Práctica 3

Conquista de gnuradio a nivel de programación

Prácticas de programación en Python

Autores:

Lugo Vásquez Vanny Sofia - 2154709
Angarita Pertuz Jose David - 2142992
Cancino Rey William Andrés - 2151534

Grupo:
B1A.g2

Aspectos a mejorar en la guia

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

- Por ahora no hay apuntes

Enlace a materiales de apoyo

- Manual de manuales
- El libro de la asignatura
- Página del libro

El Problema:

Esta es una continuación de la práctica anterior, por lo tanto el problema es el mismo.
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.

El objetivo general es:

Retar al estudiante a construir su propio conocimiento para programar por objetos usando python y herramientas profesionales como Github y Visual Studio Code.

Preparativos

- Baje una version actualizada del libro para concentrarse en el capítulo 1.5 y 1.6 donde está la teoría necesaria. Tenga en cuenta que: en el libro de la asignatura. 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 interés

Objetivos específicos

Nota: para todos los casos use Python3

1. Lea el tutorial python básico en “programación orientada a objetos” y explique:
   a. cual es la diferencia entre clases y objetos
   b. que es instanciar
   c. qué es el constructor. Es decir, para qué sirve la función “__init__”
   d. Implemente en un archivo de “Visual Studio Code” el primer ejemplo que aparece en el tutorial y haga unas pruebas que le permitan tener claro para qué sirve y cómo se usa cada cosa (la clase, el constructor, las funciones, las instancia). Use las facilidades que brinda “Visual Studio Code” para observar paso a paso cada una de las sentencias que se van ejecutando.
   e. Explique cómo se implementa una herencia.
2. Programación orientada a GNU Radio. En GU Radio lo que se practica es la programación en tiempo real. revise este concepto en el libro, capítulo 1.6.17 y responda las siguientes preguntas:
   a. ¿Qué es en GNU Radio una señal tipo stream?, ¿Qué características tiene?
   b. ¿Qué modificaciones introduciría usted a las funciones ya implementadas para lograr obtener promedios de tiempo a señales tipo stream?. Mejor dicho el problema ahora es que usted tiene una señal, que ni siquiera ha llegado completamente, va llegando por trozos de N1, N2, N3, ..., muestras. Supongamos que tiene que hallar la media, pero hay que entregar un valor actualizado después de recibir cada trozo de la señal. Es como ocurre con el COVID-19, cada día llega un paquete de nuevos casos, pero justo cada día hay que actualizar los datos teniendo en cuenta los resultados ya obtenidos el día anterior y ajustándolos a los nuevos casos.
3. Revise cómo se soluciona el problema con el código que se presenta en el libro, capitulo 1.6.17.1 para el bloque e_mean_meter_ff. Responda estas preguntas:
   a. ¿qué es un bloque de tipo sync?
   b. qué concepto de los vistos se usa al definir “class mean_meter (gr. sync_block):”?
   c. ¿Qué concepto de los vistos se usa al definir “__init__ (self):”
   d. ¿Qué concepto de los vistos se usa al definir “work (self , input_items , output_items):”?

4. Implemente en GNU Radio el código anterior mediante programación embebida en un “Python Block”. Use el manual de los manuales para encontrar ejemplos para este tipo de programación.

5. Uso de una herramienta profesional como Github para guardar los trabajos realizados. Suscribase en www.github.com siguiendo las siguientes pautas:
   a. la inscripción es personal, es decir, cada persona debe crear su cuenta
   b. crear un repositorio por grupo. Es decir, solo una de las personas del grupo crea un repositorio y lo comparte con los demás. Nota: cuando se le pregunte, dele la opción de incluir README.
   c. El nombre del repositorio coincide con el nombre del grupo
   d. Desde terminal de Ubuntu use los comandos de Github que se muestran en la siguiente tabla (Nota: Encuentre más detalles en Manual de manuales, sección “Manual de Github”):

comando	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

Informe de resultados

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

a. Una clase es una plantilla para la creación de objetos según un modelo previamente definido. Cada clase es un modelo que define un conjunto de variables y métodos para operar con datos.
Los objetos son entidades que tienen atributos, métodos e identidad.
Los atributos son las características que posee dicho objeto, que en programación se refiere a las variables. Por otro lado, los métodos son las acciones que puede realizar el objeto. Finalmente, la identidad es la propiedad de un objeto que lo diferencia del resto.

