


Open · Opened 4 days ago by  **Rubén Montero**

# Si he dicho a, b ó c... Es a, b ó c

## Resumen

- Usaremos un `while` para verificar la entrada del usuario en `contest.py`

## Descripción

En la tarea anterior escribimos un fichero `contest.py` que pregunta al usuario cuál es la universidad más antigua de Europa.

Le da tres opciones, **a**), **b**) ó **c**), y lee la entrada con:

```
choice = input("Escoge tu respuesta (a, b ó c): ")
```

### ¿Y qué pasa si elijo d)?

Si tecleas **d** u otra opción inválida, como **3** ó **pato**, saltará el mensaje de *fallo* del `else`.

### También podemos engañar al usuario sin querer

¿Qué pasa si el usuario teclea **B** (en mayúsculas) ó **b**) (con el paréntesis ))? El código **saltará** al `else`, y, ¡puede ser **inesperado** para el usuario!

## Bucle `while`

Existe en prácticamente todos los lenguajes de programación y Python no es una excepción. Su sintaxis es:

```
while condition:
    # Hacer cosas DENTRO del bucle

# A este nivel de indentación, ya hemos salido del bucle
print("El bucle ha terminado")
```

Mientras `condition` sea `True`, el cuerpo del bucle se repetirá.

En el momento en que se evalúe `condition` y sea `False`, el bucle **no** se repetirá.

### ¿`True`? ¿No es `true`?

¡Ah!

**No**. En Python, los valores booleanos básicos se expresan con letra **mayúscula**: `True`, `False`. A diferencia de Java: `true`, `false`.

O sea, si escribes `true`, estarás refiriéndote a una *variable* de nombre `true`.

## A lo que íbamos

Es muy sencillo.

Podemos verificar que la entrada es **a**, **b** ó **c**:

```
choice = input("Escoge tu respuesta (a, b ó c): ")
while choice != 'a' and choice != 'b' and choice != 'c':
    choice = input("Por favor, escoge una respuesta válida (a, b ó c): ")
```

## Alternativas

Si recuerdas las [leyes de Morgan](#), sabrás que:

```
choice != 'a' and choice != 'b' and choice != 'c'
```

...es equivalente a:

```
not (choice == 'a' or choice == 'b' or choice == 'c')
```

También, para ir *introduciendo* conceptos avanzados de *listas* y *strings*, podemos exponer estas otras alternativas:

```
while not (choice in ['a', 'b', 'c']):  
while choice not in ['a', 'b', 'c']:  
while not (choice in 'abc'): # Inconveniente: Admite respuesta 'ab', 'ac', 'bc' y 'abc'  
while choice not in 'abc': # Inconveniente: Admite respuesta 'ab', 'ac', 'bc' y 'abc'
```

## La tarea

Deja escrito en tu `contest.py` la validación de entrada mediante `while` como se expuso más arriba.

Usa tu opción preferida como condición de salida.

Lanza `contest.py` y pruébalo. ¿Valida la entrada como esperas?

## Por último

Verifica que tu código pasará el *test* asociado a la tarea correctamente.

Haz `commit` y `push` para subir los cambios al repositorio.



Rubén Montero @ruben.montero changed milestone to [%Sprint 3](#) 4 days ago