Programming with Python

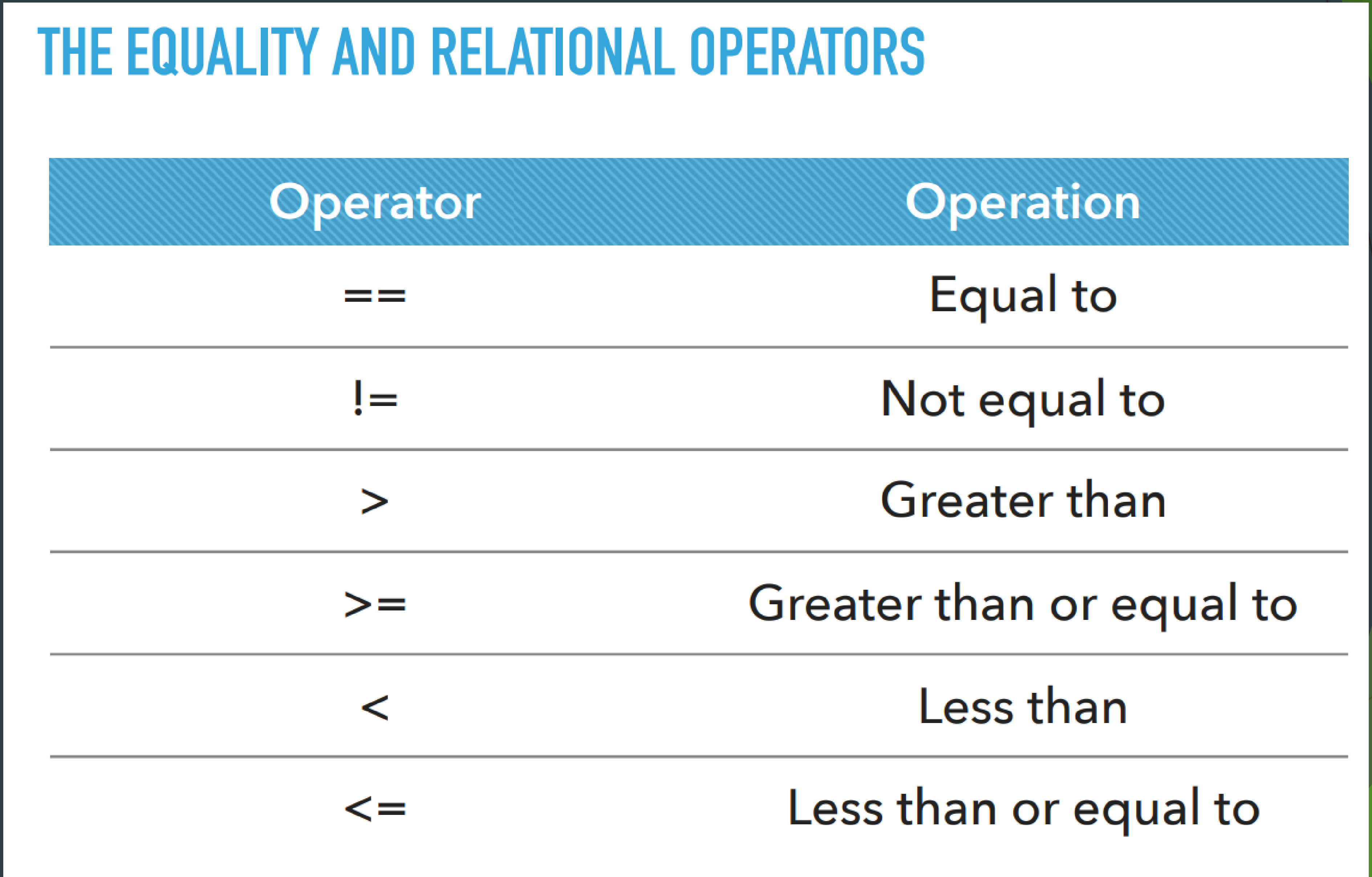
Konstantins Tarasjuks

Agenda for Today

Flow control - IF

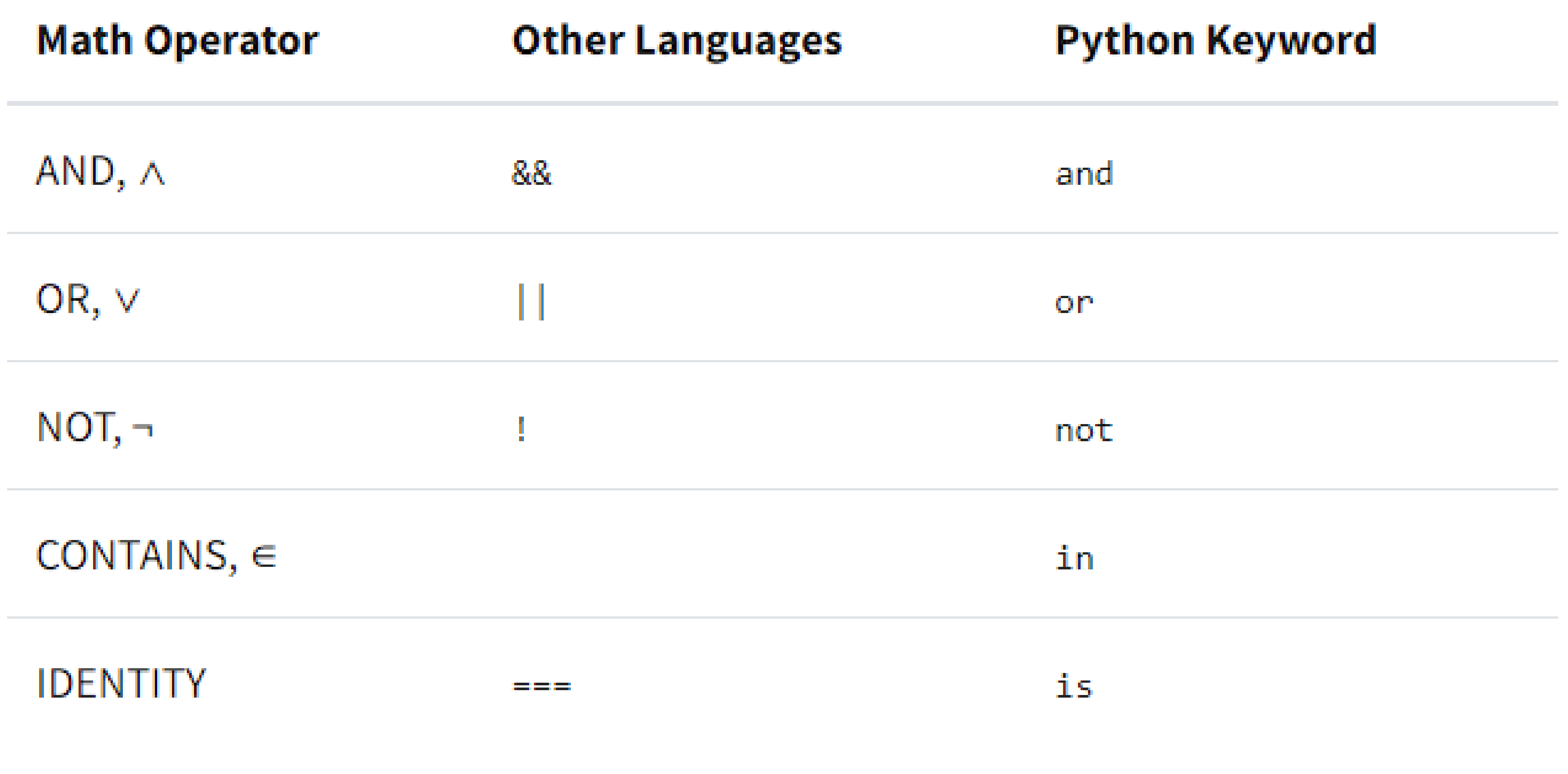
Flow control – Switch

Practice

