

LABORATORIO DE COMUNICACIONES

Laboratorio 2 (fase 2)

Conquista de gnuradio a nivel de programación

Prácticas de programación en Python

Practica 2

Autores:	Carolina Viasus Africano
	Davidson Duvan Leal
	Alex Julian Mantilla Rios
Perteneciente al grupo:	B1A.G6

[Aspectos a mejorar en la guía](#)

[El Problema:](#)

[El objetivo general es:](#)

[Preparativos](#)

[Apuntes de python para nunca olvidar](#)

[Generalidades](#)

[El formato para definir una Función](#)

[El formato para definir una clase](#)

[El formato crear un objeto](#)

[Ejemplo para usar una función sola](#)

[Ejemplo de uso de funciones de una clase](#)

[Objetivos específicos](#)

[Referencias usadas](#)

[Informe de resultados](#)

[Desarrollo del Objetivo 1. Presente a continuación los resultados del objetivo 1.](#)

[Desarrollo del Objetivo 2. Presente a continuación los resultados del objetivo 2.](#)

[Desarrollo del Objetivo 3. Presente a continuación los resultados del objetivo 3.](#)

[Desarrollo del Objetivo 4. Presente a continuación los resultados del objetivo 4.](#)

[Desarrollo del Objetivo 5. Presente a continuación los resultados del objetivo 5.](#)

Aspectos a mejorar en la guía

Los siguientes son apuntes del profesor para introducir mejoras a futuras prácticas:

- Por ahora no hay apuntes

El Problema:

Por ahora el problema a resolver consiste en que el estudiante no tiene las suficientes bases de programación por objetos en Python para pasar a realizar desarrollos usando GNU Radio y herramientas profesionales. En el Lab1 se realizaron prácticas sobre las bases en que se sustenta python y además sobre GITHUB. Debido a que los estudiantes traen buena bases de Matlab, el Lab1 se aprovechó para demostrar que python puede ser usado como Matlab es aspectos como: programación por consola; operaciones vectoriales y matriciales; programación usando archivos. El reto ahora es pasar directamente a la programación orientada a objetos, mientras al mismo tiempo se practican conceptos vistos en clase como: variable aleatoria, promedios de tiempo, procesos estocásticos. También

aprovecharemos para conquistar una nueva herramienta profesional de programación en Python: los Entornos Virtuales.

El objetivo general es:

Usar programación por objetos en Python y los Entornos virtuales para producir una calculadora científica vectorial.

Preparativos

- Baje una version actualizada del: [el libro de la asignatura](https://sites.google.com/saber.uis.edu.co/comdig/sw). Observe que en los capítulos del libro ofrecen enlaces a código de software, a flujogramas y otros recursos que son parte del libro. Por ejemplo, observa que debajo de cada gráfica con flujogramas hay una nota que dice: “Flujograma usado”. Esos recursos usados en el libro están en la página del libro: <https://sites.google.com/saber.uis.edu.co/comdig/sw>

Apuntes de python para nunca olvidar

Generalidades

- Python es un lenguaje interpretado. Eso significa que un programa escrito en python no debe ser convertido previamente a código binario para ponerlo en funcionamiento.
- Python es un lenguaje indentado. Esto significa que cualquier espacio que se deje antes de una línea de código es un mensaje clave, es decir tiene un sentido particular.
- En Python los tipos de las variables se deduce de manera automática, por ejemplo:
 - Si se escribe `x=0.4` el lenguaje decide que `x` es una variable de tipo flotante
 - Si se escribe `x=4` el lenguaje decide que `x` es una variable de tipo entero
 - Si se escribe `x=[1,2,d,2,4,fuerte,9]` el lenguaje decide que `x` es una variable de tipo lista flotante
 - para trabajar con vectores se debe usar una librería llamada numpy:
 - `import numpy as np` # para importar la librería
 - `x=np.array([1,2,3,4,5])` # crea un vector
 - `x=np.linspace(2.0, 3.0, num=5)` # es otro ejemplo para crear un vector
 - ver más ejemplos de trabajo con vectores en: [Manual de manuales](#), sección “Python para desmemoriados”
- Para programar más fácilmente siga estos consejos:
 - los nombres de los objetos haga que comienzan con letra mayúscula para distinguirlos mejor de otras cosas como las funciones.
 - use la programación por consola cuando no se sienta seguro de un comando o secuencia de comandos. La idea es: no use en su archivo final comandos que no conoce del todo, experimentarlos primero por consola hasta asegurarse que eso funciona como Ud espera.
 - Use la programación en un archivo para los productos finales.

