

Оглавление

Об авторе	18
О техническом редакторе	18
Благодарности	18
Введение	19
Для кого написана эта книга и почему	20
О книге	20
Часть I. Первые шаги	21
Часть II. Передовые практики, инструменты и методы	21
Часть III. Объектно-ориентированный Python	22
Путешествие в мир программирования	23

ЧАСТЬ I ПЕРВЫЕ ШАГИ

Глава 1. Обработка ошибок и обращение за помощью	26
Как понять сообщения об ошибках Python	26
Анализ трассировки	27
Поиск сообщений об ошибках	30
Предотвращение ошибок при помощи статического анализатора	31
Как обратиться за помощью по программированию	33
Избегайте лишних разговоров, предоставляйте информацию заранее	34
Формулируйте свой вопрос как вопрос	34
Задавайте вопросы на подходящем веб-сайте	34
Включите краткое описание вопроса в заголовок	34
Объясните, что должен делать ваш код	35
Включите полное сообщение об ошибке	36
Приведите полный код	36
Правильно отформатируйте свой код	36

Сообщите, что вы уже пытались сделать	37
Опишите свою рабочую конфигурацию	37
Примеры вопросов	38
Итоги	39
Глава 2. Подготовка среды и командная строка	40
Файловая система	40
Пути в Python	41
Домашний каталог	42
Текущий рабочий каталог	43
Абсолютные и относительные пути	43
Программы и процессы	44
Командная строка	45
Открытие окна терминала	46
Запуск программ из командной строки	47
Аргументы командной строки	48
Выполнение кода Python из командной строки с ключом -c	50
Выполнение программ Python из командной строки	50
Запуск программы ru.exe	50
Выполнение команд из программы Python	51
Сокращение объема вводимого текста при помощи автозавершения	51
Просмотр истории команд	52
Часто используемые команды	53
Переменные среды и PATH	60
Просмотр переменных среды	61
Переменная среды PATH	62
Изменение переменной среды PATH в командной строке	63
Постоянное включение папок в PATH в Windows	63
Постоянное включение папок в PATH в macOS и Linux	65
Запуск программ Python без командной строки	65
Запуск программ Python в Windows	66
Запуск программ Python в macOS	67
Запуск программ Python в Ubuntu Linux	67
Итоги	68

ЧАСТЬ II ПЕРЕДОВЫЕ ПРАКТИКИ, ИНСТРУМЕНТЫ И МЕТОДЫ

Глава 3. Форматирование кода при помощи Black	70
Как потерять друзей и настроить против себя коллег	70
Руководства по стилю и PEP 8	71
Горизонтальные отступы	72
Использование пробелов для создания отступов	72
Отступы в середине строки	73
Вертикальные отступы	76
Пример использования вертикальных отступов	77
Рекомендации по использованию вертикальных отступов	78
Black: бескомпромиссная система форматирования кода	79
Установка Black	80
Запуск Black из командной строки	80
Отключение Black для отдельных частей кода	83
Итоги	84
Глава 4. Выбор понятных имен	85
Схемы регистра имен	86
Соглашения об именах PEP 8	86
Длина имен	87
Выбирайте имена, пригодные для поиска	90
Избегайте шуток, каламбуров и культурных отсылок	91
Не заменяйте встроенные имена	92
Худшие из возможных имен	93
Итоги	93
Глава 5. Поиск запахов в коде	95
Дублирование кода	95
«Магические» числа	97
Закомментированный и мертвый код	100
Отладочный вывод	101
Переменные с числовыми суффиксами	102
Классы, которые должны быть функциями или модулями	103

