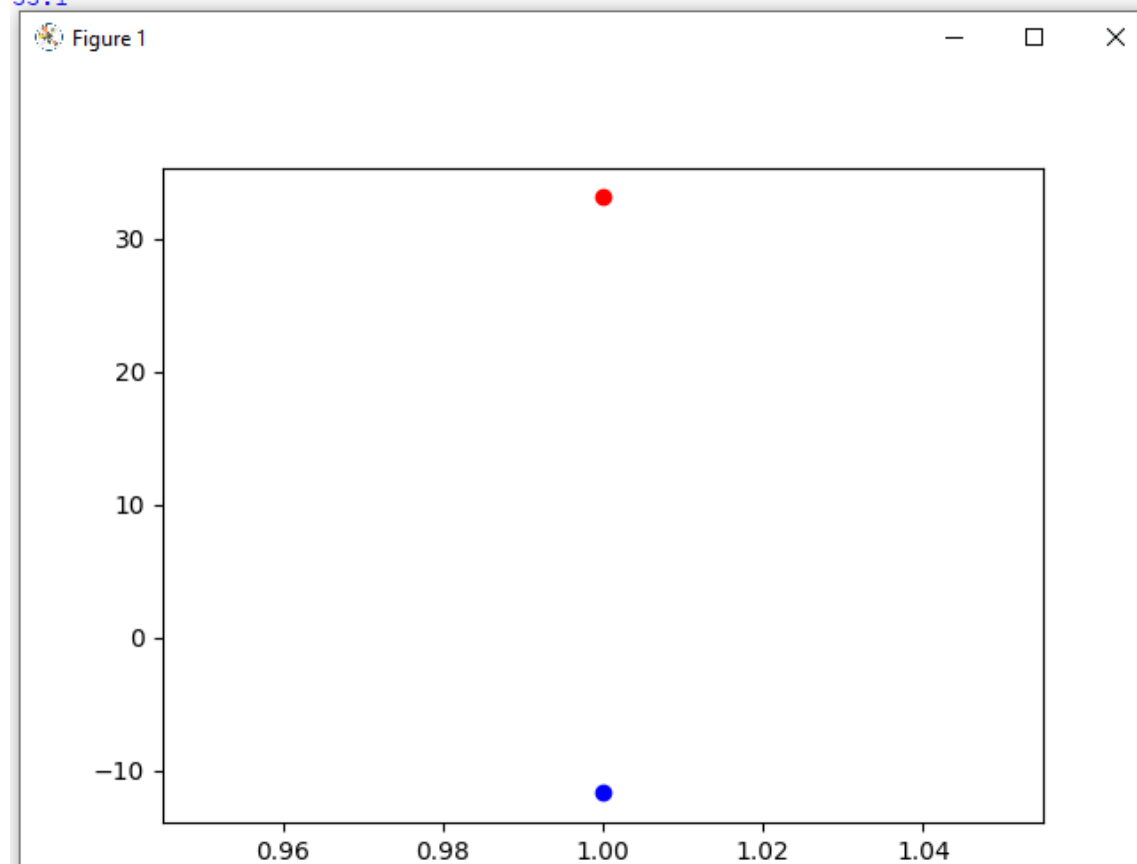
A

Resultado:

```

= RESTART: C:\Users\Ivan PC\Desktop\Examen Domiciliario\Resolucion Examen Domici
liario\Ejercicios Resueltos\Ejercicioll.py
Ingrese numero de la estacion: 3
['BAHIA', 'BLANCA', 'AERO']
-11.7
33.1

```





## Ejercicio 12- en python – utilice la librería numpy, matplotlib y pandas.

Se dispone del archivo datoEjercicio12.csv formada por las siguientes columnas (Fecha, Temperatura, Precipitaciones, Velocidad del viento y dirección del viento). Cada fila corresponde a una medición.

Observaciones: la fecha tiene el siguiente formato: 20201110T0300 (los primeros cuatro caracteres corresponde al año, los dos siguientes el mes, el siguiente la letra “T”, los dos siguientes caracteres la hora y los dos últimos los minutos).

El programa deberá graficar la Temperatura, Precipitaciones, Velocidad del viento y dirección del viento en función del número de registro.

Indicar la temperatura y precipitaciones máxima del periodo.

