

**Este curso se asienta sobre la premisa de que los participantes han adquirido previamente conocimientos fundamentales en Python. Su objetivo radica en consolidar prácticas avanzadas en los siguientes conceptos:**

**En Python, la conceptualización de objetos atraviesa todas las instancias. En Python todo son objetos.**

1. **Las variables son objetos que actúan como elementos para el resguardo de información. Otros objetos incluyen las funciones integradas (built-in) y las definidas por el usuario, que son bloques de código reutilizables, así como los operadores, que realizan operaciones en objetos y valores.**
2. **Existen variables como objetos capaces de contener un único dato, como tipos numéricos (int, float, complex, oct, hex, bin), strings y booleanos, que coexisten con colecciones más complejas, tales como listas (list), conjuntos (set) y diccionarios (dict), caracterizadas por su estructura clave-valor.**
3. **En Python, solo hay un puñado de constantes preestablecidas en el intérprete, y el programador no puede crear nuevas constantes de manera estricta. Lo más cercano al concepto de constantes son las tuplas o los frozensets, pero estas son colecciones (estructuras que admiten múltiples datos).**

**Aunque no hay un concepto estricto de "constantes" en el sentido tradicional, la comunidad de programadores Python establecio poner en mayusculas el nombre del objeto para indicar que ciertas variables deben tratarse como constantes, (valores si bien pueden, no deben modificarse durante la ejecución del programa). Ver convención de estilo de Python.**

1. **Los objetos son representaciones o instancias de una clase y poseen tanto métodos como atributos. Los atributos pueden entenderse como características o propiedades que describen el estado de un objeto, mientras que los métodos son funciones asociadas a la clase que actúan sobre esos atributos o realizan operaciones específicas.**

**Principales funciones y estructuras de control de flujo en Python**

* **print: La función print() se utiliza para mostrar información en la consola. Puedes imprimir valores, variables o mensajes.**
* **if, else, elif: La estructura de control if, else, y elif se utiliza para tomar decisiones basadas en condiciones. Se ejecuta un bloque de código si la condición es verdadera y, en caso contrario, se ejecuta otro bloque.**
* **input: La función input() se utiliza para obtener la entrada de datos del usuario desde la consola. Esta captura siempre devuelve un string (cadena de texto).**
* **while: La estructura de control while se utiliza para repetir un bloque de código mientras una condición sea verdadera.**
* **for: La estructura de control for se utiliza para iterar sobre una secuencia de elementos como listas, tuplas, cadenas, string entre objetos iterables.**

**Funciones propias**

* **def y return: La palabra clave def se utiliza para definir una función en Python. La función puede tener parámetros y realiza operaciones específicas dentro del bloque de código indentado. La palabra clave return se utiliza para devolver un resultado desde la función.**

**Algunas funciones incorporadas en Python que trabajan con secuencias de datos.**

* **min(): Devuelve el valor mínimo de una secuencia.**
* **max(): Devuelve el valor máximo de una secuencia.**
* **sum(): Devuelve la suma de todos los elementos en una secuencia.**
* **len(): Devuelve la longitud (número de elementos) de una secuencia.**

**El propósito de este curso es revisar y ampliar los conocimientos previos desde un enfoque más profesional en programación. Abordaremos la manipulación de grandes cantidades de información mediante el uso de bases de datos y exploraremos una introducción al manejo de interfaces gráficas de usuario (GUI- Graphical User Interface)**

**Las funciones de orden superior son aquellas funciones que toman una o más funciones como argumentos o devuelven funciones como resultados. Aquí te presento algunas funciones de orden superior en Python y sus descripciones:**

* **map(func, iterable, ...): Aplica la función dada a cada elemento del iterable (o varios iterables) y retorna un iterable con los resultados.**
* **filter(func, iterable): Filtra los elementos de un iterable basándose en si la función devuelve True o False. Retorna un iterable con los elementos que cumplen la condición.**
* **reduce(func, iterable[, initializer]): Aplica la función de manera acumulativa a los elementos del iterable, de izquierda a derecha, reduciendo el iterable a un solo valor. El argumento opcional initializer proporciona un valor inicial. from functools import reduce**

