Documentación Técnica:

- Manual de Referencia.
- Guía de Usuario.

Manual de Referencia - PyLearningHub

Este manual presenta una explicación clara y detallada de los temas abordados en el curso **PyLearningHub**, acompañada de ejemplos y ejercicios realizados.

Lección 1: Variables y Tipos de Datos

Concepto:
Las variables almacenan datos y los tipos de datos definen la naturaleza de esos datos (número, texto, booleano, etc.).

Tipos básicos:

- `int`: números enteros
- `float`: números decimales
- `str`: cadenas de texto
- `bool`: verdadero o falso

Ejemplo:

```
nombre = "Ana"
edad = 25
estudiante = True
```

Ejercicio realizado: El usuario introduce su nombre mediante `input()` y se almacena usando una variable. Se valida que no sea numérico.

Lección 2: Operadores y Estructuras Condicionales

Concepto: Los operadores permiten realizar operaciones matemáticas y lógicas. Las estructuras condicionales (`if`, `else`, `elif`) ejecutan código según condiciones.

Ejemplo:

```
edad = 20
if edad >= 18:
    print("Eres mayor de edad")
else:
    print("Eres menor de edad")
```

Ejercicio realizado: Se pide al usuario su edad y se determina si es mayor o menor de edad.

Lección 3: Funciones y Módulos

Concepto: Las funciones encapsulan código reutilizable. Los módulos permiten importar funciones externas o propias.

Ejemplo:

```
def saludar(nombre):
    print(f"Hola {nombre}!")

saludar("Carlos")
```

Ejercicio realizado: Se define una función para saludar y se llama desde otra parte del programa.

Lección 4: Manejo de Excepciones

Concepto: El manejo de excepciones permite capturar y gestionar errores durante la ejecución del programa para evitar que se detenga abruptamente.

Ejemplo:

```
try:
    valor = int(input("Ingresa un número: "))
except ValueError:
    print("Error: Debes ingresar un número válido.")
```

Ejercicio realizado: Se captura un error cuando el usuario ingresa un dato no numérico.

Lección 5: Programación Orientada a Objetos (POO)

Concepto: La POO organiza el código en clases y objetos, con atributos (características) y métodos (funciones).

Ejemplo:

```
class Persona:
    def __init__(self, nombre, edad):
        self.nombre = nombre
        self.edad = edad

    def presentarse(self):
        print(f"Hola, soy {self.nombre} y tengo {self.edad} años")
```

Ejercicio realizado: Se crea una clase `Estudiante` con atributos y métodos.

Lección 6: Uso Avanzado del Lenguaje

Concepto: Incluye funciones lambda, desempaquetado, tipado dinámico, comprensiones de listas, y funciones como objetos.

Ejemplo:

```
doble = lambda x: x * 2
print(doble(4))

pares = [x for x in range(10) if x % 2 == 0]
```

Ejercicio realizado: Se evalúa el uso de funciones lambda y listas por comprensión para obtener números pares.

Este manual complementa el código y los ejercicios prácticos, y sirve como referencia para seguir practicando Python.

Manual de Referencia - PyLearningHub

Este manual presenta una explicación clara y detallada de los temas abordados en el curso PyLearningHub, acompañada de ejemplos y ejercicios realizados.

Lección 1: Variables y Tipos de Datos

Concepto: Las variables almacenan datos y los tipos de datos definen la naturaleza de esos datos (número, texto, booleano, etc.).

Tipos básicos:

- `int`: números enteros
- `float`: números decimales
- `str`: cadenas de texto
- `bool`: verdadero o falso

Ejemplo:

```
nombre = "Ana"
edad = 25
estudiante = True
```

Ejercicio realizado: El usuario introduce su nombre mediante `input()` y se almacena usando una variable. Se valida que no sea numérico.

Lección 2: Operadores y Estructuras Condicionales

Concepto: Los operadores permiten realizar operaciones matemáticas y lógicas. Las estructuras condicionales (`if`, `else`, `elif`) ejecutan código según condiciones.

Ejemplo:

```
edad = 20
if edad >= 18:
    print("Eres mayor de edad")
else:
    print("Eres menor de edad")
```

Ejercicio realizado: Se pide al usuario su edad y se determina si es mayor o menor de edad.

Lección 3: Bucles y Colecciones

Concepto: Los bucles (`for`, `while`) permiten repetir bloques de código. Las colecciones como listas, diccionarios, tuplas y conjuntos almacenan múltiples datos.

Ejemplo:

```
frutas = ["manzana", "pera", "uva"]
for fruta in frutas:
    print(fruta)
```

Ejercicio realizado: Se crea una lista de tareas y se imprime con un bucle.

Lección 4: Funciones y Módulos

Concepto: Las funciones encapsulan código reutilizable. Los módulos permiten importar funciones externas o propias.

Ejemplo:

```
def saludar(nombre):
    print(f"Hola {nombre}!")

saludar("Carlos")
```

Ejercicio realizado: Se define una función para saludar y se llama desde otra parte del programa.

