

Conditional Execution

To write useful programs, we almost need the ability to check the conditions and change the behaviour accordingly. Conditional statements give us this ability.

The simplest form is the if statement:-

if $x > 0$:
 print (" x is positive ")

??

Reserved word ~~if~~ if $x > 0$ $:$ Condition
print('x is positive')

- The boolean expression after if-statement is called condition.

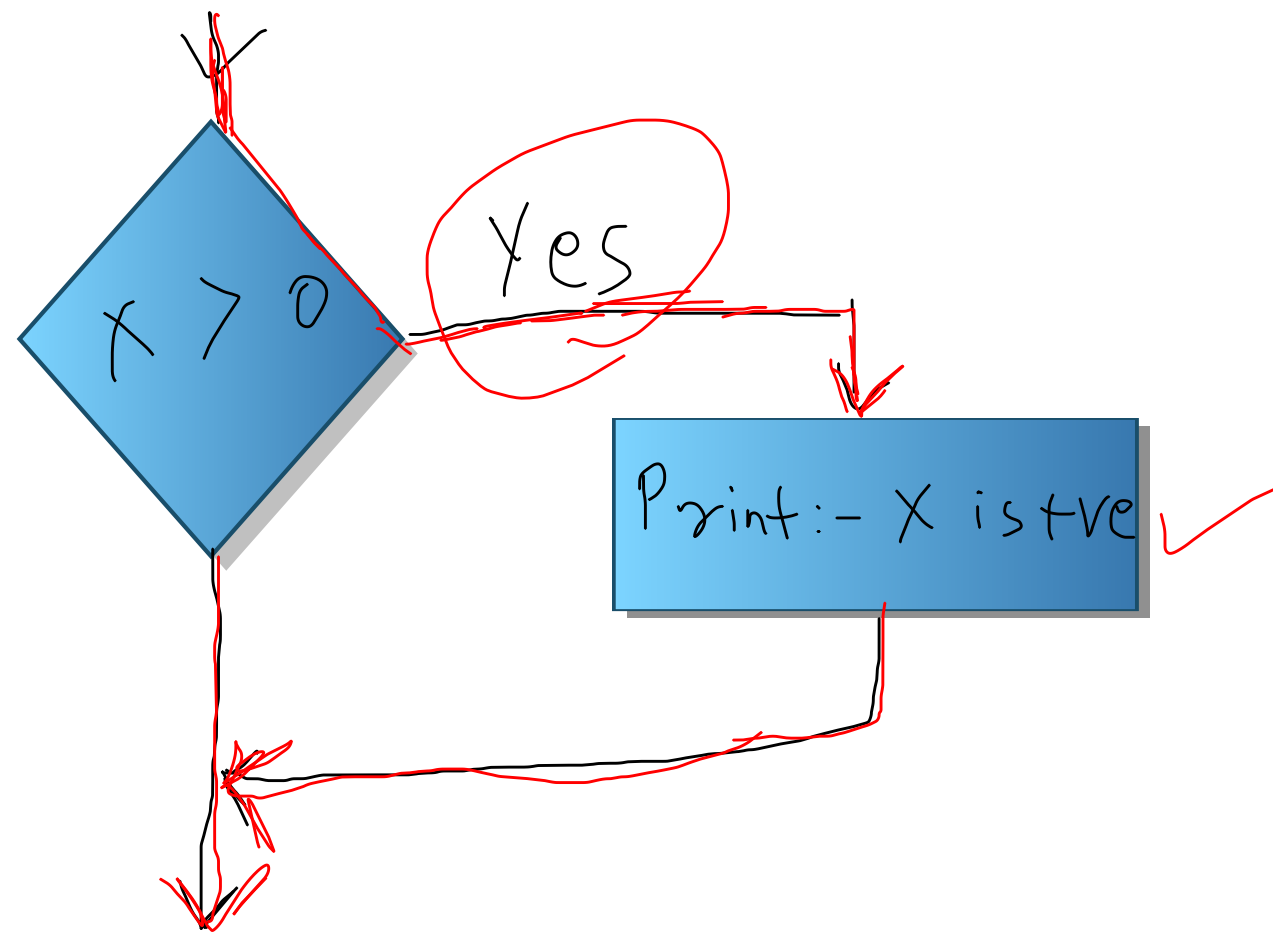
$x = -1$;

$x = 2$;

$x > 0$ \rightarrow False
 $x > 0$ \rightarrow True

- We end the if statement with a colon character (:)

— The line(s) after the if - Statement are indented.
(4 SPACES)



