CURSO DE PROGRAMACIÓN

POR ING. MATÍAS DIEZ

python™

CURSO DE PROGRAMACIÓN

Fechas del curso

- ~~Primer clase: 05/09/2019~~
- Segunda clase: 12/09/2020
- Tercera clase: 18/09/2020
- Cuarta clase: 25/09/2020

CURSO DE PROGRAMACIÓN

Temario del Curso

- ~~Primer clase:~~

~~> Introducción VS Code~~

~~> Variables y datos alfanuméricos~~

- Segunda clase:

> Variables Numéricas y librerías matemáticas.

> Expresiones lógicas y condicionales

- Tercera clase:

>Funciones, listas y Diccionarios

- Cuarta clase:

>Tuplas, manejo de Archivos, Módulos

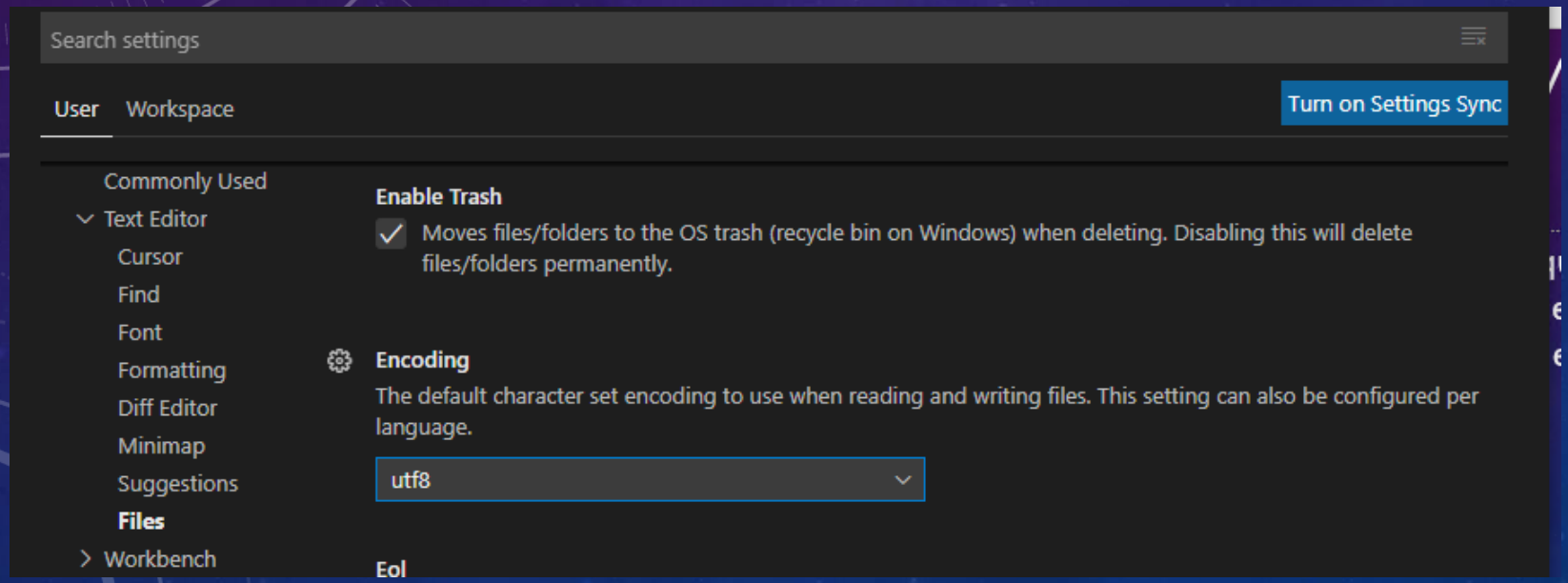
CURSO DE PROGRAMACIÓN

“Encoding” de los archivos.

Como el lenguaje Python esta preparado para poder ejecutarse en cualquier entorno lo que necesitamos es poder configurar un set de caracteres que sea reconocido de forma estandarizada para eso sugerimos emplear el set de caracteres UTF-8.

