**Соглашения об именах**

* «Внутренний» означает внутренний по отношению к модулю, защищенный или частный внутри класса.
* Добавление одного подчеркивания ( \_) в некоторой степени поддерживает защиту переменных и функций модуля (линтеры будут помечать доступ к защищенному элементу). Обратите внимание, что модульные тесты могут получать доступ к защищенным константам из тестируемых модулей.
* Добавление двойного подчеркивания ( \_\_также известного как «dunder») к переменной экземпляра или методу фактически делает переменную или метод частной для своего класса (с использованием искажения имени); мы не одобряем его использование, поскольку оно влияет на читаемость и тестируемость и не является *конфиденциальным* . Предпочитайте одно подчеркивание.
* Поместите связанные классы и функции верхнего уровня вместе в модуль. В отличие от Java, здесь нет необходимости ограничивать себя одним классом для каждого модуля.
* Используйте CapWords для имен классов, но low\_with\_under.py для имен модулей. Хотя существуют некоторые старые модули с именем CapWords.py, сейчас это не рекомендуется, потому что это сбивает с толку, когда модуль назван в честь класса. («подожди – я написал import StringIOили from StringIO import StringIO?»)
* Новые файлы *модульных тестов* соответствуют именам методов low\_with\_under, совместимым с PEP 8, например test\_<method\_under\_test>\_<state>. Для согласованности(\*) с устаревшими модулями, которые следуют за именами функций CapWords, в именах методов, начинающихся с, могут появляться символы подчеркивания test для разделения логических компонентов имени. Один из возможных шаблонов test<MethodUnderTest>\_<state>.

|  |  |  |
| --- | --- | --- |
| **Тип** | **Общественный** | **Внутренний** |
| Пакеты | lower\_with\_under |  |
| Модули | lower\_with\_under | \_lower\_with\_under |
| Классы | CapWords | \_CapWords |
| Исключения | CapWords |  |
| Функции | lower\_with\_under() | \_lower\_with\_under() |
| Глобальные/классовые константы | CAPS\_WITH\_UNDER | \_CAPS\_WITH\_UNDER |
| Глобальные переменные/классовые переменные | CAPS\_WITH\_UNDER | \_CAPS\_WITH\_UNDER |
| Переменные экземпляра | lower\_with\_under | \_lower\_with\_under(защищено) |
| Имена методов | lower\_with\_under() | \_lower\_with\_under()(защищено) |
| Параметры функции/метода | lower\_with\_under |  |
| Локальные переменные | lower\_with\_under |  |

**Длина функции**

Отдавайте предпочтение небольшим и сфокусированным функциям.

Мы понимаем, что длинные функции иногда уместны, поэтому на длину функции не налагается жесткого ограничения. Если функция превышает примерно 40 строк, подумайте, можно ли ее разбить без ущерба для структуры программы.

Даже если ваша длинная функция сейчас работает идеально, кто-то, изменивший ее через несколько месяцев, может добавить новое поведение. Это может привести к ошибкам, которые трудно обнаружить. Если ваши функции будут короткими и простыми, другим людям будет легче читать и изменять ваш код.

При работе с некоторым кодом вы можете встретить длинные и сложные функции. Не пугайтесь изменения существующего кода: если работа с такой функцией окажется сложной, вы обнаружите, что ошибки сложно отлаживать или вы хотите использовать ее часть в нескольких разных контекстах, рассмотрите возможность разбиения функции на более мелкие. и более управляемые части.

**Математическая** **запись**

Для кода связного с математикой предпочтительны краткое наименование переменных связанных на прямую с со статьей или алгоритмом. При этом не обходима ссылаться на источник откуда взят материалом.

**Длина** **строки**

Максимальная длина строки — 80 символов .

Явные исключения из ограничения в 80 символов:

* Длинные заявления об импорте.
* URL-адреса, имена путей или длинные флаги в комментариях.
* Длинные строковые константы уровня модуля, не содержащие пробелов, которые было бы неудобно разбивать на строки, например URL-адреса или имена путей.

**Отступ**

Отступы в блоках кода составляют 4 пробела .

Никогда не используйте вкладки. Подразумеваемое продолжение строки должно выравнивать обернутые элементы по вертикали или использовать висячий отступ в 4 пробела. Закрывающие скобки (круглые, квадратные или фигурные) могут располагаться в конце выражения или на отдельных строках, но тогда должны иметь такой же отступ, как и строка с соответствующей открывающей скобкой.

Yes: *# Aligned with opening delimiter.*

foo **=** long\_function\_name(var\_one, var\_two,

var\_three, var\_four)

meal **=** (spam,

beans)

*# Aligned with opening delimiter in a dictionary.*

foo **=** {

'long\_dictionary\_key': value1 **+**

value2,

...

}

*# 4-space hanging indent; nothing on first line.*

foo **=** long\_function\_name(

var\_one, var\_two, var\_three,

var\_four)

meal **=** (

spam,

beans)

*# 4-space hanging indent; nothing on first line,*

*# closing parenthesis on a new line.*

foo **=** long\_function\_name(

var\_one, var\_two, var\_three,

var\_four

)

meal **=** (

spam,

beans,

)

*# 4-space hanging indent in a dictionary.*

foo **=** {

'long\_dictionary\_key':

long\_dictionary\_value,

...

}