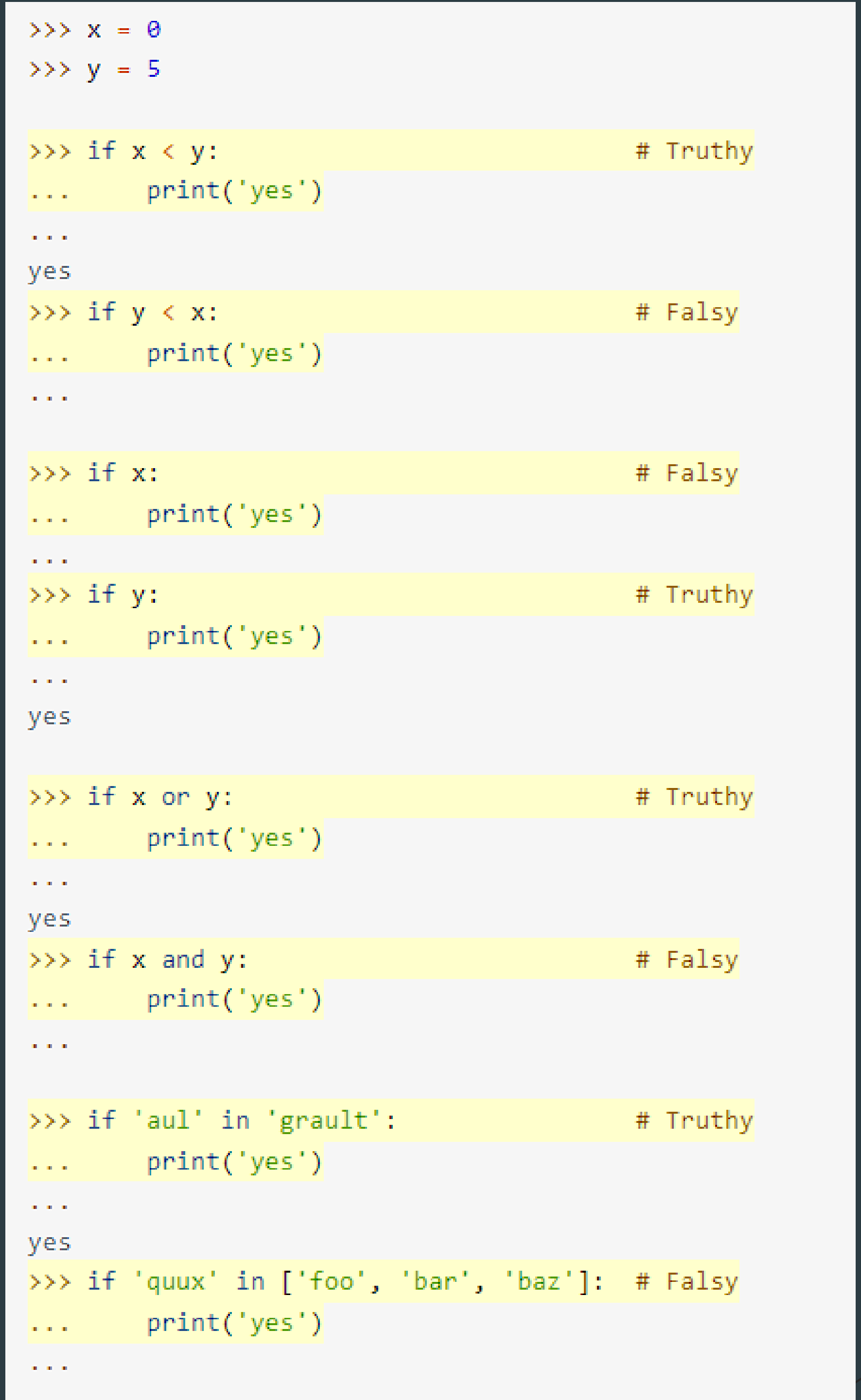


THE EQUALITY AND RELATIONAL OPERATORS

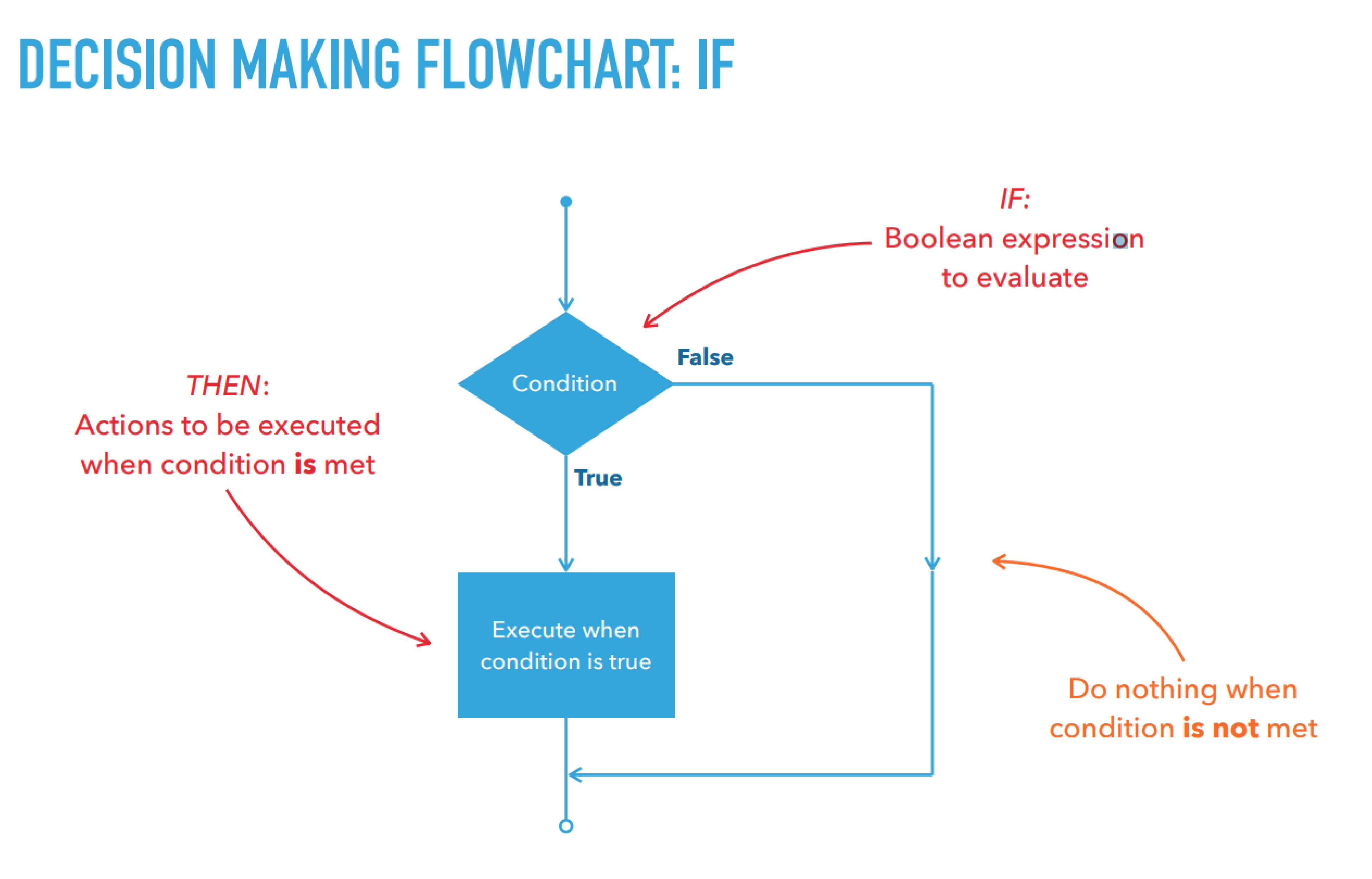
Operator	Operation
==	Equal to
!=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to