La cantidad de horas entre la temperatura máxima y mínima para cada día. Por ejemplo para el día 11 de noviembre de 2020 la temperatura mínima se produce a las 8 hs y la máxima a las 15hs. Es decir el programa nos deberá entregar el valor de la diferencia entre 15 y 8 como así también graficar esta magnitud en función del número de día.

```
20201110T0000,10.240529,0.0,6.6087217,119.35774
20201110T0100,9.680529,0.0,6.792466,122.00538
20201110T0200,9.490529,0.0,7.10031,120.465546
20201110T0300,9.210529,0.0,7.10031,120.465546
20201110T0400,8.820529,0.0,7.289445,122.90524
20201110T0500,8.5505295,0.0,7.5942082,121.429565
20201110T0600,8.340529,0.0,7.9036193,120.06859
20201110T0700,7.7105284,0.0,8.707237,119.74400
20201110T0800,7.3005285,0.0,8.534353,117.64597
20201110T0900,8.230529,0.0,7.5685663,115.34817
20201110T1000,10.830529,0.0,9.0,106.2602
20201110T1100,12.820529,0.0,7.9932976,97.76516
20201110T1200,14.570529,0.0,6.8399997,89.99999
20201110T1300,14.8005295,0.0,1.8,360.0
20201110T1400,15.690529,0.0,3.219938,233.43417
20201110T1500,16.110529,0.0,4.3498964,335.55603
20201110T1600,15.890529,0.0,4.6800003,337.38013
20201110T1700,15.160529,0.0,4.6800003,337.38013
20201110T1800,14.440529,0.0,4.6938257,327.5288
20201110T1900,13.470529,0.0,4.0249224,296.56506
20201110T2000,13.0505295,0.0,5.0528407,265.91437
20201110T2100,12.700529,0.0,6.379216,253.61046
20201110T2200,11.920529,0.0,6.763786,244.79889
20201110T2300,11.270529,0.0,5.6002855,225.0
```

Temperatura  
Mínima

Temperatura  
Máxima

### Variables utilizadas:

i int

Fechas type

cFechas list

Choras list

cTemp list

cTempDiferencia list

tabla pandas.core.frame.DataFrame

tablatimestamp pandas.core.series.Series

temperatureTabla pandas.core.series.Series

precipiteTabla pandas.core.series.Series

windTabla pandas.core.series.Series

n datetime.datetime

fecha str

tfecha datetime.datetime

a datetime.datetime

dia int  
auxdia int  
hora int  
min0 numpy.float64  
max0 numpy.float64  
hMax int  
hMin int  
hDiferencia0 int  
hDiferencia numpy.int32  
min1 float  
max1 float

**Funciones utilizadas:**

pd.read\_csv  
for  
dt.datetime.strptime()  
append()  
if()  
min()  
max()  
np.absolute  
print()  
clear()  
plt.subplot()  
plt.plot()  
plt.title()  
plt.show()  
input()