Lección 5: Programación Orientada a Objetos (POO)

Concepto: La POO organiza el código en clases y objetos, con atributos (características) y métodos (funciones).

Ejemplo:

```
class Persona:
    def __init__(self, nombre, edad):
        self.nombre = nombre
        self.edad = edad

    def presentarse(self):
        print(f"Hola, soy {self.nombre} y tengo {self.edad} años")
```

Ejercicio realizado: Se crea una clase `Estudiante` con atributos y métodos.

Lección 6: Uso Avanzado del Lenguaje

Concepto: Incluye funciones lambda, desempaquetado, tipado dinámico, comprensiones de listas, y funciones como objetos.

Ejemplo:

```
doble = lambda x: x * 2
print(doble(4))

pares = [x for x in range(10) if x % 2 == 0]
```

Ejercicio realizado: Se evalúa el uso de funciones lambda y listas por comprensión para obtener números pares.

Este manual complementa código y ejercicios prácticos, y sirve como referencia para seguir practicando Python.

Guía de Usuario – PyLearningHub

Descripción General

PyLearningHub es un curso interactivo de Python desarrollado para ejecutarse en Google Colab, que te llevará desde conceptos básicos hasta temas avanzados del lenguaje, combinando teoría, práctica y ejercicios evaluativos.

Esta guía explica **cómo usar el curso paso a paso**, desde el inicio hasta el proyecto final.

Requisitos Previos

Antes de comenzar, asegúrate de tener:

- Una cuenta de Google activa.
- Acceso a Google Colab.
- Conexión a internet estable.

Estructura del Curso

El curso está dividido en **seis lecciones** numeradas y tituladas:

1. Variables y Tipos de Datos
2. Operadores y Estructuras Condicionales
3. Funciones y Módulos
4. Manejo de Excepciones
5. Programación Orientada a Objetos (POO)
6. Uso Avanzado del Lenguaje

Cada lección incluye:

- Introducción teórica.
- Explicaciones claras con ejemplos prácticos.
- Código interactivo que debes ejecutar.
- Preguntas evaluativas con `input()` para validar lo aprendido.
- Registro automático de tu progreso para saber qué lecciones has aprobado.

Cómo Usar el Curso

1. Abrir el archivo en Google Colab

1. Accede a Google Colab.
2. Sube o abre el archivo `.ipynb` que contiene el curso (puedes obtenerlo desde GitHub o desde tu equipo).
3. Ejecuta cada celda secuencialmente desde la **primera celda** para evitar errores y asegurar el correcto funcionamiento del curso.

---

2. Iniciar el Curso

- Al ejecutar la **primera celda**, se mostrará un mensaje de bienvenida.
- Se solicitará ingresar tu **nombre**; este será usado para registrar tu progreso en el curso dentro de una estructura interna.
- Se inicializa un diccionario `progreso` para hacer seguimiento de tus avances.

---

3. Avanzar por las Lecciones

- Cada lección es una función que debes ejecutar llamándola con tu nombre como argumento, por ejemplo:

```
leccion_1_variables(nombre_alumno)
```

- Dentro de cada lección encontrarás explicaciones, código para ejecutar y evaluaciones interactivas.
- Al aprobar la evaluación, el sistema actualizará automáticamente tu progreso.

---

4. Verificar el Progreso

- Hay una función llamada `mostrar_progreso(nombre_alumno)` que puedes ejecutar en cualquier momento para ver:
  - Qué lecciones has aprobado.
  - Cuántas lecciones faltan para completar el curso.

---

5. Proyecto Final

- Para acceder al proyecto final, debes haber aprobado **todas las lecciones** del curso.
- El proyecto final te propondrá crear un programa completo que integre los conceptos aprendidos, incluyendo:
  - Definir al menos una clase con atributos y métodos.
  - Utilizar al menos una función externa a la clase.
  - Implementar manejo básico de errores con `try-except`.
  - Usar listas o diccionarios para manejar datos.
  - Interactuar con el usuario mediante `input()`.

---

6. Guardar y Compartir tu Trabajo

- Puedes guardar tu progreso y trabajo desde Google Colab:
  - Selecciona “Archivo” > “Guardar una copia en Drive” para guardar tu avance en tu Google Drive personal.
- Cuando termines el curso y el proyecto final, puedes compartir tu archivo `.ipynb`:
  - Por Google Drive (compartiendo el enlace).
  - O subiéndolo a GitHub como parte de tu portafolio o entrega.
- Asegúrate de que todas las celdas estén ejecutadas y los resultados visibles para mostrar evidencia de tu trabajo.

Recomendaciones Finales

- **No borres celdas** ni cambies el orden de ejecución para evitar errores inesperados.
- **Ejecuta las celdas en orden**, una por una.
- Si alguna evaluación falla, no te preocupes: vuelve a ejecutar la lección y responde nuevamente.
- Guarda tu progreso frecuentemente para evitar pérdidas.

Créditos

Este curso fue diseñado para facilitar el aprendizaje activo y práctico de Python, fomentando la experimentación y la autoevaluación para que desarrolles habilidades sólidas en programación.

¡Mucho éxito en tu aprendizaje con PyLearningHub!