

# Условный оператор if

Итак, условная инструкция в Python имеет следующий синтаксис:

**if** Условие:

    Блок инструкций 1

**else:**

    Блок инструкций 2

Блок инструкций 1 **будет выполнен, если Условие истинно. Если Условие ложно, будет выполнен Блок инструкций 2.**

Для выделения блока инструкций, относящихся к инструкции **if** или **else**, в языке Python используются отступы.

Все инструкции, которые относятся к одному блоку, должны иметь равную величину отступа, то есть одинаковое число пробелов в начале строки. Рекомендуется использовать отступ в **4 пробела** и **не рекомендуется** использовать в качестве отступа символ табуляции.

*Это одно из существенных отличий синтаксиса Python от синтаксиса большинства языков, в которых блоки выделяются специальными словами или фигурными скобками.*

## Неполное ветвление

В условной инструкции может отсутствовать слово **else** и последующий блок. Такая инструкция называется неполным ветвлением. Например, если дано число  $x$ , и мы хотим заменить его на абсолютную величину  $x$ , то это можно сделать следующим образом:

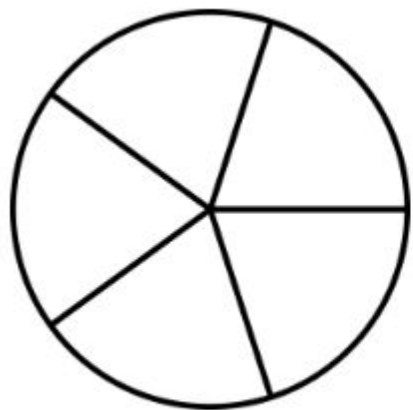
```
if x < 0:  
    x = -x  
print(x)
```

В этом примере переменной  $x$  будет присвоено значение  $-x$ , но только в том случае, когда  $x < 0$ . А вот инструкция **print**( $x$ ) будет выполнена всегда, независимо от проверяемого условия.

## Разрезание торта

Сколько разрезов нужно сделать, чтобы разрезать круглый торт на  $N$  одинаковых по форме и размеру кусков?

Например, для  $N = 6$  необходимо сделать 3 разреза, а для  $N = 5$ —5 разрезов.



$n = 5$



$n = 6$

### Входные данные

На вход подаётся натуральное число  $N$  ( $1 < N \leq 1000$ ).

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод

Вывод

6

3

5

5

## Тип данных bool, логические операции.

### Операции сравнения

Как правило, в качестве проверяемого условия используется результат вычисления одной из следующих операций сравнения (или величин логического типа):

< — меньше. Условие верно, если первый операнд меньше второго.

> — больше. Условие верно, если первый операнд больше второго.

<= — меньше или равно. Условие верно, если первый операнд не больше второго.

>= — больше или равно. Условие верно, если первый операнд не меньше второго.

== — равенство. Условие верно, если два операнда равны. Обратите внимание, что просто знак равенства для сравнения двух значений использовать нельзя, так как он означает присваивание.

!= — неравенство. Условие верно, если операнды не равны.

## Тип данных `bool`

Операции сравнения возвращают значения специального логического типа `bool`. Выражения логического типа могут принимать одно из двух значений: `True` (истина) или `False` (ложь).

Если преобразовать логическое `True` к типу `int`, то получится 1, а преобразование `False` даст 0. При обратном преобразовании число 0 преобразуется в `False`, а любое ненулевое число — в `True`.

При преобразовании `str` в `bool` пустая строка преобразуется в `False`, а любая непустая строка — в `True`.

## Логические операции

Иногда нужно проверить одновременно не одно, а несколько условий. В этом помогают логические операции. В Python существуют стандартные логические операции: логическое И, логическое ИЛИ, логическое отрицание.

- **Логическое И** (конъюнкция) является бинарной операцией (то есть операцией с двумя операндами: левым и правым) и имеет вид **and**.  
Операция **and** возвращает **True** тогда и только тогда, когда оба её операнда имеют значение **True**.
- **Логическое ИЛИ** (дизъюнкция) является бинарной операцией и возвращает **True** тогда и только тогда, когда хотя бы один операнд равен **True**.  
Операция «логическое ИЛИ» имеет вид **or**.
- **Логическое НЕ** (отрицание) является унарной (то есть с одним операндом) операцией и имеет вид **not**, за которым следует единственный операнд.  
Логическое НЕ возвращает **True**, если операнд равен **False**, и наоборот.