Списковые включения внутри списковых включений	104
Пустые блоки except и плохие сообщения об ошибках	106
Мифы о запахах кода	107
Миф: функции должны содержать только одну команду return в самом конце	108
Миф: функции должны содержать не более одной команды try	108
Миф: аргументы-флаги нежелательны	109
Миф: глобальные переменные нежелательны	110
Миф: комментарии излишни	111
Итоги	112
Глава 6. Написание питонического кода	113
«Дзен Python»	113
Как полюбить значимые отступы	117
Использование модуля timeit для оценки быстродействия	118
Неправильное использование синтаксиса	120
Использование enumerate() вместо range()	120
Использование команды with вместо open() и close()	121
Использование is для сравнения с None вместо ==	122
Форматирование строк	123
Использование необработанных строк, если строка содержит много символов \ (обратный слэш)	123
Форматирование с использованием F-строк	124
Поверхностное копирование списков	125
Питонические способы использования словарей	126
Использование get() и setdefault() со словарями	127
Использование collections.defaultdict для значений по умолчанию	128
Использование словарей вместо команды switch	129
Условные выражения: «некрасивый» тернарный оператор Python	130
Работа со значениями переменных	132
Сцепление операторов присваивания и сравнения	132
Проверка того, что переменная содержит одно из нескольких значений	133
Итоги	133

Глава 7. Жаргон программистов	135
Определения	135
Язык Python и интерпретатор Python	136
Сборка мусора	137
Литералы	137
Ключевые слова	138
Объекты, значения, экземпляры и идентичность	139
Элементы	143
Изменяемость и неизменяемость	143
Индексы, ключи и хеш-коды	146
Контейнеры, последовательности, отображения и разновидности множеств	149
Dunder-методы, или магические методы	150
Модули и пакеты	150
Вызываемые объекты и первоклассные объекты	151
Частые ошибки при использовании терминов	152
Команды и выражения	152
Блок, секция и тело	153
Переменные и атрибуты	154
Функции и методы	155
Итерируемые объекты и итераторы	155
Синтаксические ошибки, ошибки времени выполнения и семантические ошибки	157
Параметры и аргументы	159
Явные и неявные преобразования типов	159
Свойства и атрибуты	159
Байт-код и машинный код	160
Сценарии и программы, языки сценариев и языки программирования	161
Библиотеки, фреймворки, SDK, ядра и API	161
Итоги	162
Дополнительные ресурсы	163

Глава 8. Часто встречающиеся ловушки Python	164
Не добавляйте и не удаляйте элементы из списка в процессе перебора ...	164
Не копируйте изменяемые значения без <code>copy.copy()</code> и <code>copy.deepcopy()</code> ...	171
Не используйте изменяемые значения для аргументов по умолчанию	174
Не создавайте строки посредством конкатенации	176
Не рассчитывайте, что <code>sort()</code> выполнит сортировку по алфавиту.....	178
Не рассчитывайте на идеальную точность чисел с плавающей точкой	179
Не объединяйте операторы <code>!=</code> в цепочку	182
Не забудьте запятую в кортежах из одного элемента	183
Итоги	183
Глава 9. Экзотические странности Python	185
Почему 256 — это 256, а 257 — не 257	185
Интернирование строк	187
Фиктивные операторы инкремента и декремента в языке Python	188
Все или ничего	189
Логические значения как целые числа	190
Сцепление разных видов операторов	192
Антигравитация в Python	193
Итоги	193
Глава 10. Написание эффективных функций	194
Имена функций	194
Плюсы и минусы размера функций	195
Параметры и аргументы функций.....	198
Аргументы по умолчанию	198
Использование <code>*</code> и <code>**</code> для передачи аргументов функции	199
Использование <code>*</code> при создании вариadicеских функций	201
Использование <code>**</code> при создании вариadicеских функций.....	203
Использование <code>*</code> и <code>**</code> для создания функций-оберток.....	205
Функциональное программирование	206
Побочные эффекты	206
Функции высшего порядка	209
Лямбда-функции	209

