Primeros pasos en Programación 1 en Python

Este cuaderno te guiará a través de los pasos para comenzar a programar en Python, incluyendo la instalación de Python, la configuración de un entorno de desarrollo, y la ejecución de tu primer programa.

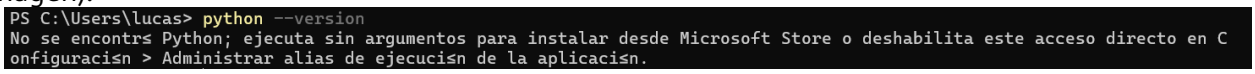
Índice

1. Verificar si Python está instalado

Antes de instalar Python, verifica si ya está instalado en tu sistema. Abre la consola de comandos (puedes buscar `cmd` en el menú de inicio) y escribe el siguiente comando:

```
python --version
```

Si Python está instalado, verás una salida con la versión de Python. Si no, Windows te sugerirá descargarlo desde la Microsoft Store o no reconocerá el comando (como se ve en la siguiente imagen).



Primero es definir la variable numérica en °C

```
# Desafío 3
#Convertir una temperatura dada en Celsius a Fahrenheit y Kelvin
utilizando operaciones directas sobre variables.

#Temperatura en Celsius.
#"Ingresar la temperatura en Celsius"
Celsius = float()

# Conversión a Fahrenheit
#Aplicar fórmula de conversión
Fahrenheit = (Celsius * 1.8) + 32
#Operaciones aplicadas: multiplicación y suma.
```

```
#Conversion a Kelvin
#Aplicar fórmula de conversión
Kelvin= Celsius +273.15
#operación aplicada: suma.

#Validar los resultados
#Imprimir los resultados
print: Celsius = Fahrenheit
print: Celsius = Kelvin
```

Primero identificar las variables a utilizar. Luego usar la opción float ya que se va a trabajar con datos números reales que incluyen decimales. Allí identificar las operaciones aplicadas para convertir Grado Celsius a Fahrenheit. Luego la conversion de Celsius a Kelvin. Imprimir los resultados a partir de las conversiones estableciendo las relaciones.

Webgrafía utilizada: Tipos_de_datos_y_operadores.ipynb
https://www.newark.com/es/convertidor-temperatura
Estrategias_de_resolucion_de_problemas.ipynb https://offers.hubspot.es/guia-introduccion-python https://peps.python.org/pep-0008/

```
#Desafío 4: Verificar múltiplos de varios números
#Crear un programa que, dado un número, verifique si es múltiplo de:
2, 3, 5, 7, 9, 10 y 11.Imprimir un mensaje por cada verificación.
#Expresar un número entero (int)
número=int (30)
#Realizar lista de divisores
list=[2,3,5,7,9,10,11]
#Verificar múltiplos
#Usar el operador % para comprobar.
for divisor in list:
  número=30
  divisor=2
  if  (número % divisor) == 0:
    print(f"{número} es múltiplo de {divisor}")

  número:30
  divisor:3
  if  (número % divisor) == 0:
    print (f"{número} es múltiplo de {divisor}")

  número:30
  divisor:5
  if  (número % divisor) == 0:
    print(f"{número} es múltiplo de {divisor}")

    número :30
    divisor:7
     else:
```

```
      print(f"{número} no es múltiplo de {divisor}")

      número:30
      divisor:9
      else:
        print(f"{número} no es múltiplo de {divisor}")
```

```
  File "<ipython-input-224-879775133c70>", line 27
    else: (número % divisor) != 0.
    ^
IndentationError: unexpected indent
```

Lo primero fue seleccionar el número entero,que es expresado por el código: int. Luego realicé la lista de divisores (list) Al momento de expresar la operación tilicé (if) para indicar que el número es múltiplo y el comando (else) para marcar que no es múltiplo. Analizando la propuesta, se percibe que hubo errores al programar. Lo intenté varias veces leyendo diferntes materiales pero no logré resolver el desafío totalmente. Igualmente en la propeusta quedaron registrados los intentos. la confusión se generó al momento de imprimir cada múltiplo. Hay simbología que todavía no logro interpretar y que al incorporarlas marcan el error. La cambié en varias oportunidades, analiando la letra de la situación.