Pare esto se pone al inicio de cada bloque de código la sentencia:

 # -- coding: utf-8 -*-*



python™

CURSO DE PROGRAMACIÓN

Estructuras Básicas de Control - Ciclo

El lenguaje presente la palabra reservada “while” seguida de una condición a evaluar. Se utiliza “:” como inicio de su cuerpo y su bloque debe estar “identado”:

```
# ciclo con condicion numerica

ciclos = 50
i=0
while i < ciclos:
    print ("Ciclo {}", i)
    i += 1
```

```
# opcion de bloque booleano

while True:
    entrada = input("Ingrese una letra para continuar")
    if entrada.__len__() > 0:
        print ("tecla pulsada")
        break
```

El primer bloque se utiliza un expresión matemática y en el segundo una “booleana”.

python™

CURSO DE PROGRAMACIÓN

Estructuras Básicas Bucles (I)

Se emplea la palabra reservada “for” donde siempre se emplea un iterador.
Luego se puede utilizar dos palabras reservadas “range” o “in”.

```
i=0
# loop por rango dentro de string
entrada = "TARIFF PRODUCT"
for i in range(entrada.__len__()):
    print("Rango {}", i)
# loop por rango dentro de string
entrada = "TARIFA PRODUCTO"
for i in entrada:
    print(i)
```

Nota: observar que declarar es un mas sencilla que otros lenguajes.

CURSO DE PROGRAMACIÓN

Estructura de Control de Excepciones (I)

Python presenta el siguiente bloque:

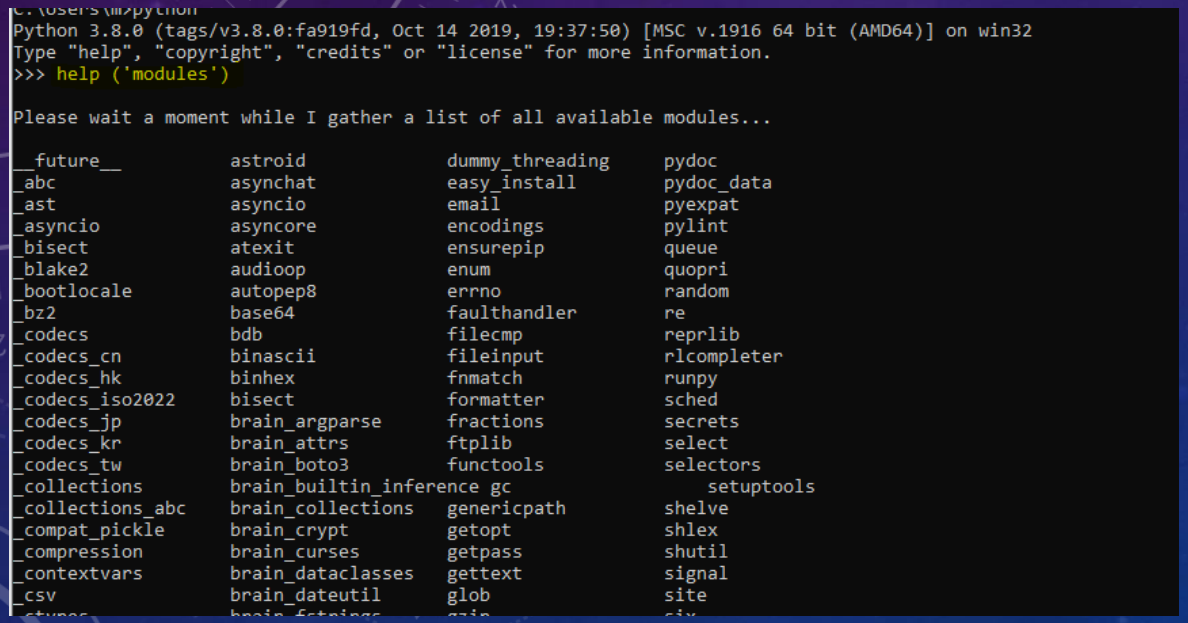
```
try:
    #sentencias a evaluar
except:
    #error a declarar
#opcional
finally:
    # bloque que se ejecutara siempre sin importar
    la situación anormal que se presente.
```

```
try:
    entero=int("udrretrwet")
except:
    print ("algo sucedio")
finally:
    print("salida del programa")
```

