

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»

Институт информационных технологий и технологического образования
Кафедра информационных технологий и электронного обучения

Основная профессиональная образовательная программа
Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль) «Технологии разработки программного обеспечения»
форма обучения – очная

Инвариантное задание 1.1
Практика

PyCharm: конспект

Подготовили студенты 2 курса
Нечаева Наталья Андреевна
Фирсов Кирилл Александрович
Яблонская Евгения Дмитриевна

Санкт-Петербург
2023

Оглавление

Общая характеристика	3
Функции	3
Создание проекта	3
Кодирование	8
Форматирование кода	9
Отладка	9
Запуск	10
Компиляция	11
Публикация в репозитории на GitHub (для этого нужна учетная запись)	
13	
Дополнительные функции	17
Навигация	17
Быстрые и безопасные рефакторинги	20
Веб-фреймворки Python	20
Поддержка научных библиотек	20
Необходимое программное и аппаратное обеспечение	20
Плагины для расширения функционала PyCharm	21
Источники:	22

Общая характеристика

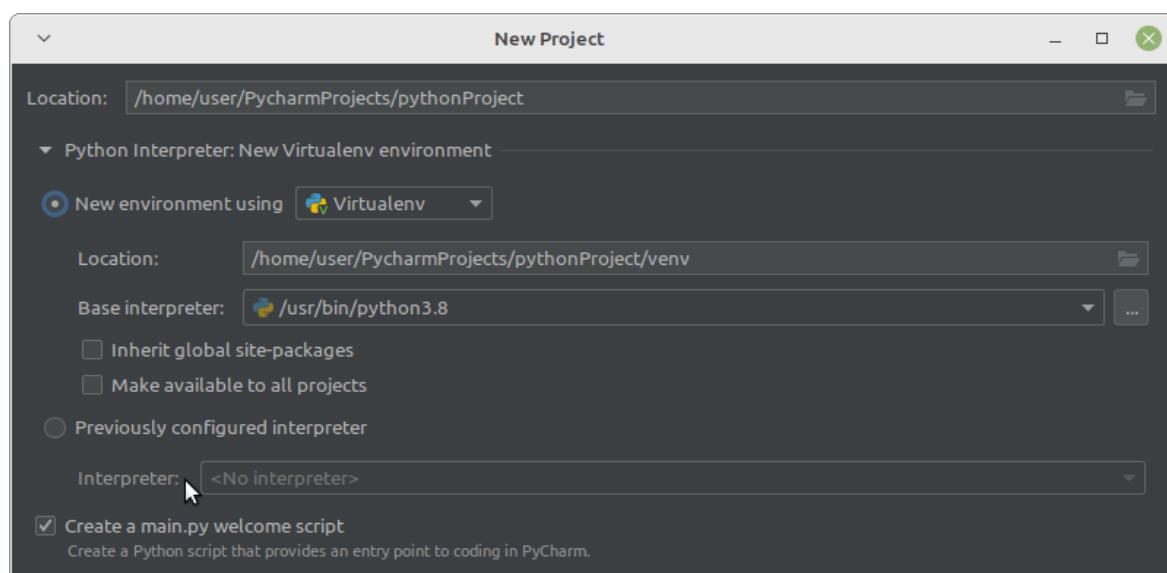
Для ускорения процесса написания программного кода удобно использовать специализированную инструментальную среду — так называемую интегрированную среду разработки (IDE, Integrated Development Environment). Эта среда включает полный комплект средств, необходимых для эффективного программирования на Python. Обычно в состав IDE входят текстовый редактор, компилятор или интерпретатор, отладчик и другое программное обеспечение. Использование IDE позволяет увеличить скорость разработки программ. В процессе разработки программных модулей удобнее работать в интерактивной среде разработки (IDE), а не в текстовом редакторе.

Для Python одним из лучших вариантов считается IDE PyCharm от компании JetBrains. Эта среда разработки доступна для Windows, Linux и macOS. Существуют два вида лицензии PyCharm: Professional (проприетарная платная версия с триальным периодом) и Community (свободно-распространяемая версия).