## Примеры:

Проверим, что число  $m$  является корректным номером месяца (лежит в интервале от 1 до 12):

```
if m >= 1 and m <= 12:  
    print("Корректно")
```

Также это можно проверить несколько иначе:

```
if 1 <= m <= 12:  
    print("Корректно")
```

## Високосный год

Дано натуральное число. Требуется определить, является ли год с данным номером високосным. Если год является високосным, то выведите YES, иначе выведите NO. Напомним, что в соответствии с григорианским календарем, год является високосным, если его номер делится на **4**, но при этом не делится на **100**, или если он кратен **400**.

### Входные данные

Вводится одно натуральное число.

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод	Вывод
100	NO



## Какое число больше?

Даны два целых числа. Программа должна вывести **1**, если первое число больше второго, **2**, если второе больше первого, или число **0**, если они равны.

### Входные данные

Вводятся два целых числа, не превышающие  $2 \cdot 10^9$  по абсолютному значению.

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод	Вывод
1 2	2
5 3	1

## Шахматная доска

Поле шахматной доски определяется парой натуральных чисел, каждое из которых не превосходит 8. По введенным координатам двух полей  $(k, l)$  и  $(m, n)$  выясните, являются ли эти поля полями одного цвета?

### Входные данные

На вход программе подаются четыре целых числа  $k, l, m, n$ .

### Выходные данные

Выведите YES или NO в зависимости от ответа на вопрос задачи.

### Примеры

Ввод	Вывод
1 1 2 2	YES
1 1 2 3	NO

## Ход ладьи

Шахматная ладья ходит по горизонтали или вертикали. Даны две различные клетки шахматной доски, определите, может ли ладья попасть с первой клетки на вторую одним ходом.

### Входные данные

Программа получает на вход четыре числа от 1 до 8 каждое, задающие номер столбца и номер строки сначала для первой клетки, потом для второй клетки.

### Выходные данные

Программа должна вывести YES, если из первой клетки ходом ладьи можно попасть во вторую, или NO в противном случае.

### Примеры

Ввод	Вывод
4 4 5 5	NO

## Ход слона

Шахматный слон ходит по диагонали. Даны две различные клетки шахматной доски, определите, может ли слон попасть с первой клетки на вторую одним ходом.

### Входные данные

Программа получает на вход четыре числа от 1 до 8 каждое, задающие номер столбца и номер строки сначала для первой клетки, потом для второй клетки.

### Выходные данные

Программа должна вывести YES, если из первой клетки ходом слона можно попасть во вторую, или NO в противном случае.

### Примеры

Ввод	Вывод
4 4 5 5	YES
4 4 5 4	NO

## Ход коня

Шахматный конь ходит буквой «Г»—на две клетки по вертикали в любом направлении и на одну клетку по горизонтали, или наоборот. Даны две различные клетки шахматной доски, определите, может ли конь попасть с первой клетки на вторую одним ходом.

### Входные данные

Программа получает на вход четыре числа от 1 до 8 каждое, задающие номер столбца и номер строки сначала для первой клетки, потом для второй клетки.

### Выходные данные

Программа должна вывести YES, если из первой клетки ходом коня можно попасть во вторую, или NO в противном случае.

### Примеры

Ввод	Вывод
2 4 3 2	YES
1 1 1 4	NO

## Яша плавает в бассейне

Яша плавал в бассейне размером  $N \times M$  метров и устал. В этот момент он обнаружил, что находится на расстоянии  $x$  метров от одного из длинных бортиков (не обязательно от ближайшего) и  $y$  метров от одного из коротких бортиков. Какое минимальное расстояние должен проплыть Яша, чтобы выбраться из бассейна на бортик? Обратите внимание, что изначально неизвестно, какой из бортиков (длиной  $N$  или  $M$  метров) является коротким.

### Входные данные

Вводятся 4 натуральных числа:  $N, M, x, y$  ( $N \neq M$ ), каждое на отдельной строке. Все числа не превосходят 100.

### Выходные данные

Программа должна вывести число метров, которое нужно проплыть Яше до бортика.

