

Как переменную назовёшь, так она и...

Уважаемые коллеги, некоторых из вас я уже, наверное, замучил требованиями называть переменные какими-то, извините, “говорящими” названиями. Что это вообще значит?

Попробуем ~~использовать машинное обучение~~ понять это на примерах.

- однобуквенные переменные, сравните два примера (чем одно из них лучше, чем другое? что вам не нравится в этих решениях?):

```
1 for i in x:
2     z = 0
3     k = 0
4     for f in x:
5         if f < i:
6             z += 1
7     for f in x:
8         if f <= i:
9             k += 1
10    if k - 1 >= len(x) // 2 and z <= len(x) // 2:
11        return i
```

```
1 bigger = 0
2 smaller = 0
3 for j in range(1):
4     if arr[j] > arr[i]:
5         bigger += 1
6     if arr[j] < arr[i]:
7         smaller += 1
8 if bigger <= 1 // 2 and smaller <= 1 // 2:
9     print(arr[i])
10    break
11 elif bigger <= 1 // 2:
12     m_max = arr[i]
13 else:
14     m_min = arr[i]
```

NB: к вопросу о том, что не нравится — прочитав пример выше спустя пару недель я сам удивился, зачем я сравниваю переменную `bigger` с нулём (целочисленно деля единицу на двойку)!

- имена логических переменных (мало сделать её логической, дайте нормальное имя, чтобы проверка не выглядела издёвкой!), сравните:

```
1 count = 0
2 for i in range(0, n):
3     a = True
4     j = i + 1
5     while j < n and a:
6         if x[i] == x[j]:
7             a = False
8         else:
9             j += 1
10    if a:
11        count += 1
```

```
1 flag = 1
2 num = 1
3 for i in range(1, len(x)):
4     flag = 1
5     for idx in range(i - 1, - 1, - 1):
6         if x[idx] == x[i]:
7             flag = 0
8             break
9     if flag == 1:
10        num += 1
```

Да, переменная `flag` выглядит лучше, чем `a`, но это переименование не решает основную задачу — сделать текст программы понятнее.

Слово `flag` (варианты: `indicator`, `check`) подчёркивает то, что переменная работает своего рода “переключателем”, т.е. является логической. Но это и так видно из присваивания вида `flag = False` или `flag = True`.

При этом ни одно имя никак не поясняет, сигналом (проверкой) **ЧЕГО** является эта переменная. А ведь именно это является основной целью разумного именования функций и переменных.

А теперь вот такой вариант. Раз уж совершенство, то поменяем внутренний цикл, который теперь выглядит не так устрашающе. Нам в этом случае всё равно как искать — слева направо или справа налево.

```
1 num = 1
2 for i in range(1, len(x)):
3     seenBefore = False
4     for idx in range(i):
5         if x[idx] == x[i]:
6             seenBefore = True
7             break
8     if not seenBefore:
9         num += 1
```

Непонятно, зачем это всё нужно, пока вы пишете программы длиной 10-20 строк. Такую программу, как правило, легко целиком “удержать” в голове и помнить смысл каждой переменной.

Но как только размер программы станет хотя бы 100+ строк, помнить смысл всех переменных вроде `flag`, `flag_2`, `check_it`, `indicator` будет непросто, если вообще возможно.

Кстати, чтобы не гадать, какие имена не стоит выбирать в качестве имён переменных и функций, чтобы они не совпали с именами встроенных в Python функций, можно посмотреть вот сюда.