- Use entornos virtuales para tener un mayor dominio sobre sus propios desarrollos
- Use GITHUB para conservar historial de versiones de su código y para aprender a trabajar en equipo.

comando de github	explicación
git clone URL	para clonar su repositorio. Desde su cuenta copia la URL del repositorio; en terminal Ubuntu envía el comando; en tu computador aparecerá la carpeta del repositorio
git add .	para agregar nuevo contenido a una lista
git commit -m "comentario"	para pasar el nuevo contenido a base local
git push	para que el contenido suba al repositorio en la nube

El formato para definir una Función

```
def funcion1(self, variable1, variable2):
    y=variable1+variable2
    return y
```

El formato para definir una clase

```
class A:
    def __init__(self):
        print 'me ejecuto primero'

    def funcionb(self):
        print 'soy una funcion!!!'
```

El formato crear un objeto

```
miobjeto = A()
```

Ejemplo para usar una función sola

```
funcion1(4,5))
```

Ejemplo de uso de funciones de una clase

```
miobjeto = A()  
miobjeto.funcionb()
```

Objetivos específicos

1. Aprenda a crear funciones en Python y use todo lo aprendido para crear de la manera más profesional posible las siguientes funciones con variables vectoriales:
 - a. crear la función que suma dos números: suma(x,y)
 - b. crear la función que resta dos números: resta(x,y)
2. Aprenda a crear objetos en Python y use todo lo aprendido para crear de la manera más profesional posible lo siguiente
 - a. calculadora basica: para dos variables x,y puede calcular diferentes operaciones básicas como. suma, resta, división, multiplicación
3. Investigue son “herencias” en python y cree una calculadora científica que hereda lo que hace la calculadora básica pero además realiza los cálculos esenciales de “variable Aleatoria” que se explican en [2], capítulo 1.5.2
4. Explique si puede usar la misma calculadora científica para realizar cálculos de “Promedios de tiempo”. En todo caso implemente un ejemplo para obtener promedios de tiempo de una señal.
5. Explique qué haría si en vez “Promedios de tiempo” el tema son los procesos estocásticos.