b. La creación de un objeto a partir de una clase se denomina instancia, y permite plasmar las características que tendrá dicho objeto.

c. La función "__init__" tiene como objetivo establecer un estado inicial en el objeto, principalmente, inicializar o declarar atributos (variables).

d. Partiendo de que una clase es una especie de plantilla que sirve para crear objetos, en este caso la clase es "Laboratorio", se define una serie de atributos (variables) y métodos. Los métodos son como funciones en python, con la única diferencia de que deben recibir por lo menos un parámetro, denominado "self".
El primer método que todas las clases deben tener es el constructor. Este define la forma en la que se crean los objetos de datos, en python el método constructor se llama "__init__", según se muestra en Figura 1.

```
1  class Laboratorio:#clase
2
3      #Métodos
4      def _init_(self,nota,aprendizaje):
5          self.nota = nota
6          self.aprendizaje= aprendizaje
7          print ('Formando ingeniera')
8
9      def Profesor(self,homeroortega):
10         self.homeroortega= homeroortega
11         print ('Homero Ortega')
12
13
14     def Lugar(self,lugar):
15         self.lugar= lugar
16         print ('Aula virtual')
17
18 comunicaciones = Laboratorio #Objeto
19 print(comunicaciones.Profesor)
20 print(comunicaciones.Lugar)
```

Figura 1. Implementación de POO.

e. La herencia se basa en que una clase diferente puede tener métodos que son parte de otra clase diferente. La herencia sirve para reutilizar código en caso de crear objetos similares.
Por ejemplo, se crea una clase de la siguiente manera: "**class Profesor:**". Dicha clase contiene una métodos o funciones. Sin embargo, cuando se crea otra clase denominada "Estudiante", con sus propias funciones, esta puede heredar los métodos propios de la clase "Profesor" recibiendo como parámetro el nombre de dicha clase, como se describe a continuación: "**class Estudiante(Profesor):**".

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

a. GNU Radio está diseñado para el procesamiento de señales en tiempo real. Sin embargo, dichas señales pueden ser bastante extensas provocando complicaciones en el procesamiento de las mismas, por ejemplo, por la falta de almacenamiento. Por lo tanto, una solución es tomar segmentos de dicha señal, que deben tener una longitud corta y adecuada para que la información sea coherente y fácilmente procesable, generando una aparente operación en tiempo real.
La señales stream básicamente se transfieren de un bloque a otro, dando la sensación de que ninguno de ellos debe realizar la tarea de empaquetado o la definición del tamaño para dichos paquetes. Estas señales llegan a los bloques en forma de vectores o "arrays" y presentan dos características muy importantes:

1. Los vectores que las conforman son unidimensionales.
2. El tamaño de los vectores es variable.

b. Para dar solución al problema se deben considerar seis variables importantes: entrada "in", salida "out", almacenador de tamaño "N", contador "cont", acumulador "acum" y sumatoria "sum". Básicamente, el sistema está diseñado para recibir vectores unidimensionales los cuales se alojan en "in". Una vez se encuentra la información allí, mediante una función, se procede a conocer la longitud de dicho vector, este valor se almacena en "N". Ahora, la variable "cont", la cual se inicializa en cero, guarda el valor actual de "N" más el valor anterior de él mismo (cont = N + cont), permitiendo actualizar constantemente la longitud de la señal. Seguidamente, la variable "sum", almacena el resultado de utilizar la función de sumatoria de todos los elementos que contienen el vector actual. Posteriormente, "acum", el cual se inicializa en cero, almacena el valor actual de "sum" más el acumulado de las sumatorias anteriores (acum = sum + acum). Una vez se tienen los valores de "N" y "acum", se puede realizar el cálculo de los promedios de tiempo, específicamente la media, la cual se aloja en "out" y se describe como:

out = acum/N

Este proceso se repite para cada uno de los vectores entrantes, y en "out" se va presentando la actualización constante de la media de dicha señal.

