INFORME NÚMERO 1 DE INFORMÁTICA.

PRIMERA CLASE: 24 AGOSTO 2022.

ENTORNO A PHYTON

Con esta clase se da a conocer el entorno de trabajo en el que se desarrollará el curso hasta su final, repasamos las herramientas que utilizaremos en todo el semestre, se instala Visual Studio Code directamente de su instalador descargado desde google, así también lo hicimos con la herramienta de programación Python el cual nos servirá para abordar diferentes aplicaciones en la vida real es por esto que nos adentraremos en su lenguaje y por último realizamos la descarga e instalación de GitHub una herramienta que nos permite subir archivos del curso a la nube incluyendo los códigos de Python ya que estas se relacionan a su vez con el VSC, por esto creamos una cuenta o login para interconectar las plataformas en cada herramienta y poder tener un uso óptimo de estas aplicaciones.

GitHub:

Es un sitio "social coding". Te permite subir repositorios de código para almacenarlo en el sistema de control de versiones Git.

Esta aplicación permite compartir proyectos de software con toda la comunidad de ingenieros y programadores de código abierto, actualmente es una de las herramientas más usadas, ya con más de 12 años desde su creación es la aplicación preferida para administrar repositorios de información.

Visual Studio Code: Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web.

Esta aplicación es desarrollada por Microsoft para editores de código fuente para diferentes sistemas operativos como Windows o Linux.

SEGUNDA CLASE: 26 AGOSTO 2022

Este dia repasamos la definición de los tipos de datos como aquel conjunto de datos que tienen características o propiedades establecidas o determinadas. Luego vimos que los tipos de datos como los de tipo numéricos trabajan con los números enteros o números de punto flotante lo cual en Py se usa la función “Float()”. Luego existe otro tipo de datos que se trabaja con booleanos y vimos que en Py se representa como “bool” y contiene dos valores (En base 2), es decir True o False. También existen otros tipos de datos como las listas, las tuplas o los diccionarios cuando son datos almacenados en un conjunto o llaves tales como “[1,2,3]”.

Los tipos de datos en Py: Los tipos de datos se consideran como un conjunto de datos que poseen una característica o propiedad determinada. por ejemplo: [1,2,3] a esto en python le llamamos tipos de datos.
Tipos de datos básicos en Py: Estos datos pueden ser específicamente numéricos, punto flotante (Números reales), complejos y las cadenas de caracteres. También hay otros tipos de datos como lo son las listas, las tuplas, mapas, conjuntos, iteradores, clases, instancias y excepciones.

Ejemplos de repaso vistos en clase:
TIPOS DE DATOS EN python

*** BOOLEANOS: True, False**
*** STRINGS: "", '', "casa", "123", """", "12.23132"**
*** ENTEROS: -1000, 1000, 0,**
*** FLOTANTES: 12.5, -100.0, 5.**

*** LISTAS: [], [1,2,3], ["A", "B", "C"], [1, "A", 2, "B", 3, "C"] ==> MUTABLES**
*** TUPLAS: (), (1,2,3), ("A", "B", "C"), (1, "A", 2, "B", 3, "C") ==> INMUTABLES**

*** CONJUNTOS: {"A", "B", "C"}**
*** DICCIONARIOS: {"CRISTIAN": 5, "DANIELA": 0, "SEBASTIAN": 3}**

CLASE NUMERO 3: 31 DE AGOSTO 2022

Hay diferentes tipos de operadores que usaremos en Py que van desde los tipos de operadores booleanos hasta tipo los tipos String; para estos operadores hay unos símbolos que son propios del lenguaje en cuestión el cual se usan para operar uno, dos o más elementos, también analizaremos la estructura jerárquica que poseen estos operadores según Py y su estructura lógica para que funcionen correctamente en la redacción de los códigos de programación.

- **Tipo de operaciones booleanas:** Son denominadas de la siguiente manera, +, And, Or y Not.
  Hay que recalcar que las operaciones no devuelven True o False, devuelve otro operador.
  A or B, si A se evalúa como falso, entonces devuelve B, sino devuelve A. Es decir, evalúa el segundo término si A es falso.
  A and B, si A se evalúa a falso, entonces devuelve A sino devuelve B. Es decir, solo evalúa el segundo operando si el primero o sea A es verdadero.
- **Operadores aritméticos:** Estos operadores me permiten realizar operaciones tales como la suma, resta, multiplicación, división etc.