CURSO DE PROGRAMACIÓN

Administración de una librería

Cuando uno instala el interprete de Python, hay librerías que vienen por defecto y otras que pueden instalar manual o automáticamente. Para poder ver fácilmente que librerías tengo disponibles puedo ejecutar la sentencia “help (‘modules’)”



CURSO DE PROGRAMACIÓN

Administración de una librería

La lista de los módulos nos es única, va variar de acuerdo a la versión de Python que se emplee y los módulos que se instalen.
Python al ser un lenguaje de alto nivel brinda cantidad de librarías básicas necesarias y mas la posibilidad de extenderlo.
Siguiendo la propuesta del curso vamos a ver como utilizar una libraría ya utilizada.
Mencionaremos algunas de las librerías que son frecuente mente usadas:
“os” , “math” ,”net”, “csv”

CURSO DE PROGRAMACIÓN

Modulo: Reversing - Function

Para entender como funcionan las librerías, vamos a hacer ver como crear una nosotros

Respondamos la primera pregunta: ¿Cuál es expresión más chica?
La expresión más chica es una función la función, se expresa siempre de la siguiente manera:

```
def  nombrefunction (parámetros):
    cuerpo de la función
    retorno de un valor o "nulo"   # el retorno es opcional. Al no existir procedimientos se lo implementan de esta forma.
```

CURSO DE PROGRAMACIÓN

Modulo: Reversing – Modulos

Vamos a ver un ejemplo de como quedaría una sola:

```
def multiplicacion(x, y):
    return x*y

def salida(mensaje):
    print (mensaje)
```

Variaciones -> los parámetros son opcionales, se pueden asignar valores, asiendo una asignación explicita ejemplo (x=0,y=0) , cuando declaramos varias funciones en un mismo archivo.py esta automáticamente se transforma en un modulo. Entre cada función se debe dejar al menos 2 saltos de carro.

python™

CURSO DE PROGRAMACIÓN

Modulo: ++ parametrizacion

*Sobre un parámetro agregando el operador * se pueden pasar multiples valores:*

```
def echoMultiple(*mensaje):
    for x in mensaje:
        print(x)
    return
```

¿Cómo uso los módulos implementado?
Mediante la sentencia “import” . Supongamos que el modulo se llama “funciones.py”.

Al iniciar el bloque se incluye dicha palabra , por ejemplo “funciones.py”

CURSO DE PROGRAMACIÓN

Modulo: ¿Cómo lo empleo?

Siguiendo la diapositiva anterior se pone la palabra reservada “import”

import funciones

Luego puede invocar directamente cualquiera de las funciones definidas.

Ejemplo:

funciones.multiplicacion(10,20)

Esto también se emplea con cualquier librería que esta disponible en carpeta Lib de la distribución elegida.

CURSO DE PROGRAMACIÓN

Alguno de los tantos módulos

“os” permite el acceso los recursos disponibles en entorno donde se ejecuta-
“math” presenta toda una serie funciones matemáticas.
“asyncio” permite la programación asincrónica
“panda” se emplea para realizar big data
“numpy” lo utilizamos para crear estructuras de datos en múltiples dimensiones
“py2exe” permite la creación de paquetes de distribución en formato Windows

