**PYTHON: Pseudocode**

Es una práctica de **codificación informal**, que no está vinculado a ningún lenguaje de programación específico como Python, JavaScript o C #, sino que **ayuda** a los programadores a **planificar algoritmos** y **entrenar el pensamiento programático**.

Mediante el uso de pseudocódigo, puede planificar cada paso de su programa sin preocuparse por la sintaxis.

Se trata de **dividir el problema en pasos** y **palabras más simples** (no deberíamos apuntar a utilizar expresiones técnicas, sino un lenguaje conversacional simple).

Se divide en DOS PASOS:

1. Pensar en **cómo abordar el problema** (qué estrategia utilizar, ya que a menudo un problema se puede resolver de muchas formas diferentes).
2. Definir **qué elementos necesitaremos** en cada paso (bucles, listas, diccionarios, variables, condicionales, etc.).

→ THINK IN ENGLISH (ya que los lenguajes serán en ese idioma)

**Ej 1: For** every element in this **list** I want to compute the square of each element.

→ Vemos dos palabras clave:

* FOR → FOR LOOP
* LIST → List variable

**Ej 2:** As long as the runner hasn't reached the finish line, keep running**.**

→ hay una condición sobre la parada, pero no sabemos cuándo será así que usaremos un WHILE LOOP

→ Tendríamos que crear dos variables: una para almacenar el estado del corredor al principio, y otra para almacenar su progreso.

Do's & Don'ts

***Do's:***

- Use control structure

- Use proper naming conventions

- **Indentations** and **white spaces** are the key

- Keep it simple

- Keep it concise

***Don'ts***

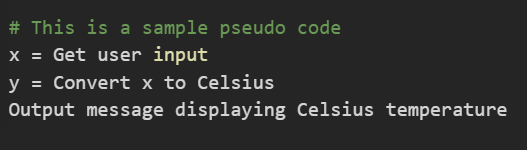
- Don't make your pseudo-code abstract

- Don't be too generalized

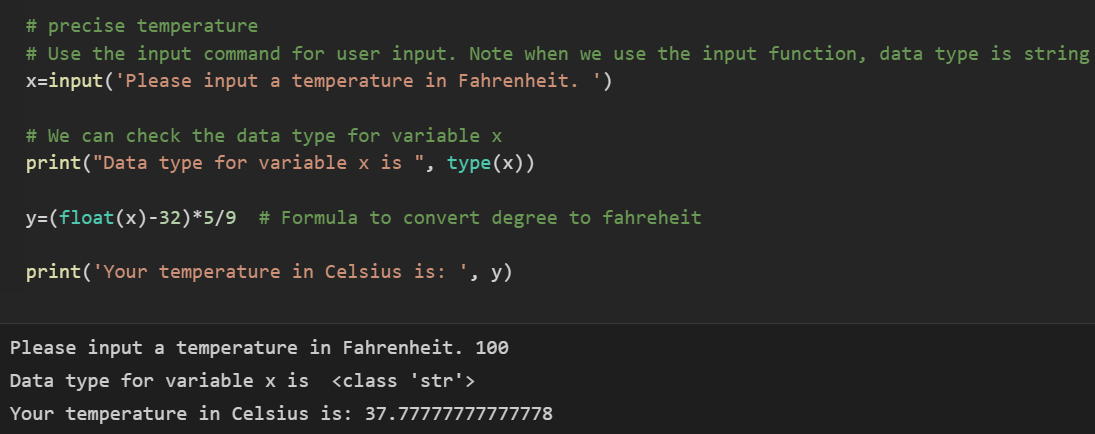
EJEMPLOS:

1. Write a program that **asks the user** for a temperature in Fahrenheit and **prints out** thesame temperature in Celsius.