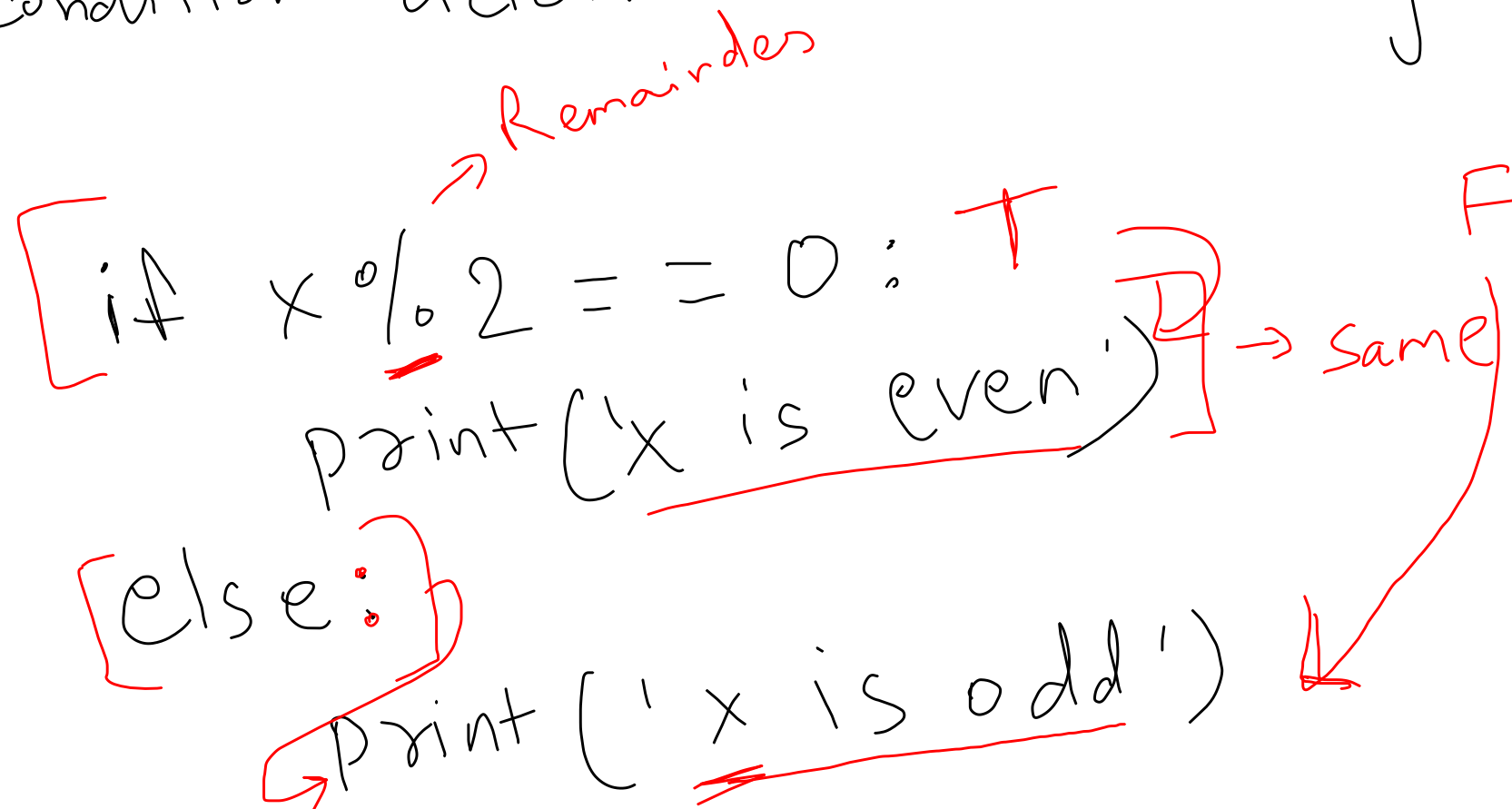
IF - logic

- If logical condition \rightarrow True, indented statement gets executed.
- If logical condition \rightarrow False, indented statement is skipped.
- Statements like this are called compound statements because they stretch across more than one line
- Pass Statement, does nothing!

