`if-then-else` Statements

`if-then` Statements

- General form of an **if-then** statement:

```
if [boolean expression "test"]:
  [then block – runs when test is True]
```

- **if-then** is a ***control statement***
  - It can be written *anywhere* you can write any other statement
  - It is like a conditional phrase at the beginning of a sentence (and does not end in a semi-colon)

- The "**test**" in must be a **`boolean expression`**

- Statements in the "**then block**" will run ***if*** the **test** evaluates to **true**.
  Else, the processor *jumps* over the then block.

- All code inside the then block must be indented one level deeper than the if.

Example Setup

In VSCode:

1. Expand Explorer
   - Open lessons
   - Right click lessons, new file... name: **`ls10_if_else.py`**

1. Enter the code to the right

2. Open a new Terminal, run:

`python -m lessons.ls10_if_else`

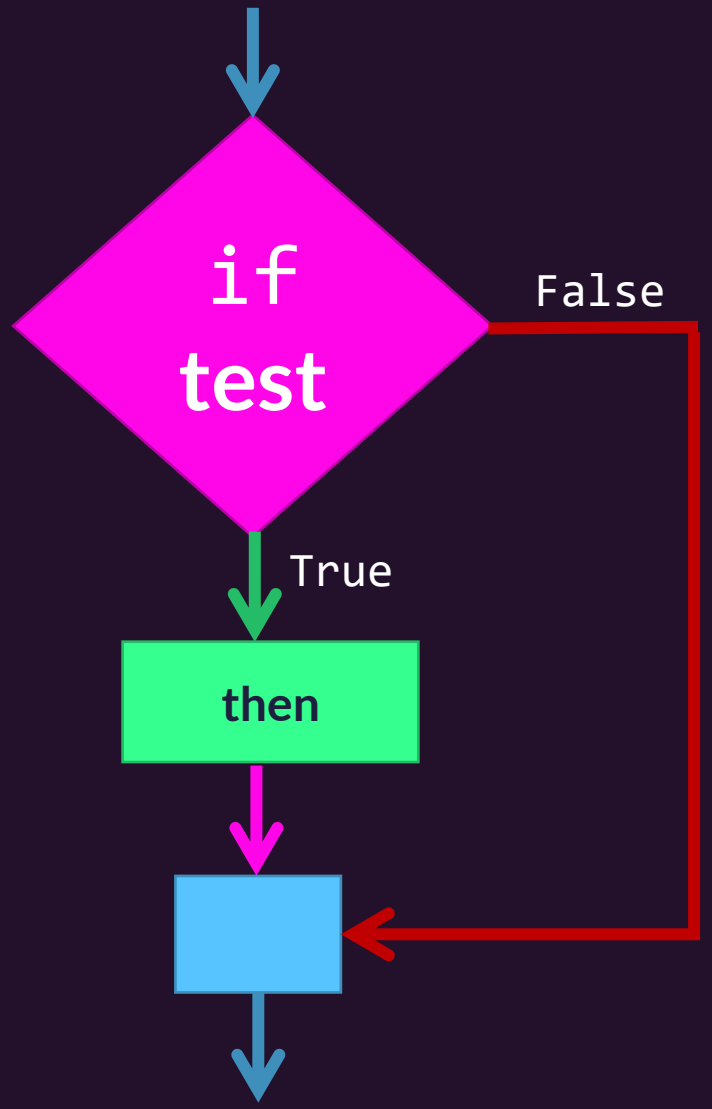
Try entering a guess of *41*, then try again with a guess of *42*.

