# Урок 03.02. Логические операторы and и or

В жизни нам очень часто приходится проверять не одно условие, а несколько. Например, если магазин открыт и ливень закончился, то можно сходить в магазин. Таким образом, мы проверили два условия. Аналогично мы проверяем несколько условий при проверке выигрышного билета, нам нужно, чтобы все цифры совпали.

Рассмотрим такую ситуацию: мы хотим купить новый велосипед. Но мы не разбираемся в велосипедах и поэтому для нас условия будут, чтобы он был современным и скоростей на нем было побольше. Чтобы не искать вручную велосипеды на онлайн площадке по продаже велосипедов мы можем написать программу, в которой задать условия год выпуска не позднее 2021 года и скоростей не меньше 24. Такая программа будет выглядеть следующим образом:

year = int(input(‘Введите год выпуска велосипеда: ’))

speed = int(input(‘Введите количество скоростей: ’))

if year >= 2021:

if speed >= 24:

print(‘Это нужный велосипед’)

else:

print(‘Этот велосипед нам не нужен’)

else:

print(‘Этот велосипед нам не нужен’)

Всё хорошо пока у нас проверяется всего два условия, но если у нас будут еще условия, например, еще 10 условий. Тогда у нас будет лесенка из условных операторов. Да еще и условия «иначе» могут повторяться, что затруднит чтение программу. На этот счет в python есть специальные логические операторы объединения условий – «and» и альтернативного условия – «or». Они помогут нам сочетать различные условия. Тогда наша программу будет выглядеть следующим образом:

year = int(input(‘Введите год выпуска велосипеда: ’))

speed = int(input(‘Введите количество скоростей: ’))

if year >= 2021 and speed >= 24:

print(‘Это нужный велосипед’)

else:

print(‘Этот велосипед нам не нужен’)

Запись стала короче и проще к пониманию и прочтению.

Теперь мы можем усложнить условие: мы можем взять велосипед с 27 скоростями и не старее 2020 года. Тогда у нас возникает два альтернативных условия, такая программа будет выглядеть так

year = int(input(‘Введите год выпуска велосипеда: ’))

speed = int(input(‘Введите количество скоростей: ’))

if year >= 2021 and speed >= 24 or year >= 2020 and speed >= 27:

print(‘Это нужный велосипед’)

else:

print(‘Этот велосипед нам не нужен’)

Давайте разберемся, как это работает. Кто знает логические операторы, тот помнит, что оператор «and»/«и» имеет выше приоритет, чем оператор «or»/«или». Это можно сравнить с арифметическими операторами, где «and»/«и» – умножение, а «or»/«или» – сложение. Тогда в тот момент, когда программа подходит к строчке с условием, то она сначала проверяет условия с «and»/«и» (выполняет умножение): вычисляет результат условий с «and»/«и», а затем вычисляет условие «or»/«или»:

year >= 2021

speed >= 24

И

ИЛИ

year >= 2020

speed >= 27

И

Вот так выглядит схема проверки условий. Однако python поступает немного предусмотрительней: так как в конце проверяется одна из альтернатив «or»/«или», то если выполнено первое условие «and»/«и» (результат выполнения «true»/«правда»), то второе условие уже не проверяется. Для этого давайте напишем такую программу:

if True or 12 / 0:

print(‘Проверка условия or.’)

else:

print(‘проверка не прошла.’)

Как мы видим на экран вывелось сообщение «Проверка условия or.», при этом ошибки деления на ноль не появилось. Если стереть «True» или условия поменять местами, то сразу возникнет ошибка деления на ноль.

Теперь рассмотрим другую задачу из мира математики. Нам дан отрезок и точка. Нужно проверить лежит ли точка на отрезке. Программа:

left\_border = 10

right\_border = 50

x = int(input(‘Введите координату точки: ’))

if x >= left\_border and x <= right\_border:  
 print(‘Точка лежит на отрезке’)

else:

print(‘Точка лежит за пределами отрезка’)

Проверим работу программы. Всё хорошо.

На самом деле в жизни очень часто встречаются задачи на проверку вхождения в диапазон и python’е это учтено. Тогда программу будет выглядеть так:

left\_border = 10

right\_border = 50

x = int(input(‘Введите координату точки: ’))

if left\_border <= x <= right\_border:  
 print(‘Точка лежит на отрезке’)

else:

print(‘Точка лежит за пределами отрезка’)

Как Вы видите – всё очень просто. Запустим программу и проверим это.

Теперь вы можете приступить к выполнению заданий.