Math Operator	Other Languages	Python Keyword
AND, ∧	&&	`and`
OR, ∨	||	`or`
NOT, ¬	!	`not`
CONTAINS, ∈		`in`
IDENTITY	===	`is`



```
>>> x = 0
>>> y = 5

>>> if x < y:                            # Truthy
...     print('yes')
...
yes
>>> if y < x:                            # Falsy
...     print('yes')
...

>>> if x:                                # Falsy
...     print('yes')
...
>>> if y:                                # Truthy
...     print('yes')
...
yes

>>> if x or y:                           # Truthy
...     print('yes')
...
yes
>>> if x and y:                          # Falsy
...     print('yes')
...

>>> if 'aul' in 'grault':                # Truthy
...     print('yes')
...
yes
>>> if 'quux' in ['foo', 'bar', 'baz']:  # Falsy
...     print('yes')
...
```



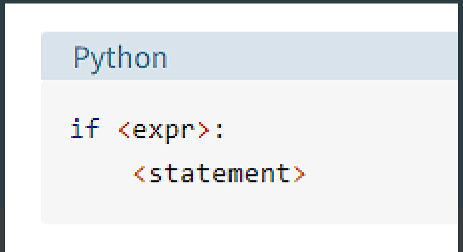
DECISION MAKING FLOWCHART: IF

IF:
Boolean expression
to evaluate

THEN:
Actions to be executed
when condition **is** met

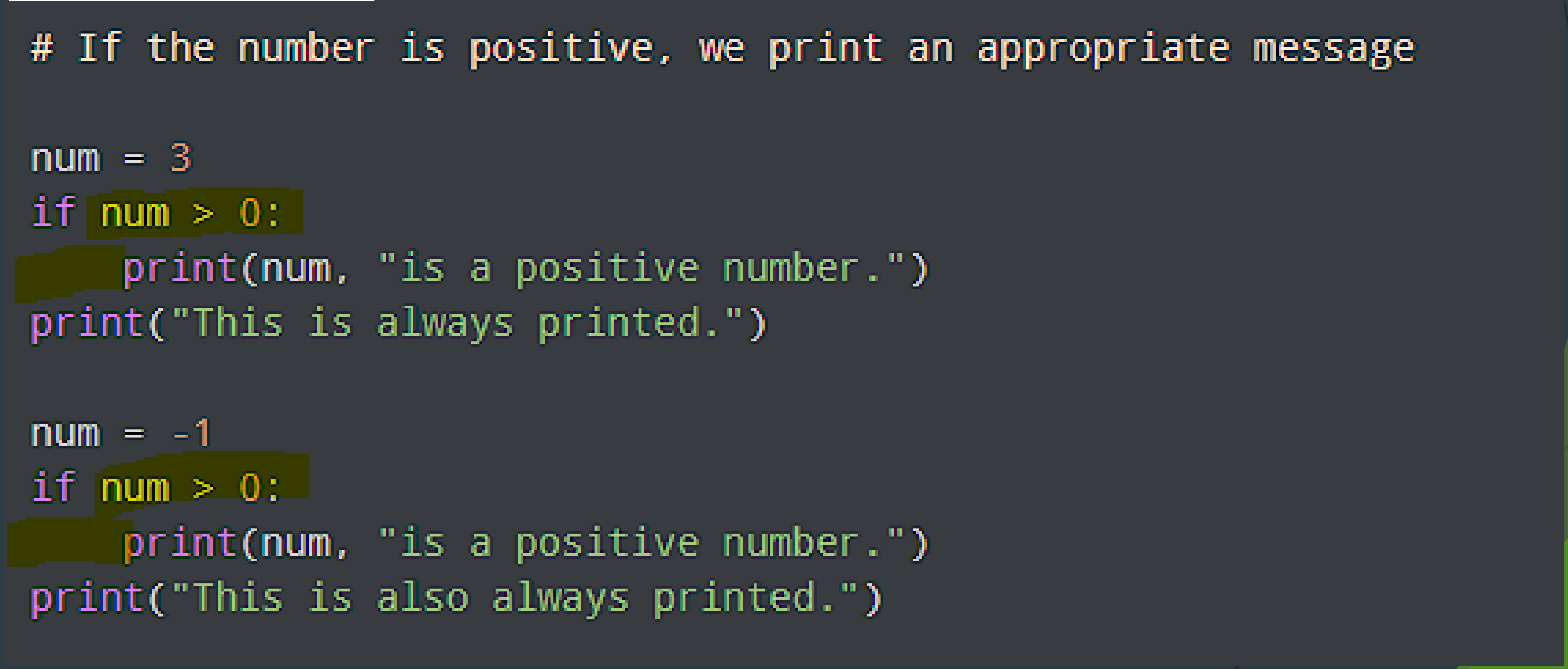
Do nothing when
condition **is not** met

If statement



Python

```
if <expr>:
    <statement>
```

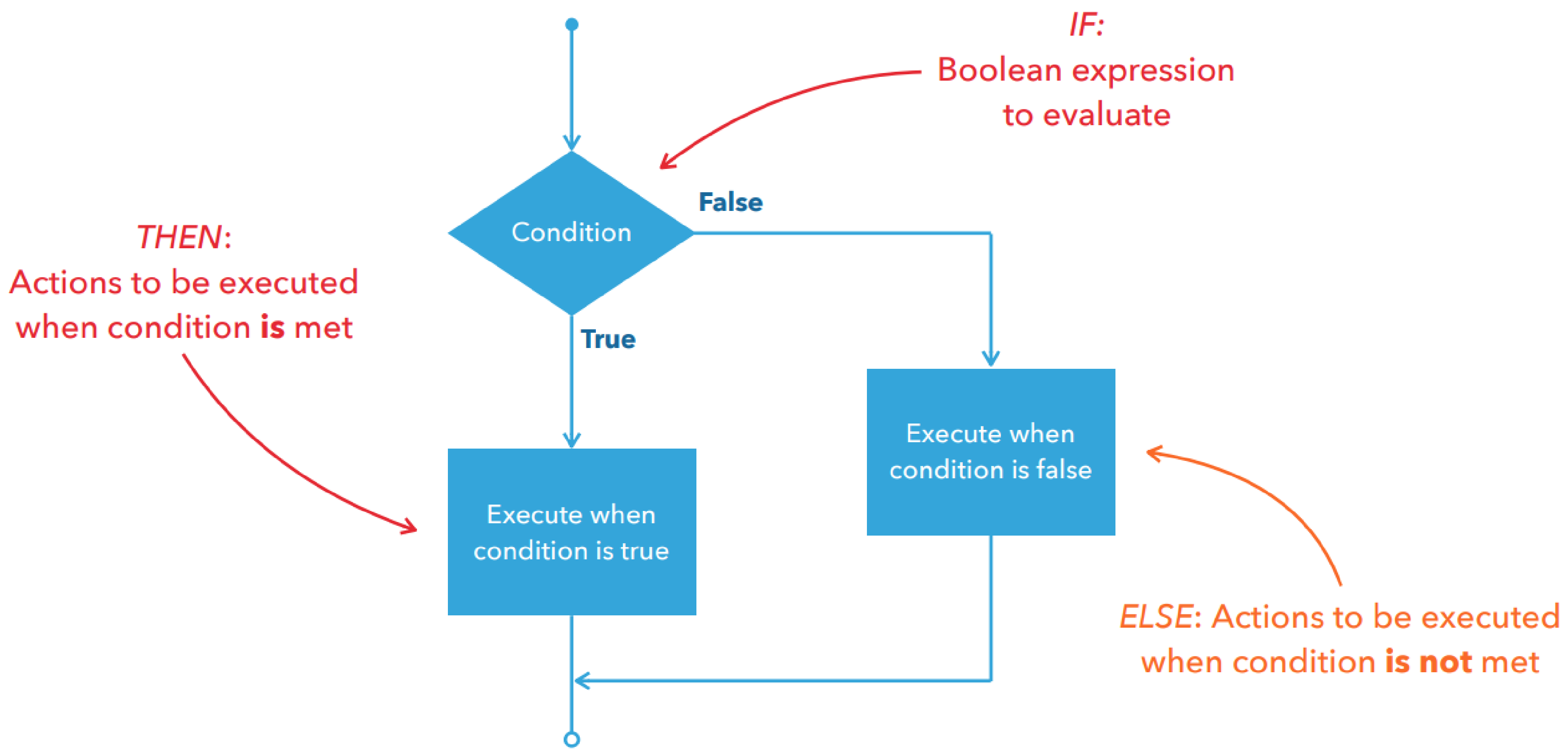


```
# If the number is positive, we print an appropriate message

num = 3
if num > 0:
    print(num, "is a positive number.")
print("This is always printed.")

num = -1
if num > 0:
    print(num, "is a positive number.")
print("This is also always printed.")
```