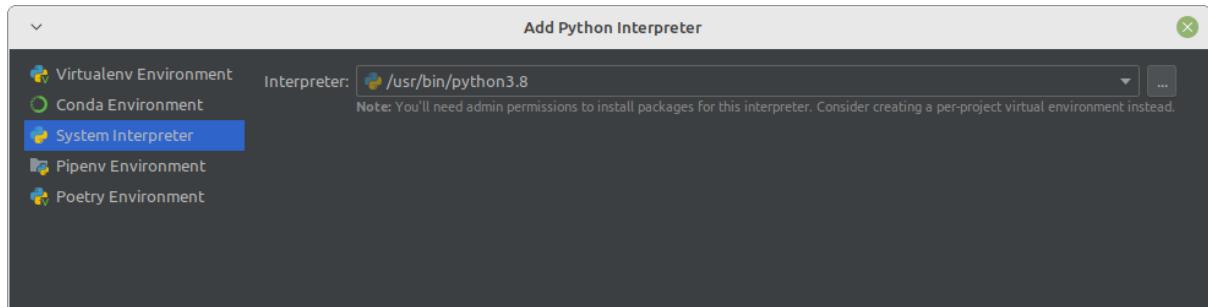
Функции

Создание проекта

При создании проекта появляется диалоговое окно, в котором следует указать путь нового каталога (или согласиться с заданным по умолчанию), создавать ли для проекта собственное виртуальное окружение:



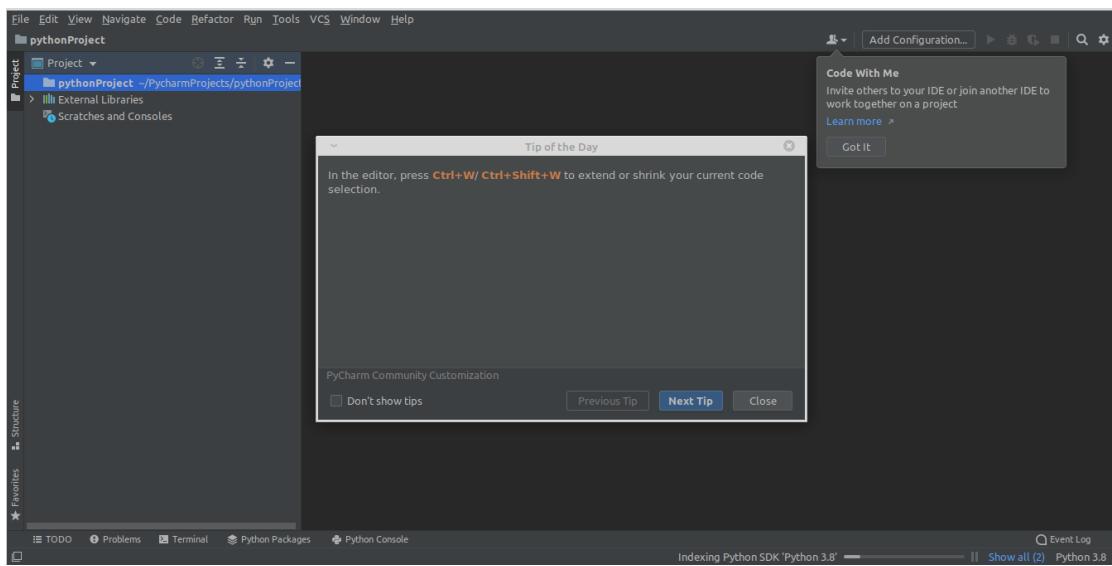
Далее нужно сделать выбор между новой виртуальной средой (New environment using) и уже существующей (Previously configured interpreter). В начале изучения языка Python может быть целесообразнее выбрать пункт Previously configured interpreter. После этого через список Interpreter: <No interpreter> выбрать системный интерпретатор (System Interpreter), указав его путь:



Но, как правило, создаётся новая виртуальная среда. В поле Location можно выбрать местонахождение папки *venv*. По умолчанию папка *venv* расположится внутри папки проекта. Если необходимо создать папку с именем отличным от *venv*, то в поле Location можно стереть *venv* и вписать название папки.