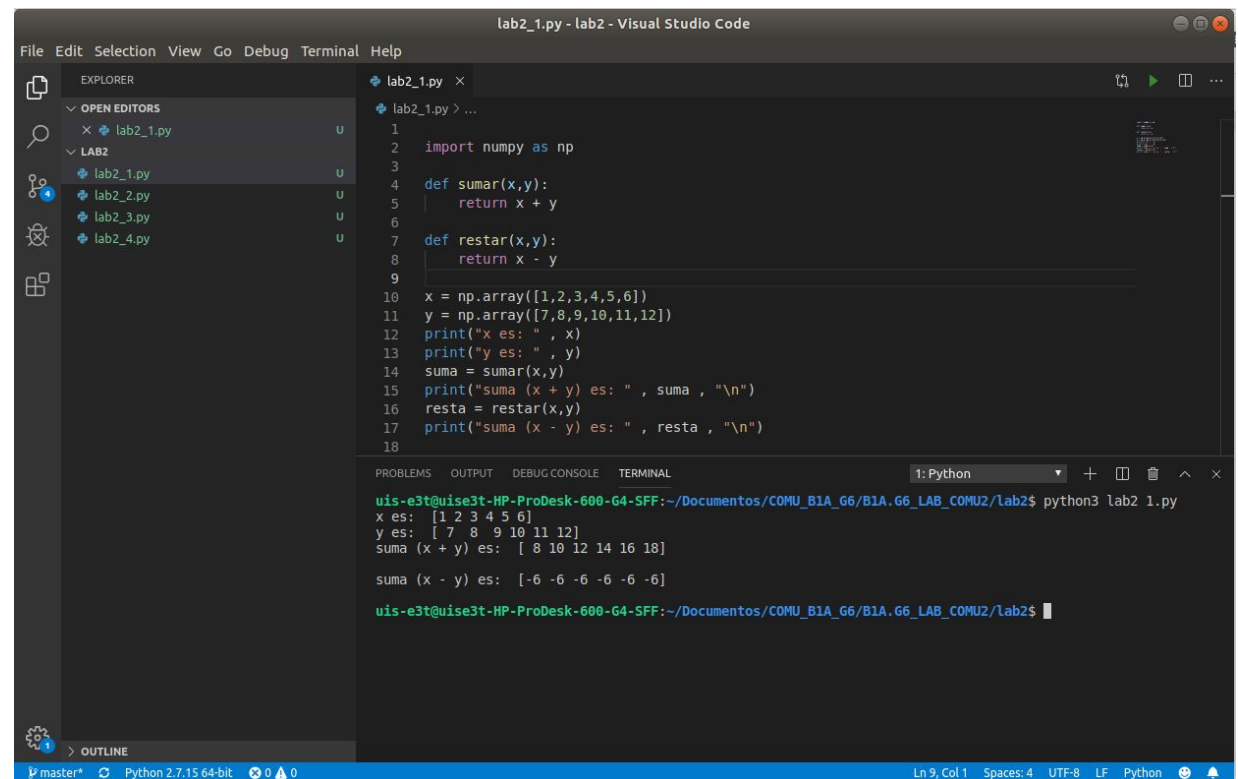
Referencias usadas

- [1] [Manual de manuales](#)
- [2] [El libro de la asignatura](#)
- [3] [Página del libro](#)
- [4] [Libro Internet de los Objetos](#)

Informe de resultados

Desarrollo del Objetivo 1. Presente a continuación los resultados del objetivo 1.

Suma de vectores:

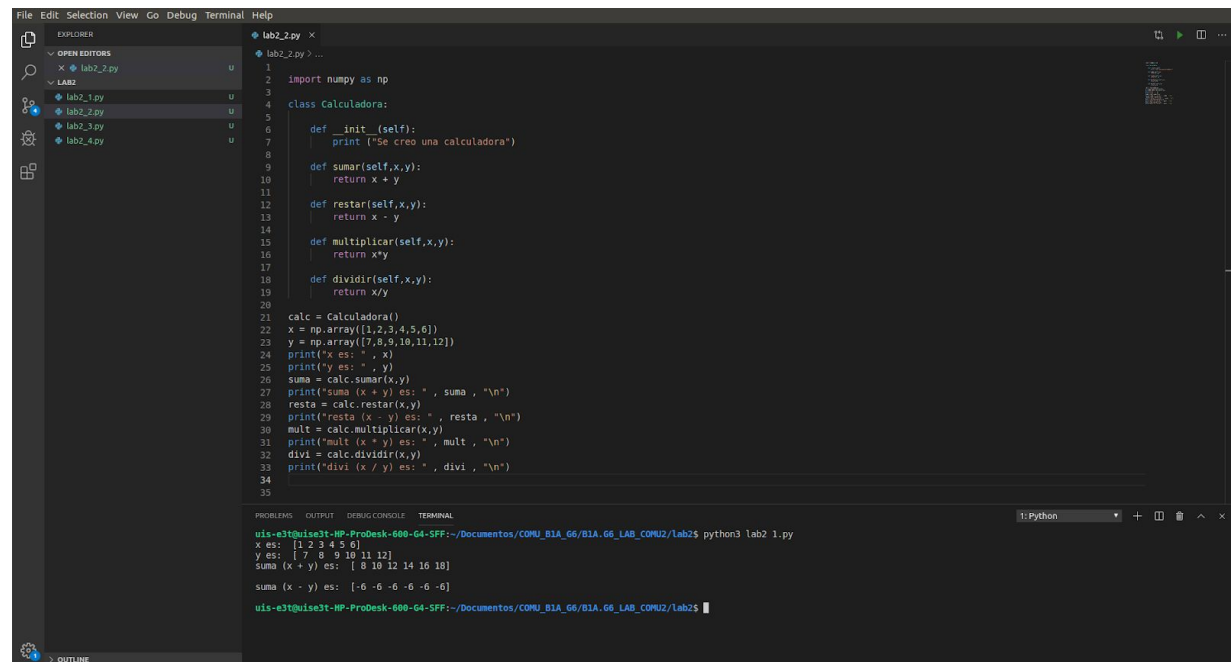


```
lab2_1.py - lab2 - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
EXPLORER
  OPEN EDITORS
    lab2_1.py
  LAB2
    lab2_1.py
    lab2_2.py
    lab2_3.py
    lab2_4.py
lab2_1.py
1
2 import numpy as np
3
4 def sumar(x,y):
5     return x + y
6
7 def restar(x,y):
8     return x - y
9
10 x = np.array([1,2,3,4,5,6])
11 y = np.array([7,8,9,10,11,12])
12 print("x es: ", x)
13 print("y es: ", y)
14 suma = sumar(x,y)
15 print("suma (x + y) es: ", suma , "\n")
16 resta = restar(x,y)
17 print("suma (x - y) es: ", resta , "\n")
18
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A.G6_LAB_COMU2/Lab2$ python3 lab2 1.py
x es: [ 1  2  3  4  5  6]
y es: [ 7  8  9 10 11 12]
suma (x + y) es: [ 8 10 12 14 16 18]

suma (x - y) es: [-6 -6 -6 -6 -6 -6]
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A.G6_LAB_COMU2/Lab2$
```

Desarrollo del Objetivo 2. Presente a continuación los resultados del objetivo 2.

