Insaforp
FORMANDO UN MEJOR EL SALVADOR

FORMÁS
CENTRO DE FORMACIÓN AVANZADO PARA PROFESIONALES

FEDISAL
Fundación para la Educación Integral Salvadoreña

MODULO 1. INTRODUCCIÓN AL DESARROLLO DE SOFTWARE CON PYTHON

1. PROGRAMACIÓN ESTRUCTURADA CON PYTHON

CONTENIDOS A DESARROLLAR

1. Argumentos en línea de comandos
2. Manejo de excepciones

Programación Orientada a Objetos

1. Clases y objetos
2. Herencia
    1. Clases abstractas
    2. Herencia múltiple
3. Variables de clase (estáticas)
4. @classmethod vs @staticmethod
5. Constantes

ARGUMENTOS LÍNEA DE COMANDOS

- Dado que Python es un lenguaje de programación tan popular, además de tener soporte para la mayoría de los sistemas operativos, se ha vuelto ampliamente utilizado para crear herramientas de línea de comandos para muchos propósitos.

$ python arguments-count.py --option "long string“

- El objeto sys.argv es una lista que contiene los argumentos pasados al programa.
- import sys
- print(sys.argv)
- Recuérdese que el primer elemento de la lista siempre se corresponde con el nombre del programa, de modo que los argumentos de éste, estrictamente hablando, se encuentran en sys.argv[1:]

EXCEPCIONES EN PYTHON

- Las excepciones en Python son una herramienta muy potente que la gran mayoría de lenguajes de programación modernos tienen. Se trata de una forma de controlar el comportamiento de un programa cuando se produce un error.
- Esto es muy importante ya que salvo que tratemos este error, **el programa se parará**, y esto es algo que en determinadas aplicaciones no es una opción válida.

EXCEPCIONES EN PYTHON

- imaginemos ahora que por cualquier motivo las variables tienen otro valor, y que por ejemplo b tiene el valor 0. Si intentamos hacer la división entre cero, este programa dará un error y su ejecución terminará de manera abrupta.

```
a = 4; b = 0

print(a/b)

# ZeroDivisionError: division by zero
```

EXCEPCIONES EN PYTHON

- Ese “error” que decimos que ha ocurrido es lanzado por Python (raise en Inglés) ya que la división entre cero es una operación que matemáticamente no está definida.

- Se trata de la excepción ZeroDivisionError. En el siguiente enlace, tenemos un listado de todas las excepciones con las que nos podemos encontrar.

- https://docs.python.org/3/library/exceptions.html

EXCEPCIONE: KEYBOARDINTERRUPT

```
if __name__ == '__main__':
        try:
                main()
        except KeyboardInterrupt:
                sys.exit()
```

USO DE RAISE

- También podemos ser nosotros los que levantemos o lancemos una excepción. Volviendo a los ejemplos usados en el apartado anterior, podemos ser nosotros los que levantemos ZeroDivisionError o NameError usando raise. La sintaxis es muy fácil.

```
if a == b:
    raise Exception("Error", "iguales")
```

Insaforp
FORMANDO UN MEJOR EL SALVADOR

FORMÁS
CENTRO DE FORMACIÓN AVANZADO PARA PROFESIONALES

FEDISAL
Fundación para la Educación Integral Salvadoreña

MODULO 1. INTRODUCCIÓN AL DESARROLLO DE SOFTWARE CON PYTHON

2. PROGRAMACIÓN ORIENTADA A OBJETOS

CONTENIDOS A DESARROLLAR

1. Clases y objetos
2. Herencia
    1. Clases abstractas
    2. Herencia múltiple
3. Variables de clase (estáticas)
4. @classmethod vs @staticmethod
5. Constantes

PROGRAMACIÓN ORIENTADA A OBJETOS

- Este modo o paradigma de programación nos permite organizar el código de una manera que se asemeja bastante a como pensamos en la vida real
- Utilizando las famosas clases.
- Estas nos permiten agrupar un conjunto de variables y funciones que veremos a continuación

CLASES Y OBJETOS

- Clase: La descripción de un objeto (estados y comportamientos)
- Objeto: La instancia de una clase
- Instancia? La lectura de la clase para convertirla en un objeto
- Miembros de una clase: atributos y métodos
- Características de la POO?
- 1. Herencia (Clase base y derivada)
  - Herencia múltiple (Varios padres para una clase)
  - Clase Abstracta (no se puede instanciar)
- 2. Encapsulamiento
- 3. Polimorfismo (sobreescritura)

CONSTRUCTOR?

- Un método opcional con un nombre especial (__init__)
- Inicializa un objeto y sus variables de instancia (reserva la memoria)
- Método que se ejecuta en la creación del objeto
- Por ejemplo: Persona("juan")

ENCAPSULAMIENTO

- Se denomina encapsulamiento al ocultamiento del estado, es decir, de los datos miembro de un objeto de manera que solo se pueda cambiar mediante las operaciones definidas para ese objeto.
- De esta forma el usuario de la clase puede obviar la implementación de los métodos y propiedades para concentrarse solo en cómo usarlos. Por otro lado se evita que el usuario pueda cambiar su estado de maneras imprevistas e incontroladas.

ENCAPSULAMIENTO

- Dicho de otra manera, encapsular consiste en hacer que los atributos o métodos internos a una clase no se puedan acceder ni modificar desde fuera, sino que tan solo el propio objeto pueda acceder a ellos.
- Niveles de aislamiento típicos
  - Publico: accesible desde cualquier parte del programa
  - Protegido: accesible por la misma clase y clases en relación de herencia
  - Privado: accesible únicamente desde la misma clase

  - Elaborar ejemplos según indicaciones del instructor