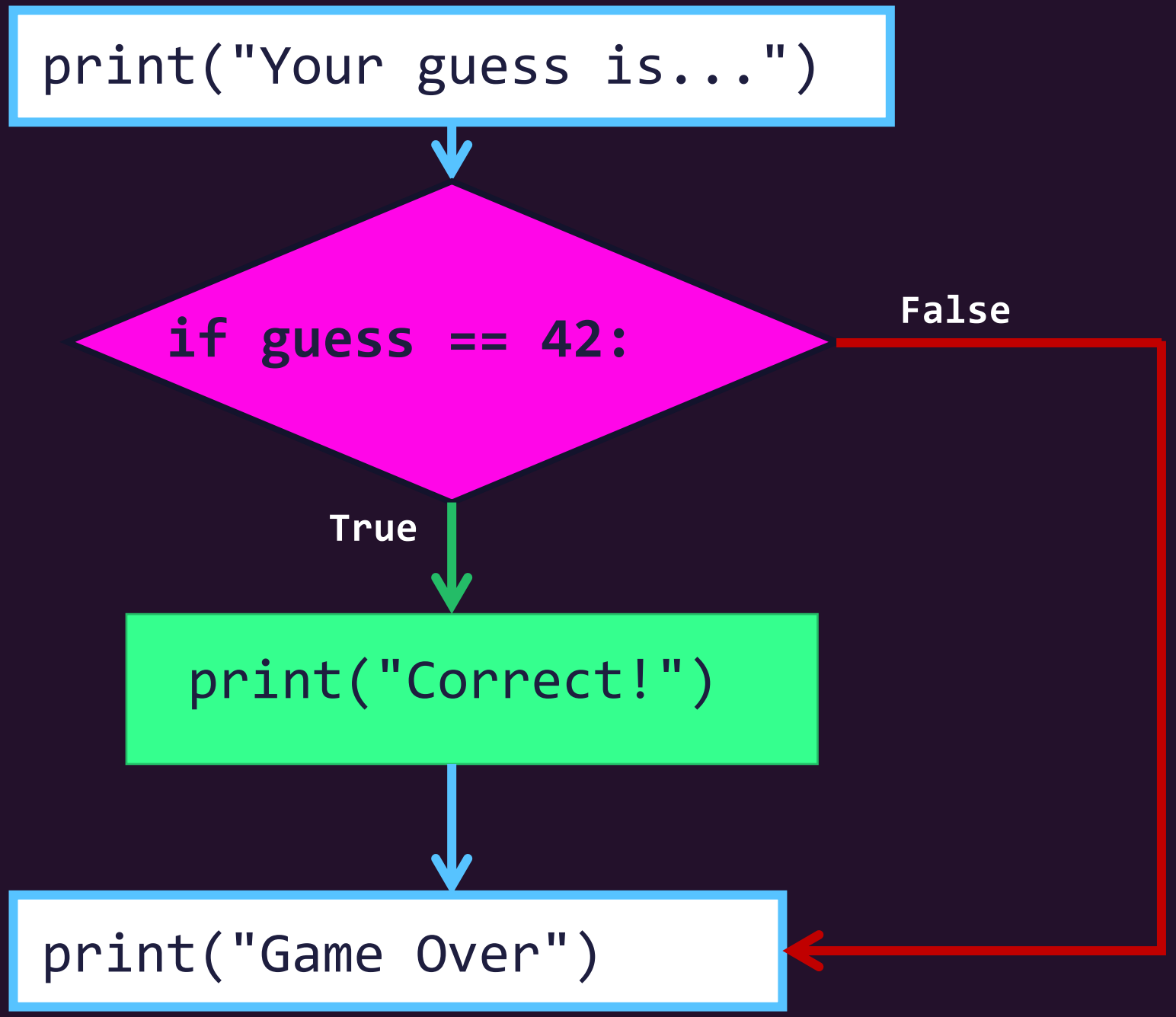
```
print("Guess a number...")

guess: int = int(input("Guess: "))

if guess == 42:
  print("Correct")

print("Game Over")
```

`if-then` Statements

- In a flow chart *("control flow")* we draw an `if-then` statement as a diamond.

- It will have two arrows coming out. We label these arrows for the two cases:
  - **`True`** will continue to code in the *then* block
  - **`False`** will continue to code after the *then* block




print("Your guess is...")
if guess == 42:
False
True
print("Correct!")
print("Game Over")

How do we follow a different path when the test condition is **False**?

`if-then-else` Statements

- General form of an **if-then-else** statement:

```
if [boolean expression - "test"]:
  [then block – runs when test is True]
else:
  [else block – runs when test is False]
```

- Works the same as an `if-then` statement, however, when the **test** expression evaluates to **False** the statements within the else block will run.

- Once either block completes, the processor resumes at the line following the else block. Code in the else block also needs to be indented.