Отображение и фильтрация со списковыми включениями	210
Возвращаемые значения всегда должны иметь один тип данных	212
Выдача исключений и возвращение кодов ошибок	214
Итоги	215
Глава 11. Комментарии, дос-строки и аннотации типов	216
Комментарии	217
Стиль комментариев	218
Встроенные комментарии	218
Пояснительные комментарии	219
Сводные комментарии	220
Комментарии «полученный опыт»	220
Комментарии об авторских правах и интеллектуальной собственности	221
Профессиональные комментарии	221
Кодовые метки и комментарии TODO	222
Магические комментарии и кодировка исходных файлов	223
Дос-строки	223
Аннотации типов	226
Статические анализаторы	228
Аннотации типов для набора типов	230
Аннотации типов для списков, словарей и т. д.	231
Обратное портирование аннотаций типов	232
Итоги	234
Глава 12. Git и организация программных проектов	235
Коммиты и репозитории	236
Создание новых проектов Python с использованием Cookiecutter	236
Установка Git	239
Настройка имени пользователя и адреса электронной почты	239
Установка графических средств Git	240
Работа с Git	241
Как Git отслеживает статус файлов	241
Для чего нужно индексирование?	243

Создание репозитория Git на вашем компьютере	243
Добавление файлов для отслеживания	245
Игнорирование файлов в репозитории	247
Сохранение изменений	248
Удаление файлов из репозитория	252
Переименование и перемещение файлов из репозитория	253
Просмотр журнала коммитов	255
Восстановление старых изменений	256
Отмена несохраненных локальных изменений	256
Деиндексирование проиндексированного файла	257
Отмена последних коммитов	257
Возврат к конкретному коммиту для отдельного файла	258
Перезапись истории коммитов	259
GitHub и команда git push	260
Отправка существующего репозитория на GitHub	261
Клонирование существующего репозитория GitHub	262
Итоги	262
Глава 13. Измерение быстродействия и анализ сложности алгоритмов	264
Модуль timeit	265
Профилировщик cProfile	267
Анализ алгоритмов с использованием нотации «О-большое»	269
Порядки нотации «О-большое»	270
Книжная полка как метафора порядков «О-большое»	271
«О-большое» как оценка худшего сценария	275
Определение порядка сложности нотации «О-большое» вашего кода	277
Почему низкие порядки и коэффициенты не важны	279
Примеры анализа «О-большое»	280
Порядок «О-большое» для часто используемых функций	283
Моментальный анализ сложности	285
«О-большое» не имеет значения при малых n... а значения n обычно малы	286
Итоги	286

Глава 14. Проекты для тренировки	288
Головоломка «Ханойская башня»	289
Вывод результатов	289
Исходный код	291
Написание кода	293
Игра «Четыре в ряд»	301
Вывод результатов	301
Исходный код	302
Написание кода	305
Итоги	314

ЧАСТЬ III ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ PYTHON

Глава 15. Объектно-ориентированное программирование и классы	316
Аналогия из реального мира: заполнение форм	317
Создание объектов на базе классов	319
Создание простого класса: WizCoin	320
Методы, <code>__init__()</code> и <code>self</code>	322
Атрибуты	323
Приватные атрибуты и приватные методы	324
Функция <code>type()</code> и атрибут <code>__qualname__</code>	326
Примеры программирования с применением ООП и без него: «Крестики-нолики»	327
Трудности проектирования классов для проектов реального мира	333
Итоги	334

Глава 16. Объектно-ориентированное программирование и наследование	335
Как работает наследование	335
Переопределение методов	338
Функция <code>super()</code>	340
Предпочитайте композицию наследованию	342
Обратная сторона наследования	343

Функции <code>isinstance()</code> и <code>issubclass()</code>	346
Методы классов	347
Атрибуты классов	349
Статические методы	350
Когда использовать объектно-ориентированные статические средства и средства уровня классов	350
Термины объектно-ориентированного программирования	351
Инкапсуляция	351
Полиморфизм	351
Когда наследование не используется	352
Множественное наследование	353
Порядок разрешения методов	354
Итоги	356
Глава 17. ООП в Python: свойства и dunder-методы	358
Свойства	358
Преобразование атрибута в свойство	359
Использование методов <code>setter</code> для проверки данных	362
Свойства, доступные только для чтения	364
Когда использовать свойства	365
Dunder-методы Python	366
Dunder-методы строкового представления	366
Числовые dunder-методы	369
Отраженные числовые dunder-методы	373
Dunder-методы присваивания на месте (<code>in-place</code>)	375
Dunder-методы сравнения	377
Итоги	382