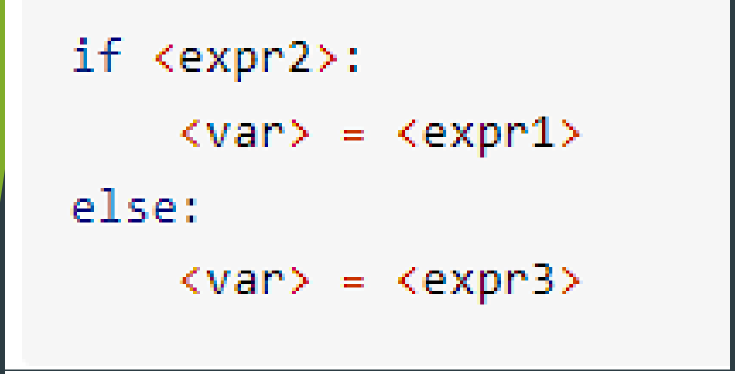
IF STATEMENT RULES RECAP

- Consists of a **boolean expression** followed by **one or more** statements
- Boolean expression can be **composed** of multiple subexpressions

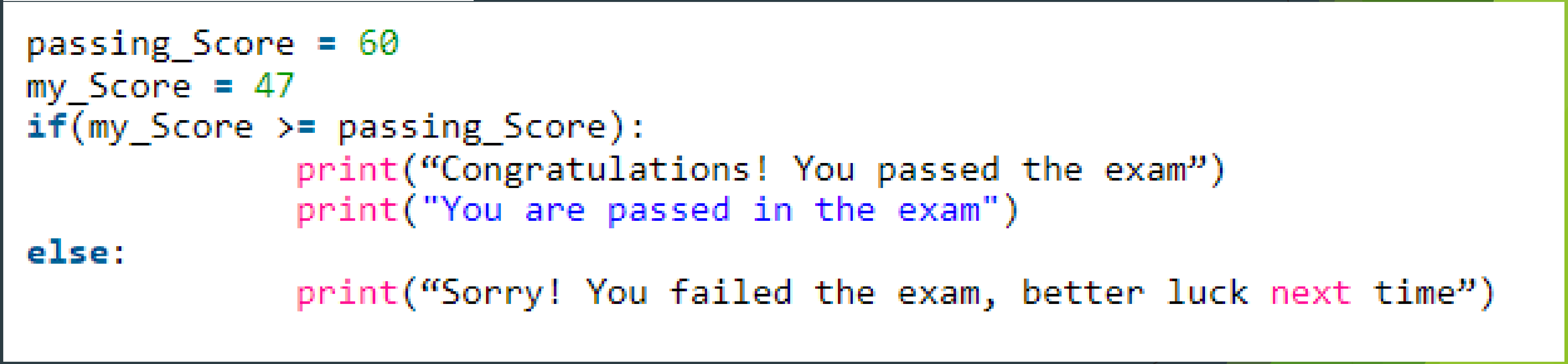
DECISION MAKING FLOWCHART: IF - ELSE



IF – Else statement



```
if <expr2>:
    <var> = <expr1>
else:
    <var> = <expr3>
```

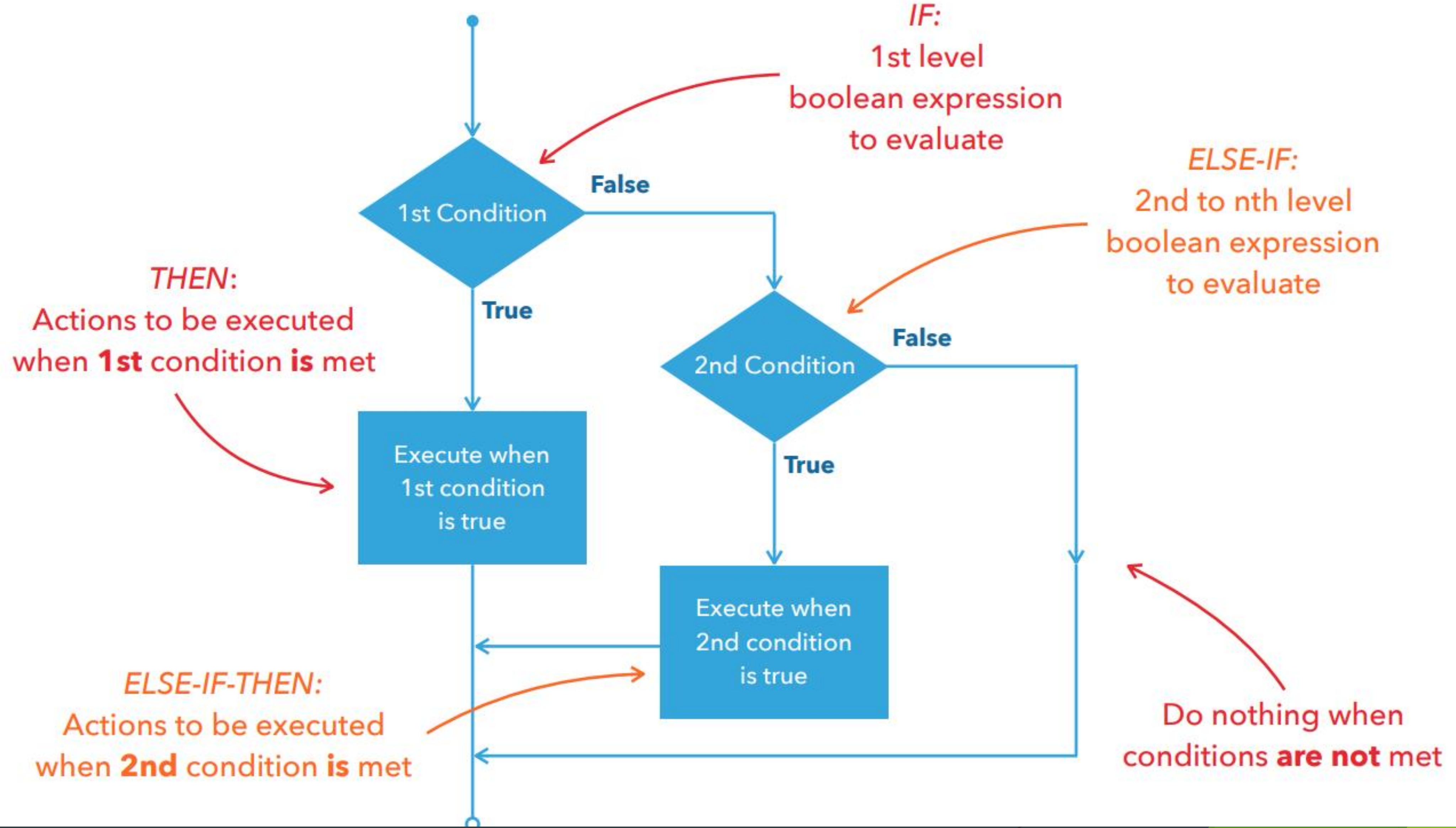


```
passing_Score = 60
my_Score = 47
if(my_Score >= passing_Score):
            print(“Congratulations! You passed the exam”)
            print("You are passed in the exam")
else:
            print(“Sorry! You failed the exam, better luck next time”)
```