Example - Add an **else** clause

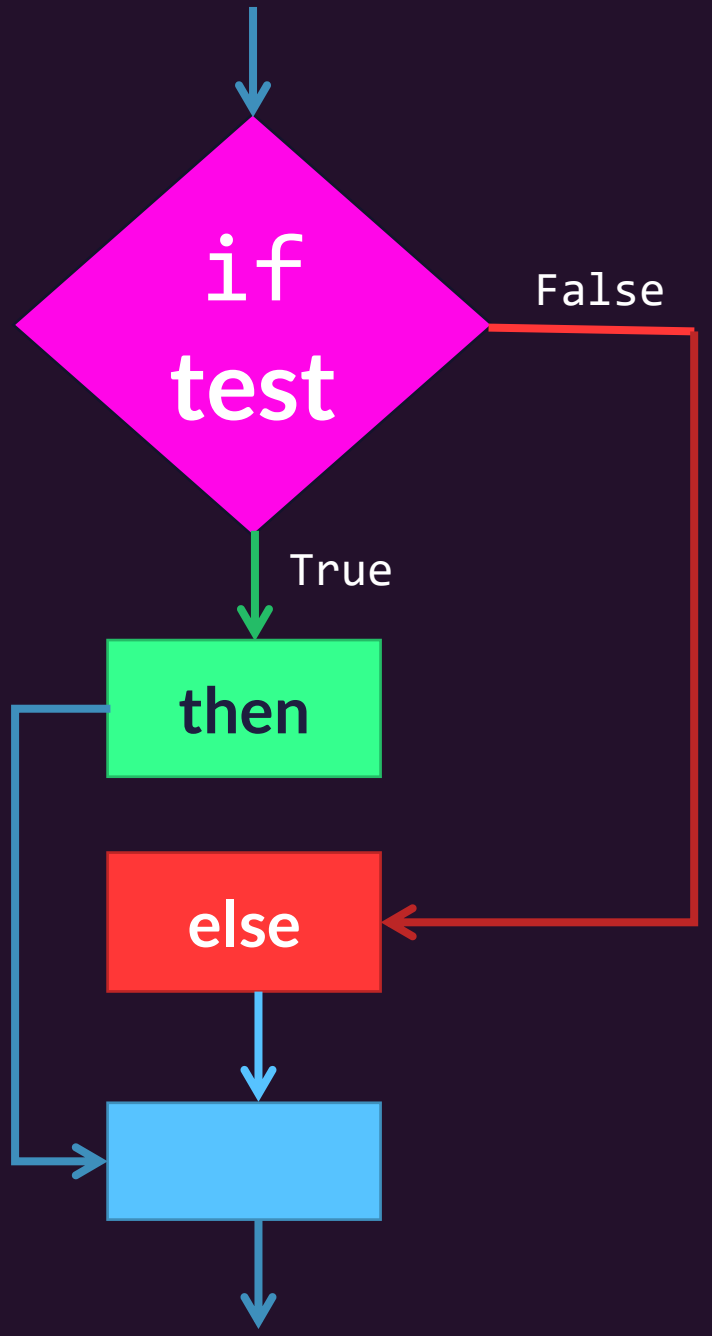
Add an **`else`** clause like the one to the right.

Try playing your game again and entering a correct guess as well as an incorrect guess.

```
if guess == 42:
  print("Correct!")
else:
  print("Nope!")
```

`if-then-else` Statements

- Notice, like the `if-then` statement, the then block runs *only* when the test condition is **True**
- Unlike the if-then statement, the else block runs *only* when the test condition is **False**
- After *either* the then-block or else-block complete, they both continue to the same next step



Nesting if-then-else statements *within* if-then-else statements

- The *then* and *else* *blocks* may contain one or more **statements**…
- …but isn't `if-then` a statement?
  - Yes!
- You can write further `if-then` statements inside of **then** or **else** blocks and the *same* rules apply.

Nested `if-then-else` statement Example

Add the nested if-then-else statement to the right inside of the else block.

Your game should now indicate if the guess was too high or too low!

```
if guess == 42:
   print("Yep!")
else:
   if guess > 42:
      print("Too high!")
   else:
      print("Too low!")
```