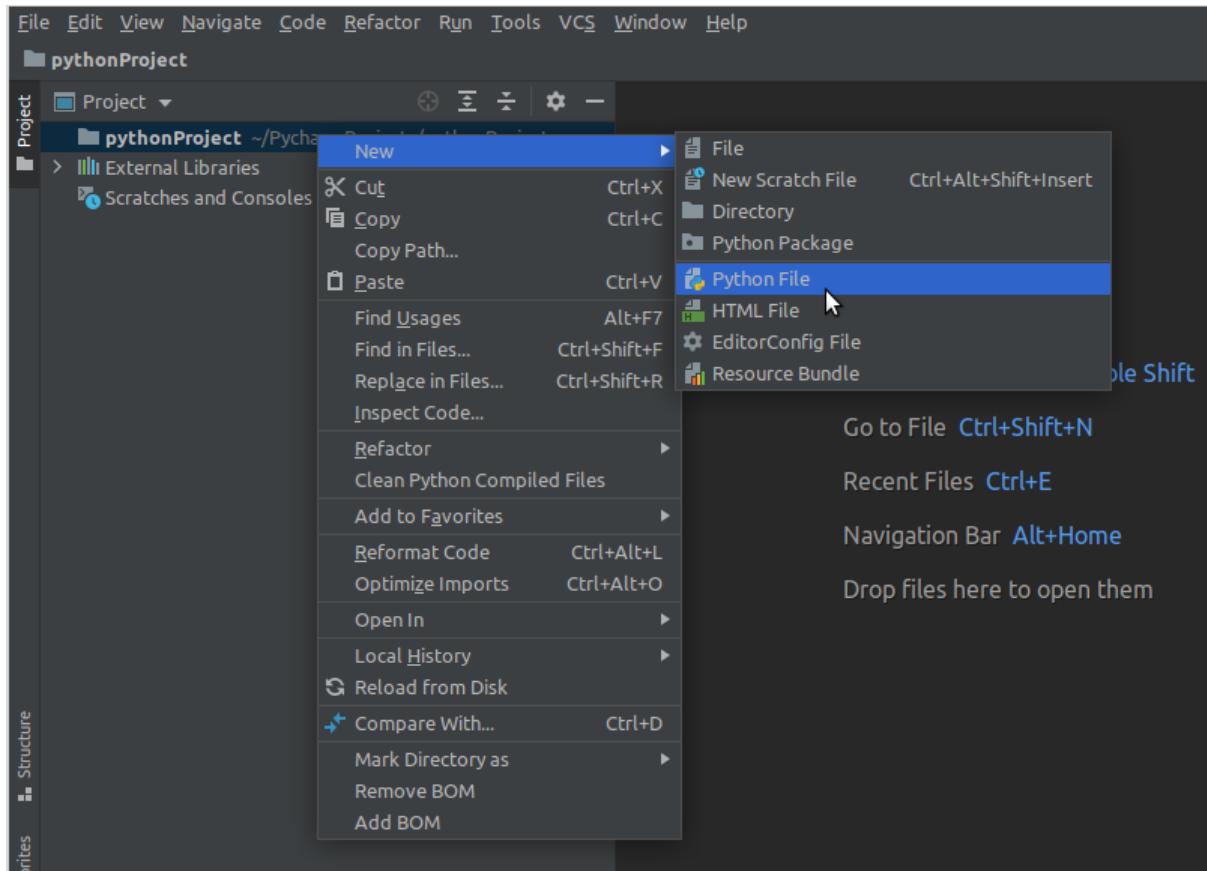
Пункт Inherit global site-packages позволяет включить в виртуальную среду все библиотеки, установленные в глобальной среде. Make available to all projects отвечает за видимость созданной виртуальной среды для других проектов. Если нужна одна виртуальная среда сразу на несколько проектов, то стоит выбрать этот пункт.

После нажатия Create запустится среда разработки, в ней будет открыт только что созданный проект:

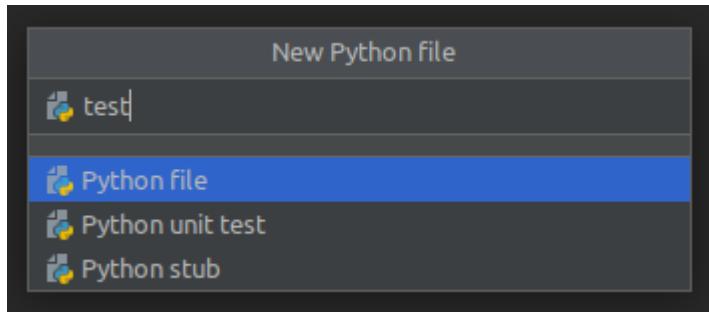


Слева на панели Project управляют файлами проекта.