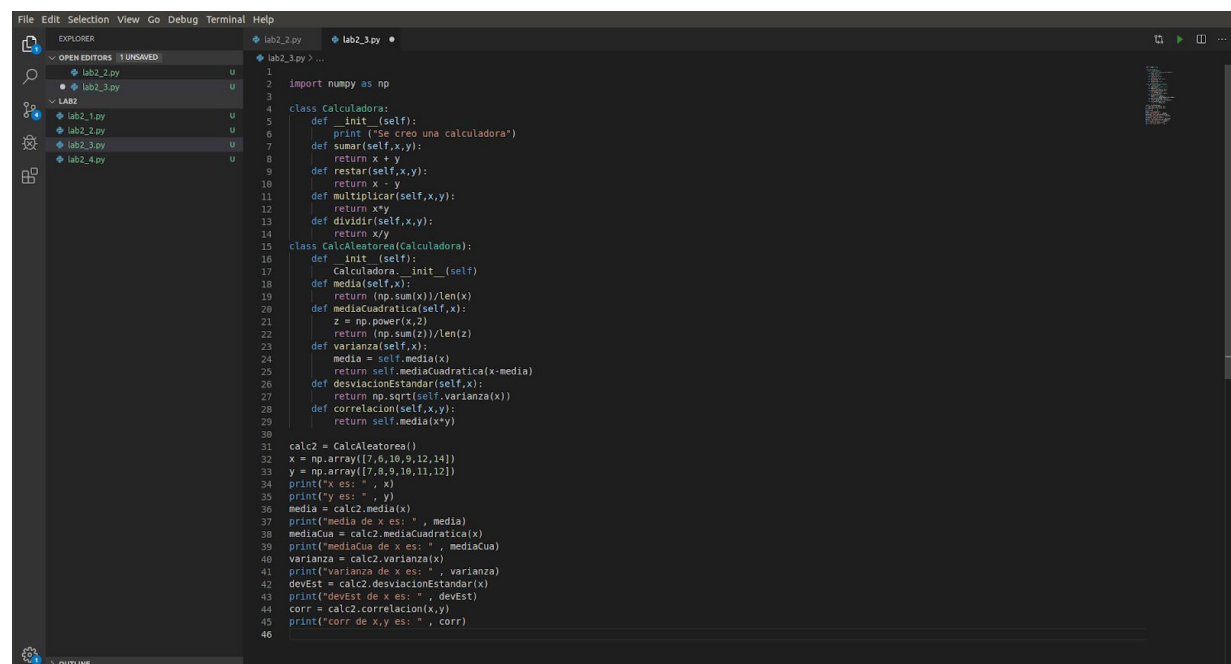
Calculadora Básica:



```
File Edit Selection View Go Debug Terminal Help
EXPLORER
  OPEN EDITORS
    lab2_2.py
  LAB2
    lab2_1.py
    lab2_2.py
    lab2_3.py
    lab2_4.py
  lab2_2.py x
lab2_2.py > ...
1
2 import numpy as np
3
4 class Calculadora:
5     def __init__(self):
6         print("Se creo una calculadora")
7
8     def sumar(self,x,y):
9         return x + y
10
11     def restar(self,x,y):
12         return x - y
13
14     def multiplicar(self,x,y):
15         return x*y
16
17     def dividir(self,x,y):
18         return x/y
19
20 calc = Calculadora()
21 x = np.array([1,2,3,4,5,6])
22 y = np.array([7,8,9,10,11,12])
23 print("x es: ", x)
24 print("y es: ", y)
25 suma = calc.sumar(x,y)
26 print("suma (x + y) es: ", suma , "\n")
27 resta = calc.restar(x,y)
28 print("resta (x - y) es: ", resta , "\n")
29 mult = calc.multiplicar(x,y)
30 print("mult (x * y) es: ", mult , "\n")
31 divi = calc.dividir(x,y)
32 print("divi (x / y) es: ", divi , "\n")
33
34
35
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/lab2$ python3 lab2 1.py
x es: [1 2 3 4 5 6]
y es: [7 8 9 10 11 12]
suma (x + y) es: [ 8 10 12 14 16 18]
suma (x - y) es: [-6 -6 -6 -6 -6 -6]
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/lab2$
```

Desarrollo del Objetivo 3. Presente a continuación los resultados del objetivo 3.