* Pseudocode:

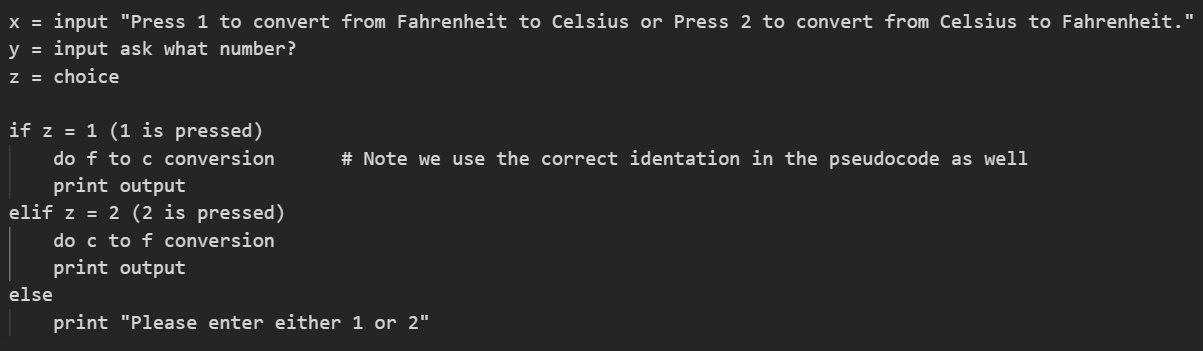
****

* Code:

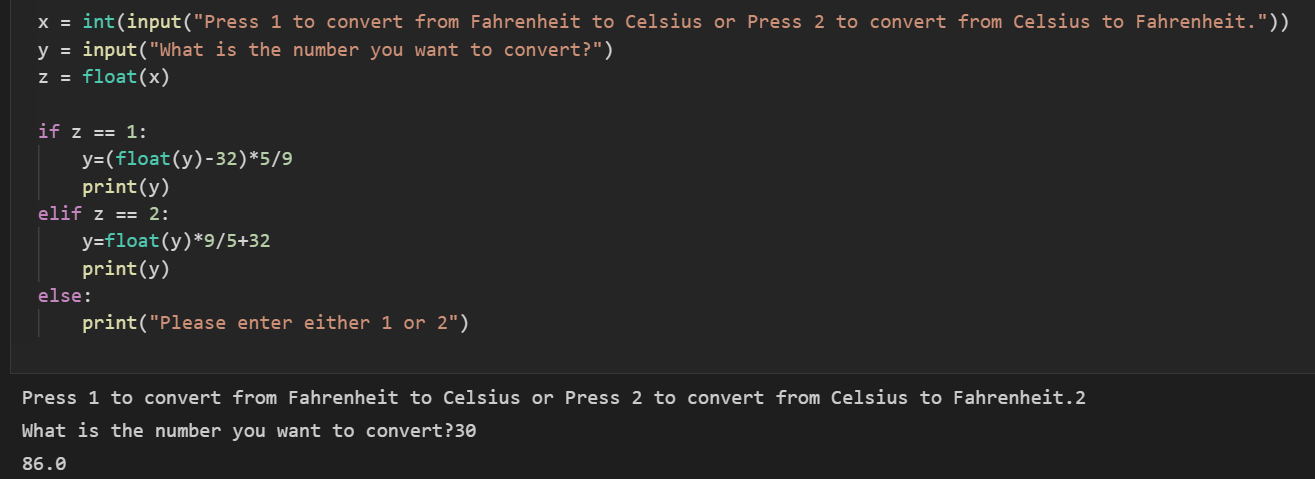
****

2. Write a program that converts from Fahrenheit to Celsius or from Celsius to Fahrenheit, depending on the user's choice.