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

a. Un bloque “sync” es aquel que es de tipo síncrono, es decir, para cada muestra de entrada existe una muestra de salida, o lo que es equivalente, consume tantas muestras como las que produce.

b. En “class mean_meter(gr.sync_block)” dicha clase está “heredando” atributos de “gr.sync_block”.

c. La sentencia “__init__(self)” es un método de las clases, o un constructor, que permite en este caso, declarar las variables que se utilizan dentro de “mean_meter”.

d. La sentencia “work(self, input_items, output_items)” es uno de los métodos o funciones de la clase “mean_meter”. Dicho método siempre está presente en un bloque y es el que contiene la lógica del mismo. Además, este método recibe dos atributos: “input_items” y “output_items”.

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

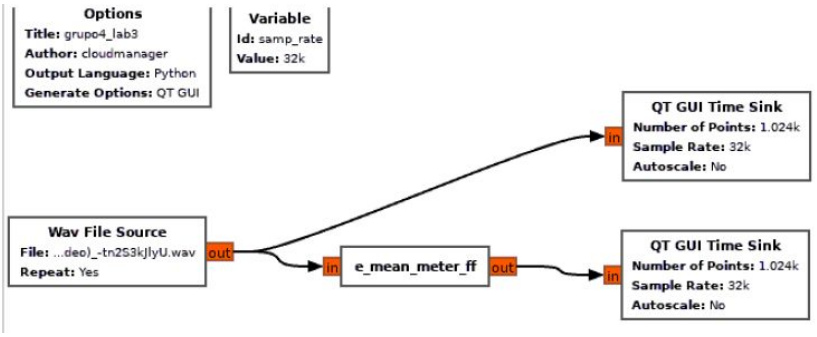

Figura 2. Flujograma para visualizar el funcionamiento del “python block”.

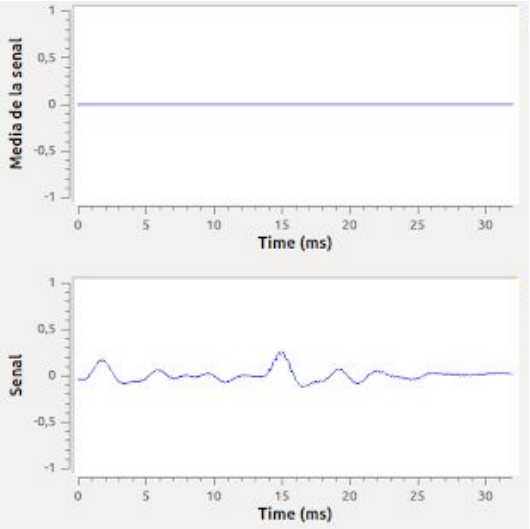

Figura 3. Señal de voz y su media.

Se presentaron algunos inconvenientes al momento de implementar el código, debido a una identación incorrecta. Sin embargo, en la ventana de ejecución de GNU Radio se indica la linea del codigo y/o el posible error que se está cometiendo.
Un una observación importante es que la media de esta señal es cero.