## Programa desarrollado:

---

```
import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
import datetime as dt

i=0 #contador choras,ctemp

class Fechas: #listas
    cFechas=[] # las fechas .strptime
    cHoras=[] #almacena temporalmente las horas de un dia para buscar min y max
    cTemp=[] #almacena las temperaturas temporalmente por dia
    cTempDiferencia=[] # guarda la diferencia de horas

tabla = pd.read_csv('datoejercicio12.csv')
#separa en columnas
tablatimestamp=tabla['timestamp']
temperatureTabla=tabla['Basilea Temperature [2 m elevation corrected]']
precipiteTabla=tabla['Basilea Precipitation Total']
windTabla=tabla['Basilea Wind Speed [10 m]']

#strptime fechas en Fechas.cFechas
for n in tablatimestamp:
    fecha=n
    tfecha=dt.datetime.strptime(fecha, '%Y%m%dT%H%M')
    Fechas.cFechas.append(tfecha)

a=Fechas.cFechas[0] #inicializa a
dia=a.day #valor inicio dia
for n in Fechas.cFechas: #buscar max y min

    auxdia=dia #valor auxiliar para comparar dia
    dia=n.day
```

```

hora=n.hour
if dia==auxdia:
    Fechas.cHoras.append(hora)
    Fechas.cTemp.append(temperatureTabla[i])
    Fechas.cHoras.append(temperatureTabla[i])
    min0=min(Fechas.cTemp)
    max0=max(Fechas.cTemp)

else:
    hMax= Fechas.cTemp.index(max0)
    hMin= Fechas.cTemp.index(min0)
    hDiferencia0=hMax-hMin
    hDiferencia=np.absolute(hDiferencia0)
    Fechas.cTempDiferencia.append(hDiferencia)
    print("dia: ",dia)
    print("min: ",min0," hora: ", hMin)
    print("max: ",max0," hora: ", hMax)
    print("Diferencia de horas: ", hDiferencia)
    Fechas.cHoras.clear()
    Fechas.cTemp.clear()
    print("_____")
    i=i+1
#imprime temperatura max y min del periodo
min1=min(temperatureTabla)
max1=max(temperatureTabla)
print("maxPeriodo: ",max1)
print("minPeriodo: ",min1)

#graficos
plt.subplot(2, 2, 1)
plt.plot(Fechas.cTempDiferencia)
plt.title("Diferencias de horas min y max")

plt.subplot(2, 2, 2)
plt.plot(temperatureTabla,'r',label="temperatureTabla")
plt.title("temperatureTabla")

plt.subplot(2, 2, 3)
plt.plot(precipiteTabla,'g')
plt.title("precipiteTabla")

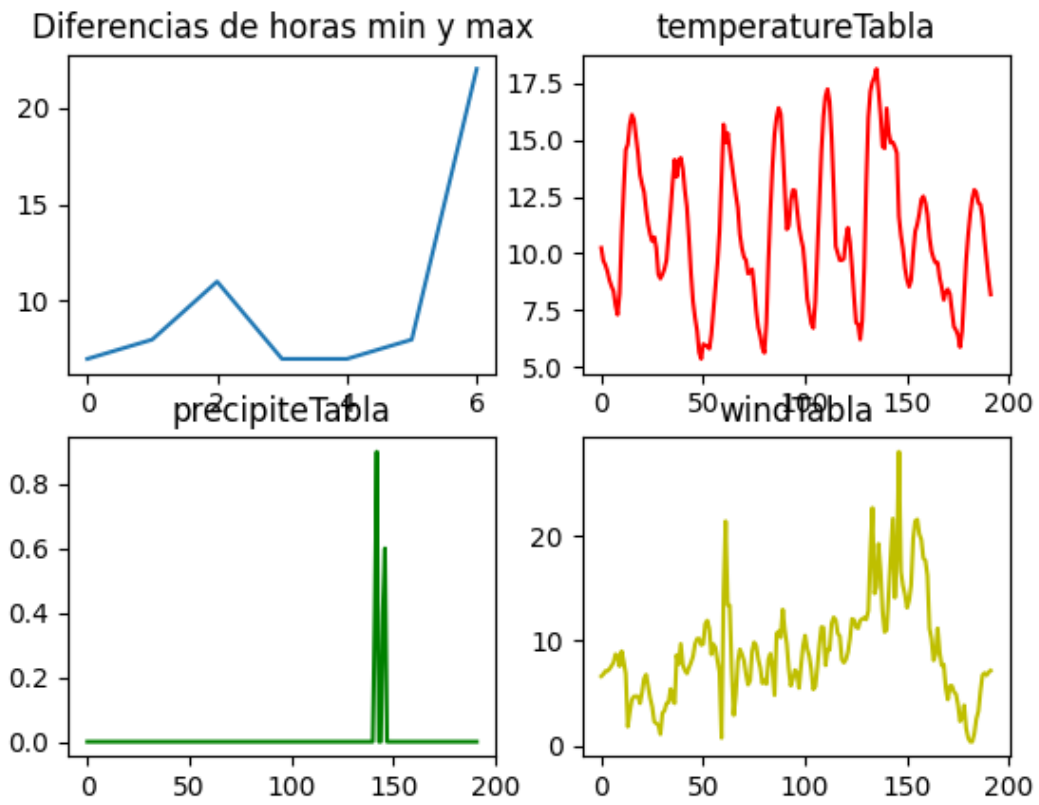
plt.subplot(2, 2, 4)
plt.plot(windTabla,'y')
plt.title("windTabla")

plt.show()

input("FIN")

```

Resultado:



```
>>>
= RESTART: C:\Users\Ivan PC\Desktop\Examen Domiciliario\Resolucion Examen Domicil
iario\Ejercicios Resueltos\Ejercicio12.py
dia: 11
min: 7.3005285 hora: 8
max: 16.110529 hora: 15
Diferencia de horas: 7

dia: 12
min: 6.6005287 hora: 22
max: 14.210529000000001 hora: 14
Diferencia de horas: 8

dia: 13
min: 5.350528700000001 hora: 0
max: 15.680529 hora: 11
Diferencia de horas: 11

dia: 14
min: 5.6205287 hora: 7
max: 16.410528 hora: 14
Diferencia de horas: 7

dia: 15
min: 6.7105284 hora: 7
max: 17.240528 hora: 14
Diferencia de horas: 7

dia: 16
min: 6.2105284 hora: 6
max: 18.13053 hora: 14
Diferencia de horas: 8

dia: 17
min: 8.510529 hora: 22
max: 14.420529 hora: 0
Diferencia de horas: 22

maxPeriodo: 18.13053
minPeriodo: 5.350528700000001
FIN
```