Por ejemplo:

```
x = 7
y = 2
x + y  # Suma
9
x - y  # Resta
```

```
5
x * y  # Producto
14
x / y  # División
3.5
x % y  # Resto
1
x // y  # Cociente
3
x ** y  # Potencia
49
```

- **Operadores de comparación:** Estos operadores como su nombre lo indica se usan para comparar 2 o más valores. Su respuesta es un True o False.

  Por ejemplo tenemos:

```
>>> x = 9
>>> y = 1
>>> x < y
False
>>> x > y
True
>>> x == y
False
```

- **Operadores de concatenación o tipo string:** Una de las operaciones básicas en Py cuando se trabaja con cadenas de caracteres lo llamamos concatenación. La forma mas simple de concatenar dos cadenas es usando el operador “+”.

CLASE NUMERO 4; 02 DE SEPTIEMBRE DEL 2022

FUNCIONES INTEGRADAS:

La librería estándar de Python incluye muchas funciones. Las hay para hacer conversiones entre tipos, matemáticas, utilidades...

Aquí un resumen de las más utilizadas incluyendo algunas que ya conocemos:

- Entrada/salida. print() input() ...
- Ayuda para programar/debugging. help() ...
- Funciones matemáticas. abs() ...
- Estructuras de datos. list() ...
- Generación de secuencias iterables. range() ...
- Operaciones con objetos iterables. len() ...
- Modificaciones de objetos iterables. sorted() ...
- Funciones aplicadas a iterables. map()

Hay diferentes funciones integradas de Python que van desde funciones de entrada y salida hasta funciones de operaciones con secuencia. Describimos cada una e hicimos unos breves ejemplos de su funcionamiento, hasta el momento podemos ver como funciona cada una de ellas y a partir de el aprendizaje de la sintaxis de Py podemos darle un mejor uso para optimizar y sentirnos más cómodos con el lenguaje de programación.

Funciones de entrada y salida: Este tipo de funciones las usamos para introducir y reproducir datos, como muy bien se menciona estas funciones manejan un dominio numérico y de tipo string, estas funciones son: input(), print() y format().

- **Funciones de ayuda para programar:**

- help(): muestra la ayuda integrada de cualquier componente de Py, si se deja libre la función sin ningún componente que queramos comprender entonces abre una box de ayuda interactiva.
- **Funciones matemáticas:** Estas funciones me permiten trabajar con números para importar valores que deben pasar por cierta operación matemática, es decir:
- abs(): Me permite sacar el valor absoluto de un número

- **Funciones de conversión:** Este tipo de funciones la usamos para convertir valores de entrada de una misma característica en otra forma, por ejemplo si son números podemos trabajar en conversión a sistemas numéricos entre sí.

CLASE NUMERO 5: 02 DE SEPTIEMBRE 2022

En esta clase pudimos describir la sentencia if-else, conocer su sintaxis en
python y aplicamos diferentes casos en los que se puede aplicar y las ventajas que tiene
este condicional.

CONDICIONAL IF:

```
Permite ejecutar un
conjunto de sentencias,
en caso de que una
condición sea verdadera.
```

```
En caso de que la
condición sea falsa, las
sentencias
contenidas son ignoradas
```

Sentencia if: Llegados a este punto en el conocimiento de nuestro lenguaje en el que ya
podemos trabajar con condicionales y variables bien definidas para este caso tenemos la
sentencia “if” el cual se utiliza para ejecutar un código si, y sólo si se cumple determinada
condición. la estructura de este condicional es:
es decir si solo se cumple la condición o está al evaluarla es un True entonces el código
puede ejecutarse. En caso contrario, si la sentencia es False, no se evalúa ningún código.

Sentencia if…else:
Debido a que la sentencia if es insuficiente porque sólo evalúa objetos a True, muchas veces vamos a requerir de evaluar objetos a False para ejecutar aplicaciones de nuestro
código.