**Funciones built in:**

* **any(iterable): Retorna True si al menos un elemento del iterable es verdadero. Si el iterable está vacío, retorna False.**
* **all(iterable): Retorna True si todos los elementos del iterable son verdaderos (o si el iterable está vacío).**
* **sorted(iterable, key=None, reverse=False): Retorna una nueva lista ordenada de los elementos de un iterable. El argumento key puede especificar una función de comparación, y reverse controla el orden ascendente o descendente.**
* **reversed(sequence): Retorna un objeto iterable que produce los elementos de la secuencia en orden inverso.**
* **zip(iterable1, iterable2, ...): Combina elementos de varios iterables en tuplas. Retorna un iterable de tuplas, donde la i-ésima tupla contiene el i-ésimo elemento de cada iterable.**
* **enumerate(iterable, start=0): Retorna pares (índice, elemento) para cada elemento en el iterable, comenzando desde el índice proporcionado (por defecto, desde 0).**

**Funciones útiles durante la depuración y la exploración de objetos en un programa Python.**

* **type(object): La función type se utiliza para obtener el tipo de un objeto. Puede devolver el tipo de cualquier objeto, ya sea un tipo incorporado o un tipo definido por el usuario.**
* **dir([object]): La función dir se utiliza para obtener una lista de nombres de atributos válidos para un objeto. Si se proporciona un objeto, la función devuelve los atributos del objeto; de lo contrario, devuelve los nombres de los atributos en el ámbito local.**
* **id(object): La función id devuelve el identificador único (identificador de objeto) de un objeto. Este identificador es único durante el tiempo de vida del objeto y se genera internamente por CPython, el intérprete de Python de referencia.**

**Funciones de casting o funciones de conversión. Se utilizan para cambiar el tipo de datos de un objeto a otro en Python.**

* **int(x [,base]): Convierte x a un entero. Si x es una cadena, puede incluir un segundo argumento opcional base que representa la base del sistema numérico (por defecto es 10).**
* **float(x): Convierte x a un número de punto flotante.**
* **str(x): Convierte x a una cadena.**
* **bool(x): Convierte x a un valor booleano. Devuelve False para 0, None, secuencias o colecciones vacías, y True para cualquier otro valor.**
* **list(iterable): Convierte un iterable (como una tupla, cadena, o conjunto) a una lista.**
* **tuple(iterable): Convierte un iterable a una tupla.**
* **set(iterable): Convierte un iterable a un conjunto.**

**Manejo de excepciones:**

* **Los bloques try, except, finally, else son utilizados en Python para gestionar excepciones, que son situaciones excepcionales o errores que pueden ocurrir durante la ejecución de un programa.**
* **try : Este bloque contiene el código que puede lanzar una excepción. El intérprete de Python intentará ejecutar este código, y si se produce una excepción, saltará a la sección except correspondiente.**
* **except: Este bloque se ejecuta cuando se produce una excepción en el bloque try. Puedes tener múltiples bloques except para manejar diferentes tipos de excepciones.**
* **else: Este bloque se ejecuta si no se produce ninguna excepción en el bloque try. Proporciona un código que se ejecutará solo si el bloque try se ejecuta sin errores.**
* **finally: Este bloque se ejecuta siempre, ya sea que se haya producido una excepción o no en el bloque try. Se utiliza comúnmente para realizar acciones de limpieza o liberar recursos, independientemente de si se maneja la excepción o no.**

**Manejo de datos con bases de datos es un aspecto fundamental en el desarrollo de aplicaciones y sistemas de información.**

* **Diseño de la Base de Datos.**
* **Modelo de Datos (Define la estructura y las relaciones entre los datos con MySQL o SQLite3)**
* **Entidades, Atributos y Relaciones (enlazar entidades entre sí esta pueden ser uno a uno, uno a muchos o muchos a muchos).**

**Creación y Conexión:**

* **Crear la Base de Datos: Utiliza un sistema de gestión de bases de datos (DBMS) para crear la base de datos según el diseño previamente establecido.**
* **Establecer Conexión: Desde tu aplicación, establece una conexión con la base de datos utilizando un conector o un controlador específico del DBMS.**