IF – ELSE STATEMENT RULES RECAP

- If statement can be followed by an **optional** else statement, which executes when the boolean expression is **false**

DECISION MAKING FLOWCHART: IF – ELSE IF


1st Condition
False
True
2nd Condition
False
True
Execute when 1st condition is true
Execute when 2nd condition is true

IF:
1st level
boolean expression
to evaluate

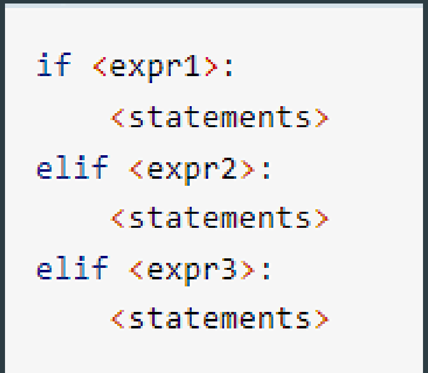
ELSE-IF:
2nd to nth level
boolean expression
to evaluate

THEN:
Actions to be executed
when **1st** condition **is** met

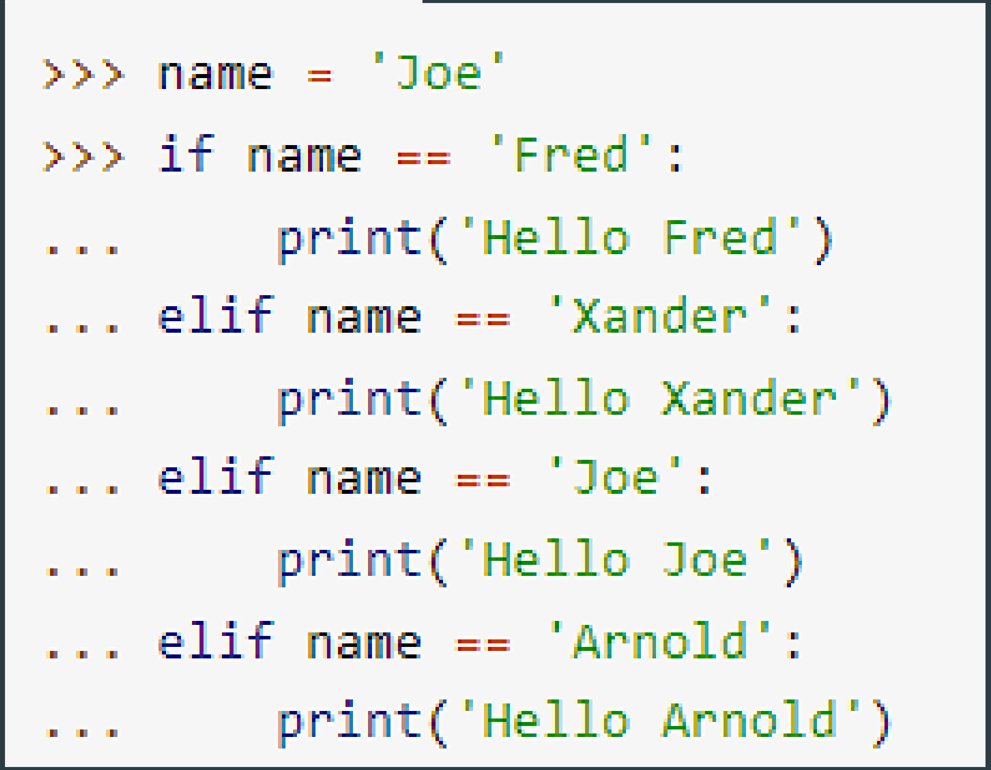
ELSE-IF-THEN:
Actions to be executed
when **2nd** condition **is** met

Do nothing when
conditions **are not** met

IF – elif statement

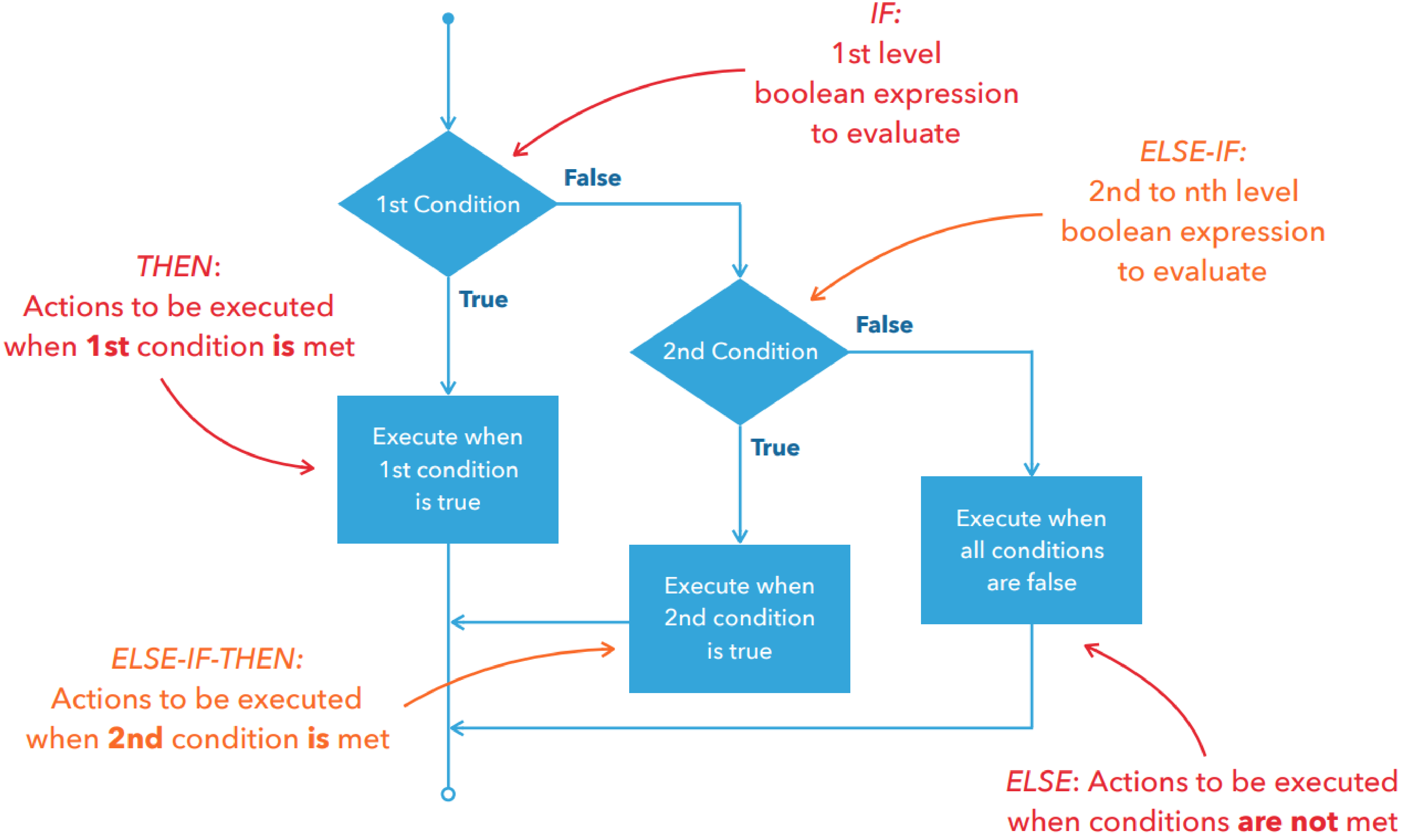


```
if <expr1>:
    <statements>
elif <expr2>:
    <statements>
elif <expr3>:
    <statements>
```



```
>>> name = 'Joe'
>>> if name == 'Fred':
...     print('Hello Fred')
... elif name == 'Xander':
...     print('Hello Xander')
... elif name == 'Joe':
...     print('Hello Joe')
... elif name == 'Arnold':
...     print('Hello Arnold')
```