### Примеры

Ввод	Вывод
23 52 8 43	8

## Узник замка Иф

За многие годы заточения узник замка Иф проделал в стене прямоугольное отверстие размером  $D \times E$ . Замок Иф сложен из кирпичей размером  $A \times B \times C$ . Определите, сможет ли узник выбрасывать кирпичи в море через это отверстие, если стороны кирпича должны быть параллельны сторонам отверстия.

### Входные данные

Программа получает на вход числа  $A, B, C, D, E$ . Все числа натуральные, не превосходящие 10000.

### Выходные данные

Программа должна вывести слово YES или NO.

### Примеры

Ввод	Вывод
1 1 1 1 1	YES
2 2 2 1 1	NO

## Вложенные условные инструкции

### Пример: Високосный год

Дано натуральное число. Требуется определить, является ли год с данным номером високосным. Если год является високосным, то выведите YES, иначе выведите NO. Напомним, что в соответствии с григорианским календарем, год является високосным, если его номер делится на 4, но при этом не делится на 100, или если он кратен 400.

Решим задачу с помощью вложенных условных инструкций:

```
year = int(input())
if year % 100 == 0:
    if year % 400 == 0:
        print("YES")
    else:
        print("NO")
else:
    if year % 4 == 0:
        print("YES")
    else:
        print("NO")
```



## Максимум трёх чисел

Даны три целых числа. Найдите наибольшее из них (программа должна вывести ровно одно целое число).

Использовать функции `max` и `min`, а также логические операции `and` и `or` нельзя.

### Входные данные

Вводится три целых числа, не превосходящие **100** по абсолютному значению.

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод	Вывод
1 2 3	3

## Сколько совпадает чисел

Даны три целых числа. Определите, сколько среди них совпадающих. Программа должна вывести одно из чисел: **3** (если все числа совпадают), **2** (если два совпадает) или **0** (если все числа различны).

### Входные данные

Вводятся три целых числа, не превосходящие **100** по абсолютному значению.

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод	Вывод
1 2 3	0

## Упорядочить 3 числа

Дано три числа. Упорядочите их в порядке неубывания. Программа должна считывать три числа  $a$ ,  $b$ ,  $c$ , затем программа должна менять их значения так, чтобы стали выполнены условия  $a \leq b \leq c$ , затем программа выводит тройку  $a$ ,  $b$ ,  $c$ . Постарайтесь обойтись небольшим числом сравнений переменных между собой. Логические операции **and** или **or** использовать нельзя.

### Входные данные

Вводятся три числа. Все числа целые, не превосходящие 100 по абсолютному значению.

### Выходные данные

Выведите ответ на задачу.

### Примечание

Дополнительные ограничения: нельзя использовать дополнительные переменные. Единственной допустимой операцией присваивания является обмен значений двух переменных вида:

$(a, b) = (b, a)$

Программа должна содержать ровно один вызов функции печати результата.

### Примеры

Ввод	Вывод
2 3 1	1 2 3
1 2 1	1 1 2

## Апельсины бочками

Бизнесмен Василий после прочтения известной книги решил открыть новый бизнес — отгружать апельсины бочками. Партнёрам важно знать, сколько именно бочек апельсинов отгружается каждый день.

Мобильный телефон Василия поддерживает только транслит, поэтому он передаёт сообщения вида «***n*** bochek». Например, «**3** bochki» или «**1** bochka».

Напишите программу, которая выбирает правильное слово (из «bochka», «bochek», «bochki») в зависимости от значения ***n***.

### Входные данные

Одно целое число ***n*** ( $0 \leq n \leq 1000$ ).

### Выходные данные

Соответствующая фраза на транслите (см. примеры).

### Примеры

Ввод	Вывод
3	3 bochki
1	1 bochka
15	15 bochek

## Линейное уравнение

Даны числа  $a$  и  $b$ . Решите в целых числах уравнение  $ax + b = 0$ . Выведите все целочисленные решения этого уравнения, если их число конечно, выведите слово NO, если решений нет, выведите слово INF, если целочисленных решений бесконечно много.

### Входные данные

Вводятся два целых числа, не превышающих 30000 по абсолютному значению.

### Выходные данные

Выведите ответ на задачу.

### Примеры

Ввод	Вывод
6 -2	NO
1 -7	7