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

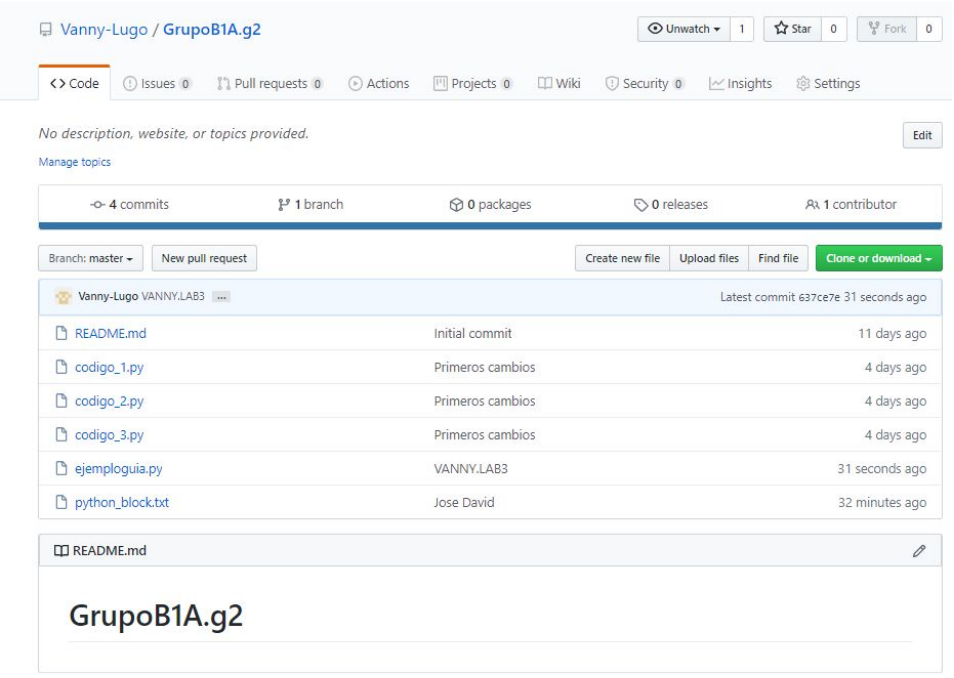

Figura 4. Repositorio en Github.

Se procede a utilizar el terminal de ubuntu para modificar el repositorio en Github. Principalmente, se clona el repositorio en la ruta correspondiente al escritorio del usuario.

```
cloudmanager@grupo4b1a:~$ cd Desktop/
cloudmanager@grupo4b1a:~/Desktop$ git clone https://github.com/Vanny-Lugo/GrupoB
1A.g2
Cloning into 'GrupoB1A.g2'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (9/9), done.
Unpacking objects: 100% (14/14), 2.95 KiB | 1008.00 KiB/s, done.
remote: Total 14 (delta 3), reused 11 (delta 3), pack-reused 0
cloudmanager@grupo4b1a:~/Desktop$
```

Ahora se agrega un nuevo archivo al repositorio local. En esta paso se modifica el archivo "README.md"

```
cloudmanager@grupo4b1a:~/Desktop/GrupoB1A.g2$ git add README.md
cloudmanager@grupo4b1a:~/Desktop/GrupoB1A.g2$
```

Se utiliza "git commit" para comentar el estado del proyecto en este momento.

```
cloudmanager@grupo4b1a:~/Desktop/GrupoB1A.g2$ git config --global user.name "Vanny-Lugo"
cloudmanager@grupo4b1a:~/Desktop/GrupoB1A.g2$ git commit -m "ejemplo de comentar"
On branch master
Your branch is up to date with 'origin/master'.
```

Por último, se envían todos los cambios mediante "git push".

```
cloudmanager@grupo4b1a:~/Desktop/GrupoB1A.g2$ git push
Username for 'https://github.com': Vanny-Lugo
Password for 'https://Vanny-Lugo@github.com':
Everything up-to-date
```

REFERENCIAS

[1] Curso de Python. Implementación herencia.
https://youtu.be/u_VbLsIyzRk

[2] Tutorial básico de Python. Carlos Huamaní.
https://frontendlabs.io/1305--tutorial-basico-de-python-parte-iv-programacion-orientada-a-objetos

[3] Libro de comunicaciones. Homero Ortega.
https://drive.google.com/file/d/1vGuvcL-3BPdIOqpK-VqGGDYXiDD_SOVE/view

[4] Funcionamiento de clases y objetos en Python.
https://programacion.net/articulo/como_funcionan_las_clases_y_objetos_en_python_1505

[5] GitHub. Repositorio grupo B1A.g2.
https://github.com/Vanny-Lugo/GrupoB1A.g2