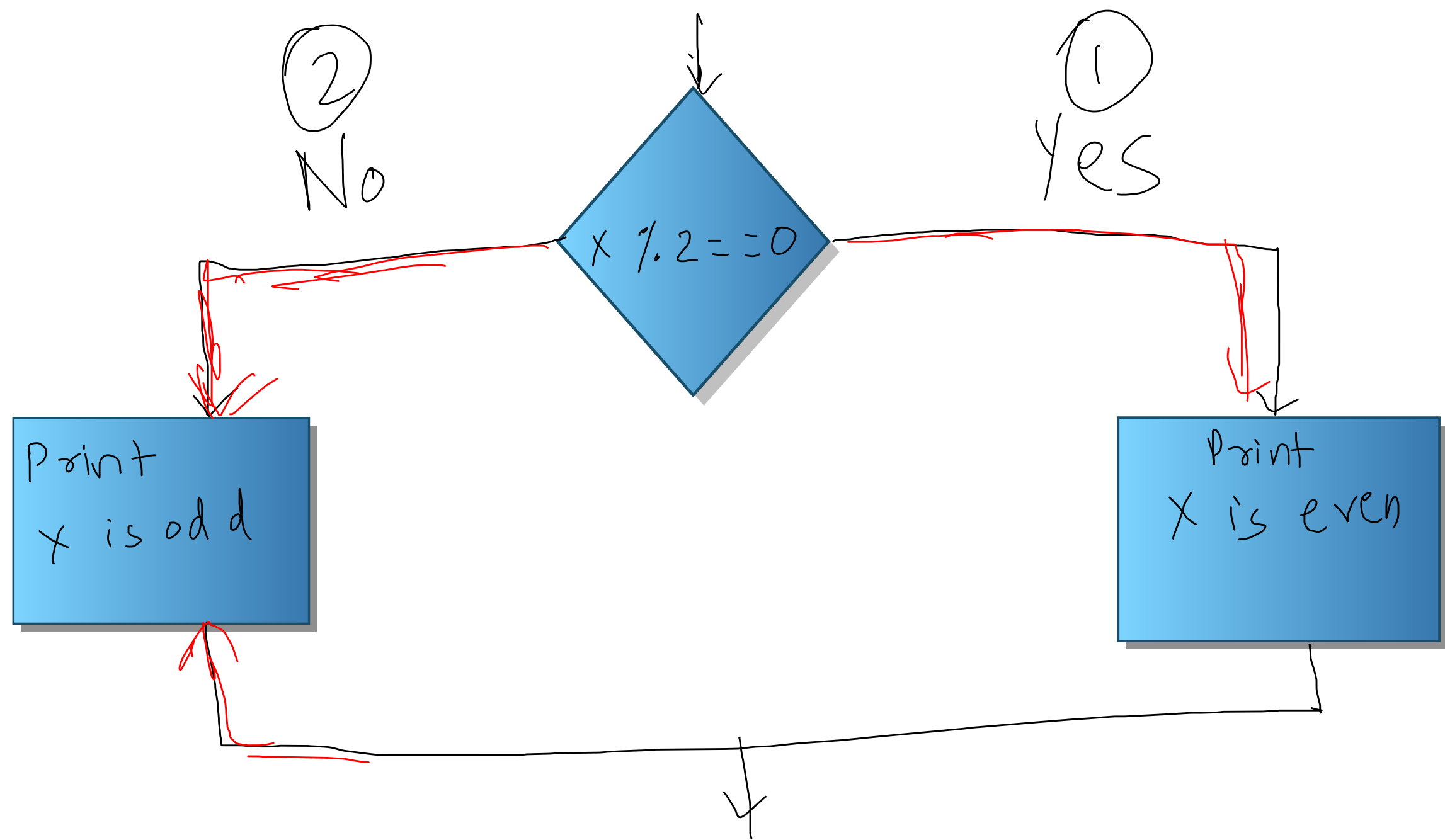
Alternative Execution

→ There are two possibilities.

→ Condition determines which ones gets executed.


The code is written in black ink: `if x % 2 == 0:` followed by `print('x is even')` on the next line, and `else:` followed by `print('x is odd')` on the next line. Red annotations include: an arrow pointing to the `%` operator with the word "Remainder"; a red 'T' above the colon in the first condition; a red 'F' above the second condition; a red bracket on the right side of the code block with an arrow pointing to it and the word "same"; a red arrow pointing from the `else:` line down to the `print('x is odd')` line; and a red arrow pointing from the `print('x is odd')` line down to the right.

```
if x % 2 == 0:
    print('x is even')
else:
    print('x is odd')
```



If - Then - Else Logic ✓

- Since, the condition must either be true or false,
exactly one of the alternatives will be
executed.

- The alternatives are called branches,
because they are branches in the flow of
execution.