**Operaciones CRUD:**

* **Crear: Inserta nuevos datos en la base de datos.**
* **Leer: Recupera datos de la base de datos mediante consultas (queries).**
* **Actualizar: Modifica datos existentes en la base de datos.**
* **Eliminar: Elimina datos de la base de datos.**

**Lenguaje SQL:**

* **Structured Query Language (SQL): Es el lenguaje estándar para realizar operaciones en bases de datos relacionales. Incluye instrucciones como SELECT, INSERT, UPDATE, DELETE, y CREATE.**

**Transacciones:**

* **Transacciones: Agrupan operaciones en una unidad lógica. Garantizan la consistencia de los datos, permitiendo confirmar o deshacer un conjunto de operaciones en bloque.**

**Indices y Optimización:**

* **Indices: Mejoran el rendimiento de las consultas al proporcionar un acceso más rápido a los datos. Se crean en columnas específicas.**
* **Optimización de Consultas: Asegura que las consultas sean eficientes y se ejecuten en un tiempo razonable. Puede implicar el uso de índices, ajuste de consultas o revisión del diseño de la base de datos.**

**CRUD: Es un acrónimo que representa las cuatro operaciones fundamentales en la gestión de datos, forman la base de las interacciones de manipulación de datos en muchas aplicaciones y sistemas:**

* **Create: La operación de creación implica la inserción de nuevos datos en una base de datos o sistema de almacenamiento.**
* **Read: La operación de lectura implica la recuperación de datos existentes de una base de datos o sistema de almacenamiento.**
* **Update: La operación de actualización implica la modificación de datos existentes en una base de datos o sistema de almacenamiento.**
* **Delete: La operación de eliminación implica la remoción de datos existentes en una base de datos o sistema de almacenamiento.**

**ABM: Es un acrónimo que se utiliza en el contexto de sistemas de información y representa las operaciones de Alta, Baja, y Modificación principalmente de datos:**

* **Alta: Similar a la operación de "Create" en CRUD, el alta implica la inserción de nuevos datos o la creación de nuevos registros.**
* **Ejemplo: Registrar un nuevo cliente en un sistema.**
* **Baja: Similar a la operación de "Delete" en CRUD, la baja implica la eliminación de datos o registros existentes.**
* **Modificación: Similar a la operación de "Update" en CRUD, la modificación implica la alteración de datos existentes.**

**Diferencias y Relación:**

**CRUD es más amplio y general, abarcando la creación, lectura, actualización y eliminación de datos, campos o estructuras de las bases de datos.**

**ABM se centra específicamente en las operaciones de alta, baja y modificación, que son un subconjunto de las operaciones CRUD mas especifico cuando se quiere enfocar en las operaciones fundamentales de manipulación de datos en sistemas.**

**Manejo de interfaces gráficas de usuario (GUI- Graphical User Interface).**

**Una GUI es un tipo de interfaz que permite la interacción entre usuarios y sistemas informáticos a través de elementos gráficos como iconos, botones, ventanas y menús, en contraste con interfaces de línea de comandos (CLI) que utilizan texto. Estas hacen que las interacciones con los sistemas informáticos sean más intuitivas y visuales, facilitando la tarea para los usuario. Estas interfaces son comunes en sistemas operativos, aplicaciones de software, aplicaciones móviles y muchos otros contextos informáticos.**

**Nuevamente en Python todo son objetos. Los componentes visuales relacionedos a la pantalla gráfica interactiva se denominan “widget” y permiten a los usuarios interactuar con una aplicación o programa mediante acciones como hacer clic, arrastrar, o ingresar datos.**

**Widgets comunes:**

**Ventanas: Contenedores para organizar y mostrar información o aplicaciones.**

**Botones: Elementos que los usuarios pueden presionar para activar una acción.**

**Iconos: Representaciones gráficas de archivos, aplicaciones u otras funciones.**

**Menús: Listas desplegables de opciones que los usuarios pueden seleccionar.**

**Cuadros de Texto: Áreas donde los usuarios pueden ingresar o editar texto.**

**Etiquetas: Componentes para mostrar información estática.**

**Listas y Cuadros Combinados: Permiten a los usuarios seleccionar entre opciones.**

**Casillas de Verificación y Botones de Opción:Para seleccionar opciones en formularios.**