DECISION MAKING FLOWCHART: IF – ELSE IF – ELSE


1st Condition
False
True
2nd Condition
False
True
Execute when 1st condition is true
Execute when 2nd condition is true
Execute when all conditions are false

IF:
1st level
boolean expression
to evaluate

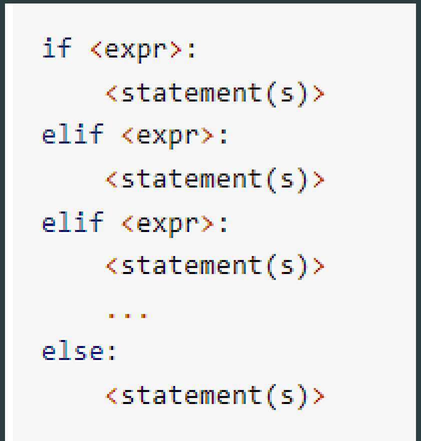
ELSE-IF:
2nd to nth level
boolean expression
to evaluate

THEN:
Actions to be executed
when **1st** condition **is** met

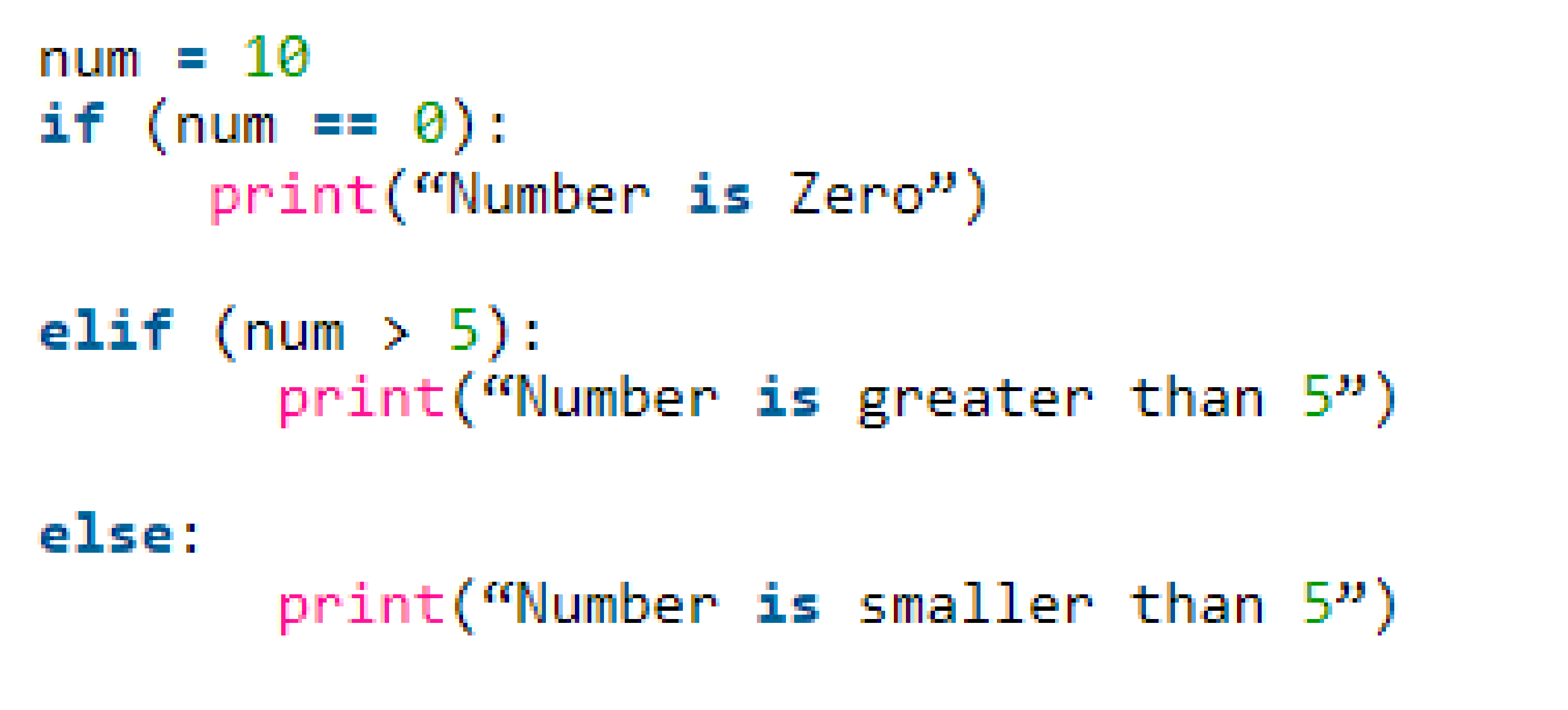
ELSE-IF-THEN:
Actions to be executed
when **2nd** condition **is** met

ELSE: Actions to be executed
when conditions **are not** met

IF – elif - else statement



```
if <expr>:
    <statement(s)>
elif <expr>:
    <statement(s)>
elif <expr>:
    <statement(s)>
    ...
else:
    <statement(s)>
```



```
num = 10
if (num == 0):
     print(“Number is Zero”)

elif (num > 5):
       print(“Number is greater than 5”)

else:
       print(“Number is smaller than 5”)
```

IF – ELSE IF – ELSE STATEMENT RULES RECAP

- An if can have **zero** or **one** else's and its must come after any else if's

- An if can have **zero** to **many** else if's and they must come **before** else

- Once an else if **succeeds**, **none** of the **remaining** else if's or else's will be tested

Agenda for Today

- **Flow control - IF**
- **Flow control - Switch**
- Practice

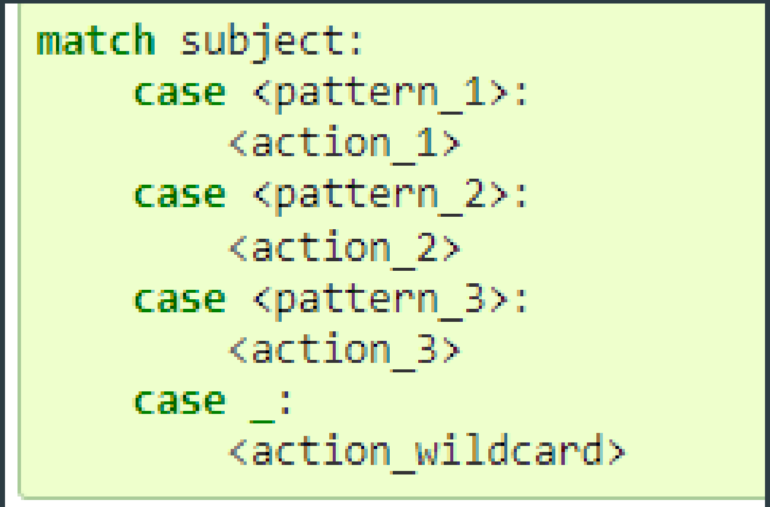
SWITCH STATEMENT OVERVIEW

- Provides an **effective** way to deal with a section of code that could branch in **multiple directions** based on **single variable**

