Ещё пару слов про переменные

Имена переменных должны раскрывать своё предназначение, могут иметь любую длину, содержать латинские буквы, цифры и символ подчеркивания " _ " . *Важно!* Имена переменных не должны начинаться с цифры и не могут совпадать с одним из зарезервированных слов! Запоминать список зарезервированных слов (*keywords*) запоминать не нужно, потому что в большинстве сред разработки они выделяются другим цветом, и Вы сразу заметите ошибку. Среди программистов принято имена переменных писать строчными буквами

In [1]:
```
counter #хорошее имя, понятно, что он будет что-то считать
45word #ошибка в имени
```

```
  File "<ipython-input-1-eb08408146be>", line 2
    45word #ошибка в имени
      ^
SyntaxError: invalid syntax
```

Условия и логические выражения

Логическим (*boolean expression*) называется выражение, которое принимает одно из двух значений: истина (**True**) или ложь (**False**). Обычно оно является результатом действия (**опрератора сравнения**)

In [1]:
```
x = 9; y = 5
x == y # x равно y
x != y # x не равно y
x < y  # x меньше y
x <= y # x меньше или равно y
x > y  # x больше y
x >= y # x больше или равно y
```

Out[1]:
```
True
```

In [2]:
```
type(9 == 7)
9 == 7
```

Out[2]:
```
False
```

Существуют (**логические операторы**)

- **and** --логическое И
- **or** -- логическое ИЛИ
- **not** -- логическое отрицание НЕ

Разобраться с логическими операторами. Составить таблицу истинности

In [1]:
```
x = 9
x % 2 == 0 or x // 3 < 5
```

Out[1]:
```
True
```

Оператор выбора

Крайне часто необходимо изменять поведение программы при выполнении (или невыполнении) какого(-их)-то условия(-ий). Для этого используется оператор выбора или ветвления **if ... else** стандартный синтаксис это конструкции выглядит так:

In [ ]:
```
if условие:
    действие1
    действие2
    ...
    действиеN
else:
    действие1
    действие2
    ...
    действиеN
```

Важно! соблюдать одинаковое количество отступов на одном уровне вложенности Иногда альтернативная ветка **else** может отсутствовать или, наоборот, содержать дополнительное условие (связанное условие). Тогда вид всей конструкции будет таким

In [ ]:
```
if условие:
    действие1
    действие2
    ...
    действиеN
elif условие:
    действие1
    действие2
    ...
    действиеN
else:
    действие1
    действие2
    ...
    действиеN
```

Задача 1

Есть три числа, a, b, c. они вводятся с клавиатуры. Необходимо вывести их в порядке возрастания

In [9]:
```
a = int(input())
b = int(input())
c = int(input())
if a < b:
    if a < c:
        print(a)
        if b < c:
            print(b)
            print(c)
        else:
            print(c)
            print(b)
    else:
        print(c)
        print(a)
        print(b)
elif b < c:
    print(b)
    if a < c:
        print(a, c)
    else:
        print(c, a)
else:
    print(c, b, a)
```

```
6
5
7
5
6 7
```

задача 2

верно ли равенство a^n + b^n = c^n, для любых натуральных a, b, c и n > 2

In [10]:
```
a = int(input())
b = int(input())
c = int(input())
n = int(input())
print (a**n + b**n == c**n)
```

```
1
2
3
4
False
```

Задача 3

Проверить, можем ли мы построить треугольник из 3-х отрезков, длины которых нам известны. Неравенство треугольника: a+b < c, a+c < b, b+c < a

In [2]:
```
a = int(input())
b = int(input())
c = int(input())
(a+b > c) and (a+c > b) and (b+c > a)
```

```
3
4
5
```

Out[2]:
```
True
```

Замечание! веток **elif** может быть сколько угодно, а **else** одна, и она пишется в самом конце.

Вопрос? чем отличается предыдущая конструкция от конструкции

In [ ]:
```
if условие:
    действие1
    действие2
    ...
else:
    if условие:
        действие1
        действие2
        ...
    else:
        действие1
        действие2
        ...
        действиеN
```

Приоритет операций

Если выражение содержит больше одной операции, то необходимо знать порядок вычислений. В Python он такой же, как и в математике. Действия выполняются слева направо, соблюдая приоритетность операций

- **скобки** имеют наивысший приоритет и вычисляются первыми
- **возведение в степень**
- **умножение и деление**
- **сложение и вычитание**
- **операции сравнения**
- **логические операции**

Итерации, цикл while

In [3]:
```
while условие_выхода_из_цикла:
    действие1
    действие2
    ...
    действиеN
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-f87f1219f1d8> in <module>
----> 1 while условие_выхода_из_цикла:
      2     действие1
      3     действие2
      4     ...
      5     действиеN

NameError: name 'условие_выхода_из_цикла' is not defined
```

In [ ]:
```
while n < 10 :
    действие1
    действие2
    if условие:
        break
    ...
    действиеN
```

задача 4

Вывести количество цифр в числе

In [3]:
```
a = int(input())
count = 0
while a > 0:
    a //= 10
    count += 1
print(count)
```

```
3462
4
```