* Pseudocode:



* Code:

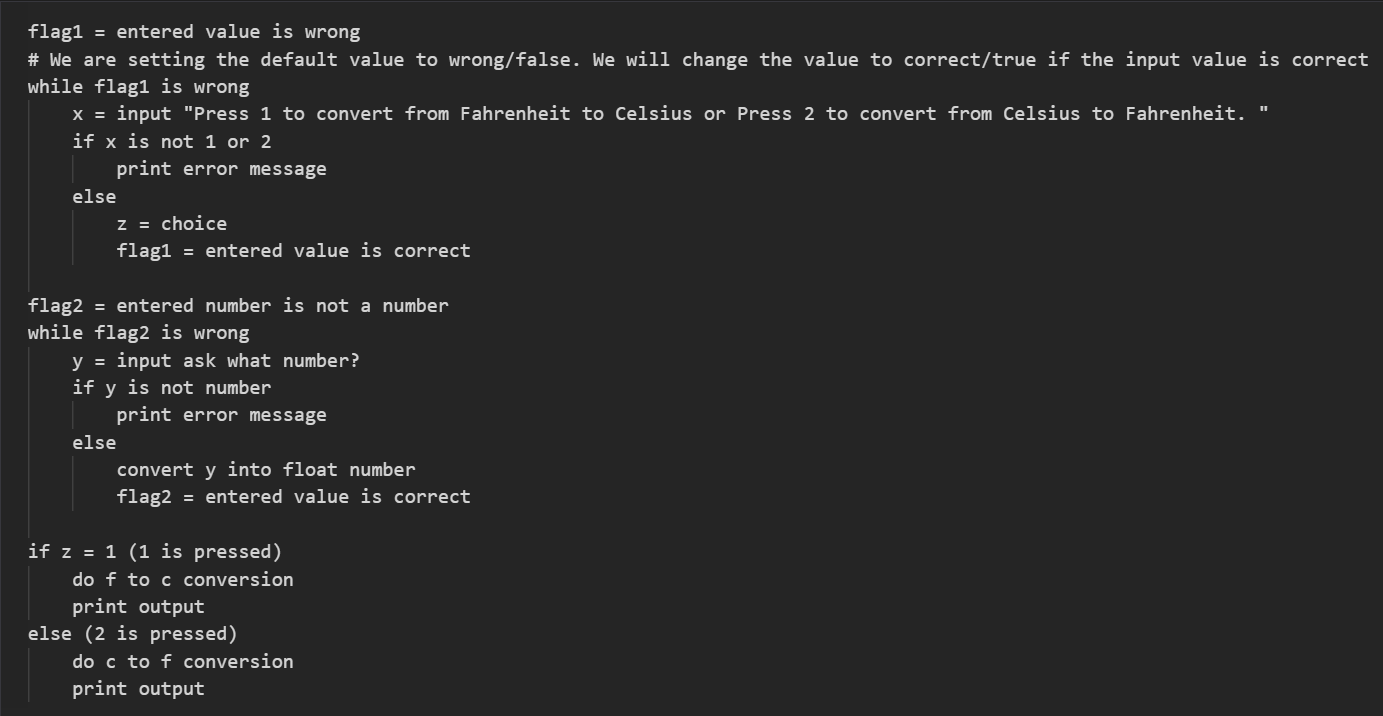


Hacer el código MÁS ROBUSTO

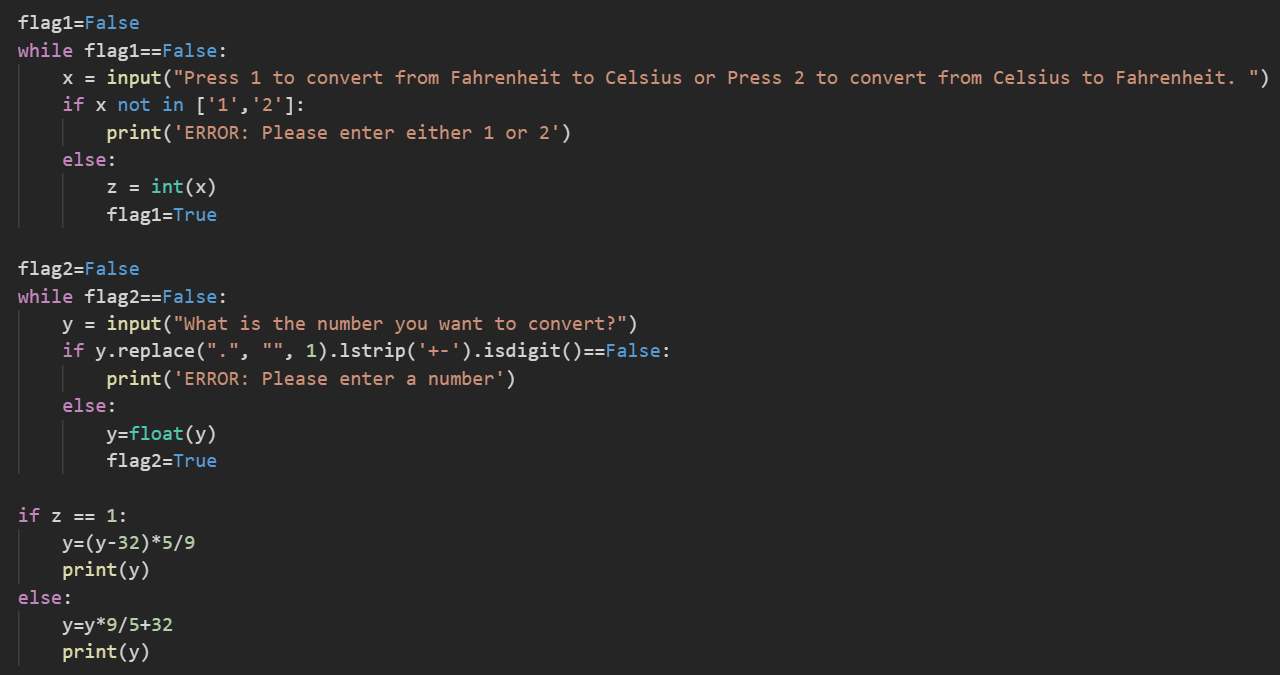
En el ejemplo anterior, si el usuario no pone 1 o 2 el código daría error y se detendría. Para tener éxito con la solicitud y mejorar la experiencia del usuario, debemos **seguir pidiendo el valor hasta recibir el correcto**. Además de eso, debemos **evitar errores de compilación**.

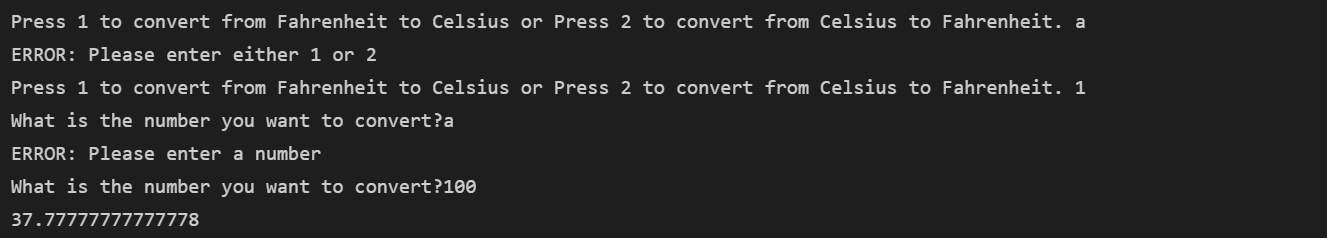
Además, la función float() solo puede recibir números, ¿qué pasa si el usuario pone letras?, el código dará error.

* Pseudocode:



* Code:





→ Creamos flags gracias a los flags y creamos condicionantes que van a ser loops con **while**. Cada bloque lo tenemos que cerrar poniendo en true el flag (me recuerda un poco al set interval de js)

**flag1= False**

while flag1==false:

#aqui metemos la acción en loop

**flag 1=True**