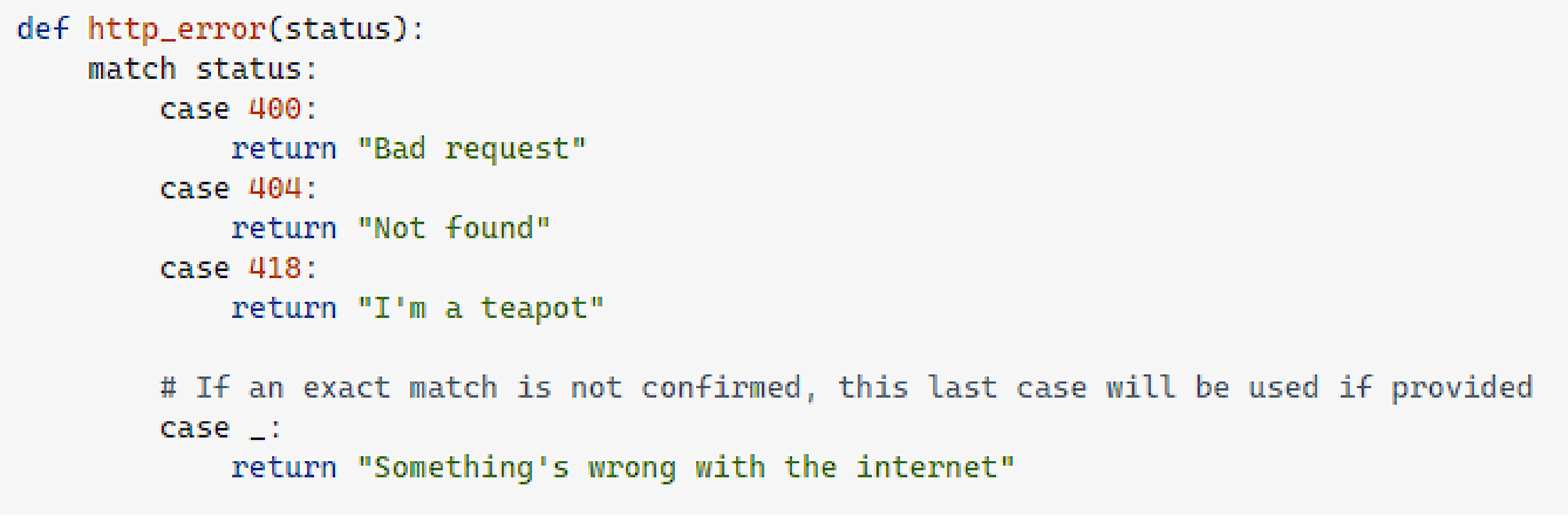
- **Doesn't** support the conditional operators that the **if statement** does

- **Can't** handle **multiple** variables

Match



```
match subject:
    case <pattern_1>:
        <action_1>
    case <pattern_2>:
        <action_2>
    case <pattern_3>:
        <action_3>
    case _:
        <action_wildcard>
```



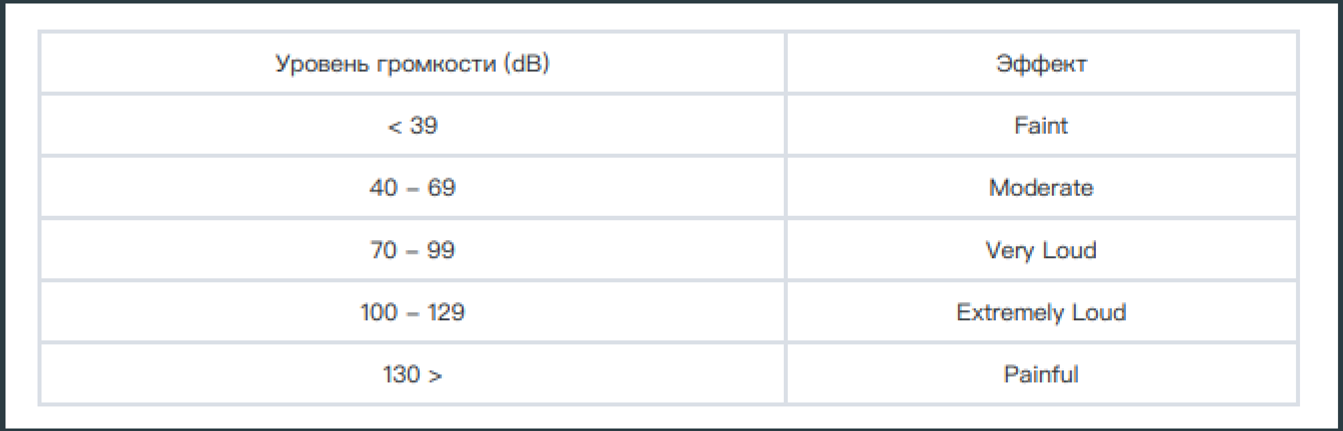
```
def http_error(status):
    match status:
        case 400:
            return "Bad request"
        case 404:
            return "Not found"
        case 418:
            return "I'm a teapot"

        # If an exact match is not confirmed, this last case will be used if provided
        case _:
            return "Something's wrong with the internet"
```

Agenda for Today

- Flow control - IF
- Flow control - Switch
- Practice

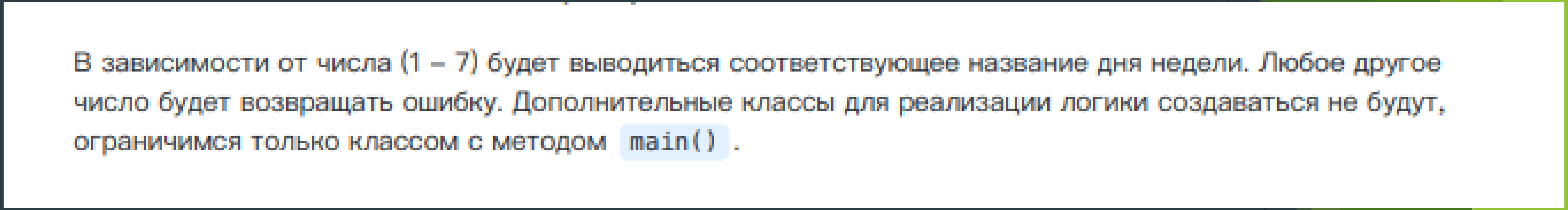
Task 1

- Create a noise detector:



Уровень громкости (dB)	Эффект
< 39	Faint
40 – 69	Moderate
70 – 99	Very Loud
100 – 129	Extremely Loud
130 >	Painful