Calculadora de datos Aleatorios:



```
File Edit Selection View Go Debug Terminal Help
EXPLORER
  OPEN EDITORS
    lab2_2.py
    lab2_3.py
  LAB2
    lab2_1.py
    lab2_2.py
    lab2_3.py
    lab2_4.py
  lab2_3.py
lab2_3.py > ...
1
2 import numpy as np
3
4 class Calculadora:
5     def __init__(self):
6         print("Se creo una calculadora")
7
8     def sumar(self,x,y):
9         return x + y
10
11     def restar(self,x,y):
12         return x - y
13
14     def multiplicar(self,x,y):
15         return x*y
16
17     def dividir(self,x,y):
18         return x/y
19
20 class CalcAleatoria(Calculadora):
21     def __init__(self):
22         Calculadora.__init__(self)
23
24     def media(self,x):
25         return (np.sum(x))/len(x)
26
27     def mediaCuadratica(self,x):
28         z = np.power(x,2)
29         return (np.sum(z))/len(z)
30
31     def varianza(self,x):
32         media = self.media(x)
33         return self.mediaCuadratica(x-media)
34
35     def desviacionEstandar(self,x):
36         return np.sqrt(self.varianza(x))
37
38     def correlacion(self,x,y):
39         return self.media(x*y)
40
41 calc2 = CalcAleatoria()
42 x = np.array([7,6,10,9,12,14])
43 y = np.array([7,8,9,10,11,12])
44 print("x es: ", x)
45 print("y es: ", y)
46 media = calc2.media(x)
47 print("media de x es: ", media)
48 mediaCua = calc2.mediaCuadratica(x)
49 print("mediaCua de x es: ", mediaCua)
50 varianza = calc2.varianza(x)
51 print("varianza de x es: ", varianza)
52 devEst = calc2.desviacionEstandar(x)
53 print("devEst de x es: ", devEst)
54 corr = calc2.correlacion(x,y)
55 print("corr de x,y es: ", corr)
56
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/Lab2$ python3 lab2_3.py
Se creo una calculadora
x es: [ 7  6 10  9 12 14]
y es: [ 7  8  9 10 11 12]
media de x es: 9.666666666666667
mediaCua de x es: 101.0
varianza de x es: 7.5555555555555554
devEst de x es: 2.748737083745107
corr de x,y es: 96.16666666666667
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/Lab2$
```

Desarrollo del Objetivo 4. Presente a continuación los resultados del objetivo 4.

Si, porque el cálculo numérico de la media de una señal en el tiempo es igual a la media de la señal discreta con suficiente número de muestras:

```
File Edit Selection View Go Debug Terminal Help
EXPLORER
lab2_2.py
lab2_3.py
lab2_4.py
LAB2
lab2_1.py
lab2_2.py
lab2_3.py
lab2_4.py
lab2_4.py ...
1
2 import numpy as np
3
4 class Calculadora:
5     def __init__(self):
6         print("Se creo una calculadora")
7     def sumar(self,x,y):
8         return x + y
9     def restar(self,x,y):
10        return x - y
11    def multiplicar(self,x,y):
12        return x*y
13    def dividir(self,x,y):
14        return x/y
15
16 class CalcAleatorios(Calculadora):
17     def __init__(self):
18         Calculadora.__init__(self)
19     def media(self,x):
20        return (np.sum(x))/len(x)
21    def mediaCuadratica(self,x):
22        z = np.power(x,2)
23        return (np.sum(z))/len(z)
24    def varianza(self,x):
25        media = self.media(x)
26        return self.mediaCuadratica(x-media)
27    def desviacionEstandar(self,x):
28        return np.sqrt(self.varianza(x))
29    def correlacion(self,x,y):
30        return self.media(x*y)
31
32 calc2 = CalcAleatorios()
33 t = np.linspace(0,20,200000)
34 y = np.exp(-0.5*t)
35 media = calc2.media(y)
36 print("media de y es: ", media)
37
38
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/Lab2$ python3 lab2_4.py
Se creo una calculadora
media de y es: 0.8999956882885287
uis-e3t@uise3t-HP-ProDesk-600-G4-SFF:~/Documentos/COMU_B1A_G6/B1A_G6_LAB_COMU2/Lab2$
```

Desarrollo del Objetivo 5. Presente a continuación los resultados del objetivo 5.

Los procesos estocásticos no son muy ajenos a los promedios de tiempo, ya que un grupo de señales diferentes en el tiempo, analizadas por separado arrojaran resultados resultados que entre ellos son aleatorios, y por ende, pueden ser analizados con las herramientas de los procesos estocásticos, tales como las trabajadas en esta guía.