Webgrafía utilizada: 2_4_Tipos_de_datos_y_operadores.ipynb30 KB
Estrategias_de_resolucion_de_problemas.ipynb
https://www.tokioschool.com/noticias/operadores-python/
https://www.cs.upc.edu/~jsierra/p1.pdf

```

```

> Añadir blockquote

```

```

2. Instalar Python

Si Python no está instalado, sigue estos pasos para instalarlo:

- Visita el sitio web oficial de Python en https://www.python.org/downloads/ y descarga la última versión para Windows.
- Ejecuta el instalador descargado. **Importante:** Asegúrate de marcar la opción `Add Python X.X to PATH` antes de hacer clic en `Install Now`.
- Para verificar que se haya instalado en su ultima versión, repetir el paso 1 de nuevo.

3. Instalar Visual Studio Code (VSCode)

Visual Studio Code (VSCode) es un editor de código fuente que soporta múltiples lenguajes de programación. Para instalarlo:

- Descarga VSCode desde https://code.visualstudio.com/ y ejecuta el instalador.

- Abre VSCode, ve a la sección de extensiones (`Ctrl+Shift+X`), busca y instala la extensión `Python` de Microsoft.

4. Métodos alternativos sin instalar VSCode

Si prefieres no instalar un IDE o editor de código por ahora, puedes utilizar IDLE (el IDE que viene con Python) o cualquier editor de texto como Notepad para escribir tu código Python y ejecutarlo desde la consola de comandos.

Usando IDLE:

- Busca `IDLE` en el menú de inicio para abrirlo, escribe tu código Python, y selecciona `Run -> Run Module` para ejecutarlo.

Usando el Bloc de notas:

- Abre el Bloc de notas, escribe tu código Python, guárdalo con la extensión `.py`, por ejemplo, `mi_programa.py`, y sigue las instrucciones de la sección 6 para ejecutarlo desde la consola.

```
print('Hola Curso 2024')
```

```
Hola Curso 2024
```

5. Crear y ejecutar tu primer programa en Python

Para tu primer programa en Python, escribirás un simple script que imprima `Hola Curso 2024` en la consola. Independientemente del editor o método que elijas, el código será el mismo:

6. Comprobación y ejecución desde la consola

Para ejecutar tu programa desde la consola de comandos, sigue estos pasos:

- Abre la consola de comandos (`cmd` en el menú de inicio).
- Navega al directorio donde guardaste `hola_curso.py`. Puedes usar el comando `cd` para cambiar de directorio, por ejemplo, `cd Desktop` para ir al escritorio.
- Escribe el siguiente comando y presiona Enter:

```
python hola_curso.py
```

Verás `Hola Curso 2024` impreso en la consola. ¡Felicidades! Ahora estás listo para comenzar tu viaje en la programación con Python.

7. Crear cuenta en GitHub

GitHub es una plataforma esencial para el desarrollo de software que facilita la colaboración y el versionado de código entre desarrolladores. Crear una cuenta en GitHub te permitirá alojar tus proyectos, colaborar con otros y contribuir a proyectos de código abierto. Es una herramienta valiosa no solo para guardar y compartir tu código, sino también para aprender de la comunidad global de desarrolladores.

Para crear tu cuenta, visita GitHub y regístrate utilizando tu correo electrónico. Considera utilizar tu nombre real o una combinación de tu nombre y apellido como nombre de usuario para que sea reconocible y profesional.

8. Otros IDE para trabajar con Python

Existen varios entornos de desarrollo integrados (IDE) que permiten escribir y ejecutar código en Python. Algunos de los más utilizados son:

- **PyCharm**: Muy popular, desarrollado por JetBrains. Tiene una versión gratuita llamada *PyCharm Community* y otra de pago. La versión Community permite trabajar con proyectos Python, pero **no permite trabajar directamente con archivos Jupyter Notebook (.ipynb)**.
- **JupyterLab**: Una evolución de Jupyter Notebook, muy completa y personalizable para quienes prefieren un entorno local.
- **Thonny**: Ideal para principiantes, con una interfaz simple y amigable.
- **Google Colab**: Basado en la nube, permite trabajar con archivos Jupyter Notebook sin necesidad de instalar nada en tu equipo.

  Nota: Si deseas trabajar con archivos `.ipynb` en PyCharm, necesitarás la versión profesional de pago. Sin embargo, la versión Community es perfecta para proyectos Python estándar.