Task 2

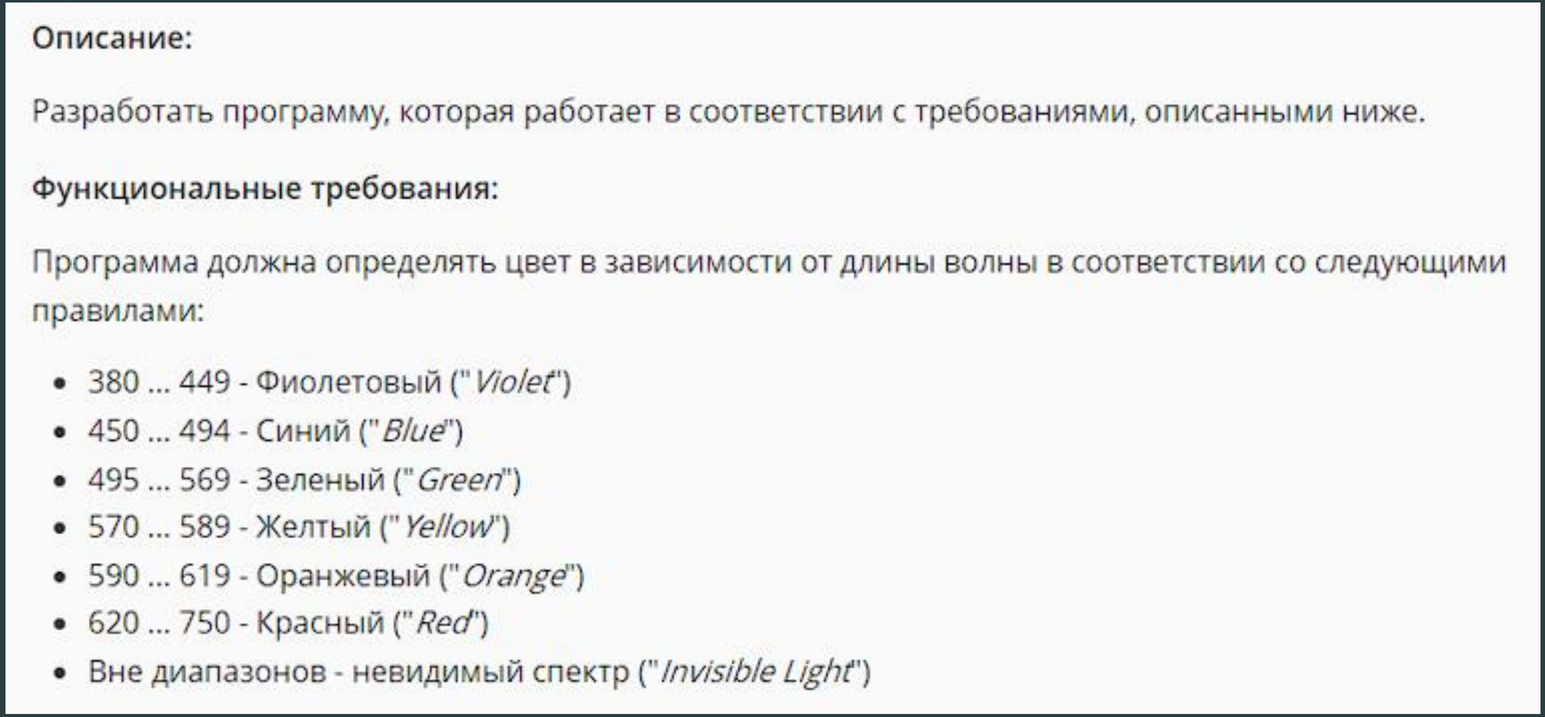


В зависимости от числа (1 – 7) будет выводиться соответствующее название дня недели. Любое другое число будет возвращать ошибку. Дополнительные классы для реализации логики создаваться не будут, ограничимся только классом с методом `main()` .

Task 3

- Modify our bank program
- Withdrawal = balance<0 – display error – you don’t have enough money
- Withdrawal = amount>700 – display error - daily limit exceeded
- Deposit = amount >= 10000 – display error – need to register sum in VID

Task 4



Описание:

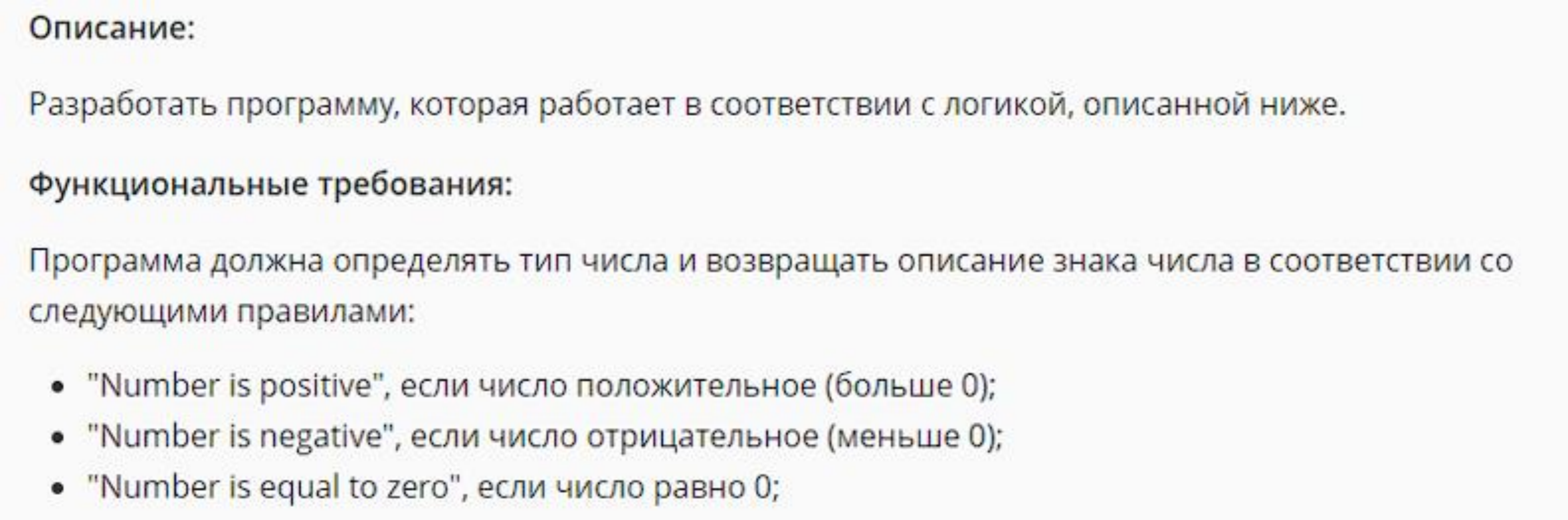
Разработать программу, которая работает в соответствии с требованиями, описанными ниже.

Функциональные требования:

Программа должна определять цвет в зависимости от длины волны в соответствии со следующими правилами:

- 380 ... 449 - Фиолетовый ("*Violet*")
- 450 ... 494 - Синий ("*Blue*")
- 495 ... 569 - Зеленый ("*Green*")
- 570 ... 589 - Желтый ("*Yellow*")
- 590 ... 619 - Оранжевый ("*Orange*")
- 620 ... 750 - Красный ("*Red*")
- Вне диапазонов - невидимый спектр ("*Invisible Light*")

Task 5



Описание:

Разработать программу, которая работает в соответствии с логикой, описанной ниже.

Функциональные требования:

Программа должна определять тип числа и возвращать описание знака числа в соответствии со следующими правилами:

- "Number is positive", если число положительное (больше 0);
- "Number is negative", если число отрицательное (меньше 0);
- "Number is equal to zero", если число равно 0;

Reference

- If - https://www.programiz.com/python-programming/if-elif-else
- Match - https://docs.python.org/3.10/whatsnew/3.10.html#pep-634-structural-pattern-matching
- Additional links - https://www.softwaretestinghelp.com/python/python-conditional-statements/

THANK YOU FOR YOUR ATTENTION!
...YOU ALWAYS HAVE MY ATTENTION.
makeameme.org