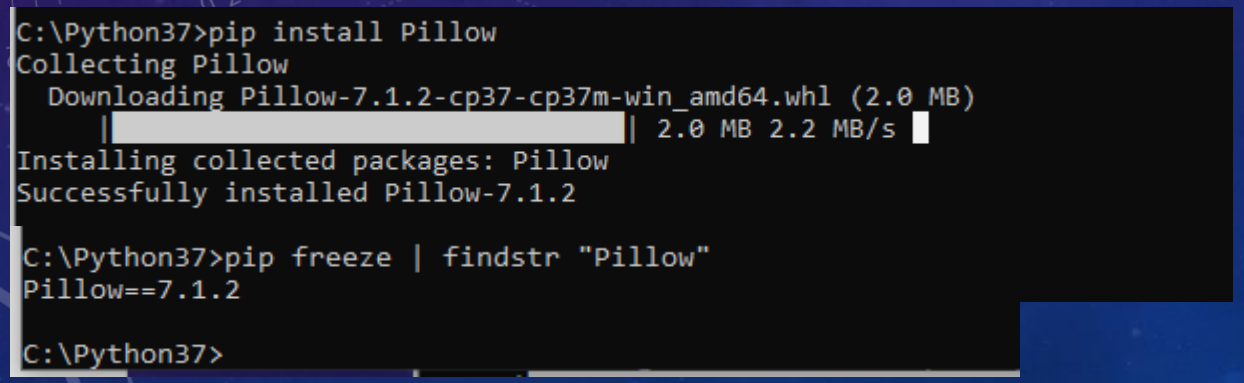
Ya que estamos … una mancha más al tigre que le hace…
¿Cómo se instalan módulos?
Se utiliza la utilidad “pip” (que también es un programa en Python).
Para no extendernos más haya por defecto necesita que el espacio de computo tenga capacidad de salida Internet.

CURSO DE PROGRAMACIÓN

PIP

El pip automáticamente se conecta al espejo mas cercano (Python tiene una CDN sobre internet que permite realizar la descarga, la instalación y en el caso que corresponda compilación / recuerden el phyton es un derivado del c */*

Veamos como ejecutar el comando:
(siempre asegurarse de estar en la carpeta donde esta el Python descargado)
pip freeze > muestra todos los paquetes previamente instalados
pip install nombremodulo >> ejemplo:



python™

CURSO DE PROGRAMACIÓN

PIP Segunda Parte

Una vez instalado ya esta disponible para usar.
Veamos el siguiente ejemplo usando 3 “modulos externos”

```
import pyscreenshot as ImageGrab
import subprocess

im = ImageGrab.grab()
im.save('pantalla.png')

subprocess.call(r'C:\Windows\system32\mspaint.exe pantalla.png’)
```

```
C:\Python37>pip install pyscreenshot
Collecting pyscreenshot
  Downloading pyscreenshot-2.2-py2.py3-none-any.whl (28 kB)
Collecting mss; python_version > "3.4"
  Downloading mss-5.1.0-py3-none-any.whl (75 kB)
     |████████████████████████████████| 75 kB 1.4 MB/s
Collecting entrypoint2
  Downloading entrypoint2-0.2.1-py2.py3-none-any.whl (8.7 kB)
Collecting EasyProcess
  Downloading EasyProcess-0.3-py2.py3-none-any.whl (7.9 kB)
Collecting argparse
  Downloading argparse-1.4.0-py2.py3-none-any.whl (23 kB)
Installing collected packages: mss, argparse, entrypoint2, EasyProcess, pyscreenshot
Successfully installed EasyProcess-0.3 argparse-1.4.0 entrypoint2-0.2.1 mss-5.1.0 pyscreenshot-2.2
```

! Instalar este modulo!

Ejecuto phyton archivo.py # genera una captura y muestra con un Notepad #

CURSO DE PROGRAMACIÓN

Modulo Matemático “math”

Función	Resultado esperado
sin(x)	Retorna el seno de x en radianes
sqrt(x)	Retorna el la raíz cuadrada de x
tan(x)	Retorna la tangente de x en radia
math.py	Devuelve el contante de 3.145xxx
atan(x)	Arcotangente en radianes
math.degrees(x)	Pasa de Radianes a grados
math.radians(x)	Pasa grados a Radianes
cos(x)	Calcula el coseno de x

>>>Mas data aquí<<

CURSO DE PROGRAMACIÓN

Final Feliz!