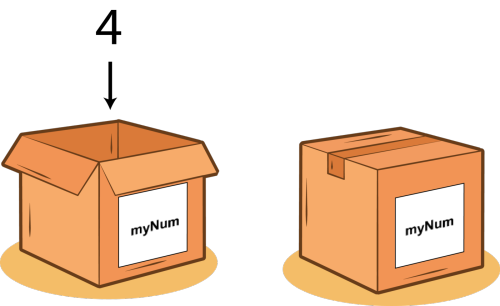
1. name = 'Bob'
2. age = 20
3. if name == 'Bob':
4. if age > 10:
5. print('OK')
6. [7:51 PM]



1. 

clojurerabbit — Today at 7:58 PM

На примере Javascript разница в традиционной русскоязычной и англоязычной терминологиях:

1. [7:59 PM]
2. // Русскоязычная терминология
3. let a = 10;
4. let b = 20;
5. let c = a + b; // В этой строке = и + называются "операция"
6. if (c > 10) { // В этой строке if называется оператор
7. ...

}

(edited)

1. [8:02 PM]
2. // Традиционная терминология
3. let a = 10;
4. let b = 20;
5. let c = a + b; // В этой строке = и + называются "оператор (operator)"
6. if (c > 10) { // В этой строке if называется инструкция (statement)
7. ...

}

1. [8:05 PM]

Чем отличаются эти примеры:

if (b > 10)

console.log('OK')

b = b + 10

1. [8:05 PM]
2. if (b > 10) {
3. console.log('OK')
4. b = b + 10

}

1. [8:07 PM]

Можно сказать, что для интерпретатора Javascript statement if выглядит так: if (<УСЛОВИЕ>) <ЧТО ДЕЛАТЬ> (edited)

1. [8:10 PM]
2. if (b > 10)

console.log('OK')

1. [8:12 PM]

В большинстве языков (c, с++, java, c#, ...) для того, чтобы ограничить блок, используются фигурные скобки:

if (a > 10) {

...

}

1. [8:13 PM]

В некоторых других (например pascal) - блоки begin end:

if a > 10 begin

...

end

1. [8:13 PM]

Python решил пойти по другому пути...

1. [8:15 PM]

В обычных языках:

if (b > 10) {

console.log('OK')

b = b + 10

}

// и вот это

if (b > 10) {

console.log('OK')

b = b + 10

}

// одно и то же, но второй пример неудобно читать

1. [8:17 PM]
2. if b > 10: # Перед блоком ставится двоеточие
3. ... # А в блок входят все строки, с одинаковым отступом
4. .,, # (тоже входит)

a = 10. # уже не входит - отступ поменялся

(edited)

1. [8:20 PM]

Поэтому инструкции у нас выглядят таким образом:

if a > 10:

print(a)

`

1. [8:22 PM]



1. [8:28 PM]

<https://www.pdfdrive.com/the-self-taught-programmer-the-definitive-guide-to-programming-professionally-e158527573.html>

[**The Self-Taught Programmer: The Definitive Guide to Programming Pro...**](https://www.pdfdrive.com/the-self-taught-programmer-the-definitive-guide-to-programming-professionally-e158527573.html)

I am a self-taught programmer. After a year of self-study, I learned to program well enough to land a job as a software engineer II at eBay. Once I got there, I realized I was severely under-prepared. I was overwhelmed by the amount of things I needed to know but hadn't learned yet. My journey learn



**NEW**

1. [8:33 PM]

if также называют условной инструкцией (conditional statement). Она нужна для того, чтобы выполнять какие-то действия не всегда, а только если выполнилось какое-то условие (отсюда и "условная"). Полностью она выглядит так:

name = "Bob"

if name == "Bob": # if <условие> - это основное условие

print("Hello Bob")

elif name == "John": # elif <условие> - если хотим проверить еще что-то, раз уже не выполнилось предыдущее

print("Hello John")

elif name == "Jane": # elif может быть много

print("Hello Jane")

else: # else - просто если ни одно предыдущее не выполнилось

print("Hello there")

(edited)

1. [8:33 PM]

while и for - инструкции-циклы.

1. 

clojurerabbit — Today at 8:43 PM

# Работает очень просто:

# 1. Проверяется условие, если оно верное, переходим к шагу 2, если нет, завершаем цикл

# 2. Выполняется блок действий

# 3. Переходим к шагу 1

while <условие>:

<блок действий>

(edited)

1. [8:45 PM]

Кстати # - символ комментария в Python, аналог // из Javascript

1. [8:45 PM]

Например

1. [8:45 PM]
2. while 2 < 10:

print("2 is still < 10")

1. [8:47 PM]

Когда пишешь цикл, то если только ты не уверен на 100 процентов, что он должен быть бесконечным, первым делом надо продумать, как он будет завершаться. Что изменится в условии?

1. [8:48 PM]
2. n = 1
3. while n <= 10:

print(n)

1. [8:49 PM]
2. n = 1
3. while n <= 10:
4. print(n)

n = n + 1