EJEMPLO VISTO EN CLASE:

```
“””Determine si un numero es mayor o no a
18,
utilizando el condicional IF-ELSE
"""
numero = 18
if numero > 18:
   print("Numero es mayor a 18")
else:
   print("Numero no es mayor a 18")
```

```
"""
Pedir a el usuario que ingrese 3 numeros,
luego, imprimir el numero mayor y el menor
"""


numero1 = int(input("Ingrese primer numero: "))
numero2 = int(input("Ingrese segundo numero: "))
numero3 = int(input("Ingrese tercer numero: "))


mayor = 0


if (numero1 >= numero2) and (numero1 >= numero3):
   mayor = numero1
elif (numero2 >= numero1) and (numero2 >= numero3):
   mayor = numero2
elif (numero3 >= numero1) and (numero3 >= numero2):
   mayor = numero3


print("El numero mayor es {}".format(mayor))
```

CLASE NUMERO 6: 09 DE SEPTIEMBRE 2022

Para esta práctica hicimos una descripción breve del bucle while en python, teniendo en cuenta los casos en los que se puede aplicar y como romper un ciclo infinito de estos hicimos algunos ejemplos para comprender este tipo de bucle.

El CICLO WHILE evalúa una condición, en caso de ser verdadera ejecuta las condiciones contenidas en el código.

Durante el ciclo la condición podría cambiar a falso, lo cual implicaría que el ciclo termine en la siguiente ejecución.

Bucle While: Este tipo de bucle maneja una estructura condicional similar al if ya que de ser cumplida una sentencia en True entonces el código se ejecutará en bloque como sigue después del while.

```
numero = 0
print('Tabla del 3')
while numero <= 10:
    print(f'{numero * 3}')
    numero += 1
print('Fin')
```

```
Tabla del 3
0
3
6
9
12
15
18
21
24
27
30
Fin
```

```python
valores = [5, 1, 9, 2, 7, 4]
eureka = False
indice = 0
long= len(valores)
while not eureka and indice < long:
    valor = valores[indice]
    if valor == 2:
        eureka = True
    else:
        indice += 1
if eureka:
    print(f'El número 2 ha sido encontrado en el índice {indice}')
else:
    print('El número 2 no se encuentra en la lista de valores')
```

```
El número 2 ha sido encontrado en el índice 3
```

En el ejemplo anterior encontramos:

- eureka: Indica si el número 2 ha sido encontrado en la lista.
- índice: Contiene el índice del elemento de la lista valores que va a ser evaluado.
- long: Indica el número de elementos de la lista de valores.

CICLO FOR:

El bucle for se utiliza para recorrer los elementos de un objeto *iterable* (lista, tupla, conjunto, diccionario, …) y ejecutar un bloque de código. En cada paso de la iteración se tiene en cuenta a un único elemento del objeto iterable, sobre el cuál se pueden aplicar una serie de operaciones.

Ejemplo de cómo usar el bucle for

Imaginemos que tenemos una lista de números y queremos mostrarlos por consola. El código podría ser así:

```python
nums = [4, 78, 9, 84]
```

```
for n in nums:
 print(n)
4
78
9
84
```

Ahora hay que tener en cuenta también el concepto de iterable; *iterable* es un objeto que se puede iterar sobre él, es decir, que permite recorrer sus elementos uno a uno. Para ser más técnico, un objeto iterable es aquél que puede pasarse como parámetro de la función iter().

Esta función devuelve un iterador basado en el objeto iterable que se pasa como parámetro.

Un caso es especial de bucle for se da al recorrer los elementos de un diccionario. Dado que un diccionario está compuesto por pares clave/valor, hay distintas formas de iterar sobre ellas.

1 . Recorrer las claves del diccionario.

```
valores = {'A': 4, 'E': 3, 'I': 1, 'O': 0}
for k in valores:
   print(k)
A
E
I
```

O

2 . Iterar sobre los valores del diccionario

```
valores = {'A': 4, 'E': 3, 'I': 1, 'O': 0}

for v in valores.values():

   print(v)
```

```
4

3

1

0
```