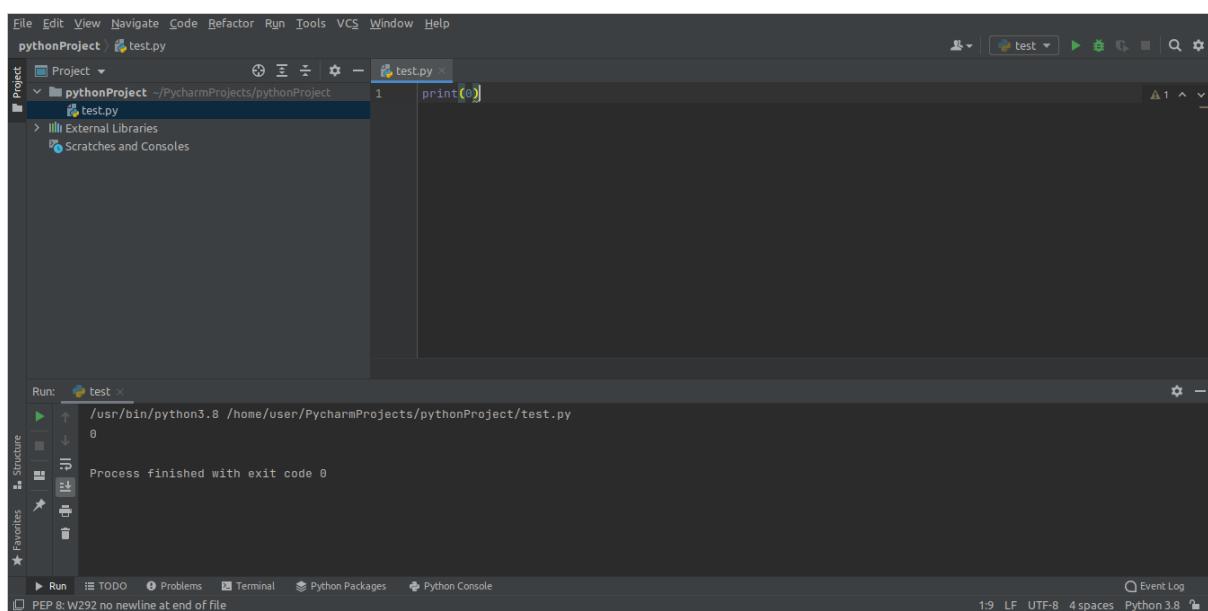
Чтобы создать файл, в котором будет написана программа на Python, кликнем по этой папке правой кнопкой мыши. В контекстном меню выбираем New → Python File:



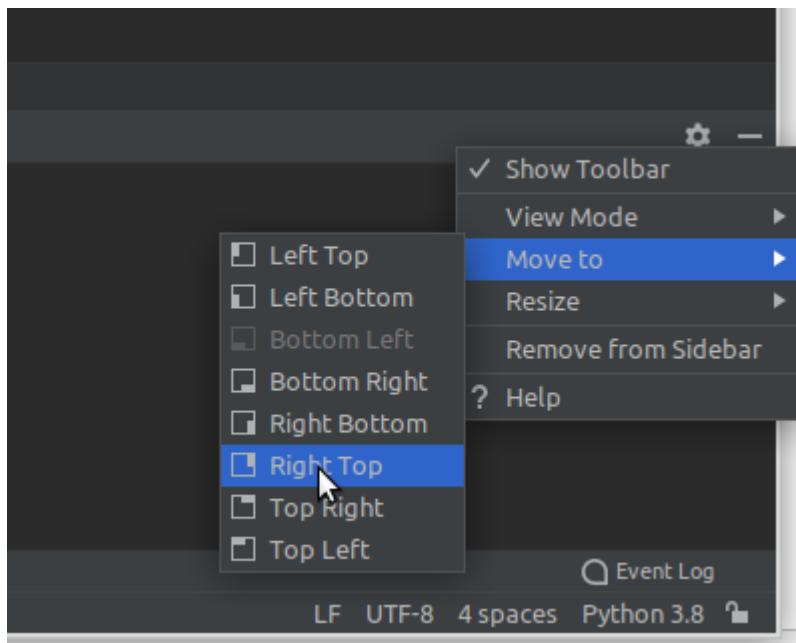
После этого в центральной части среды разработки появится небольшое окно, в которое вписываем имя файла:



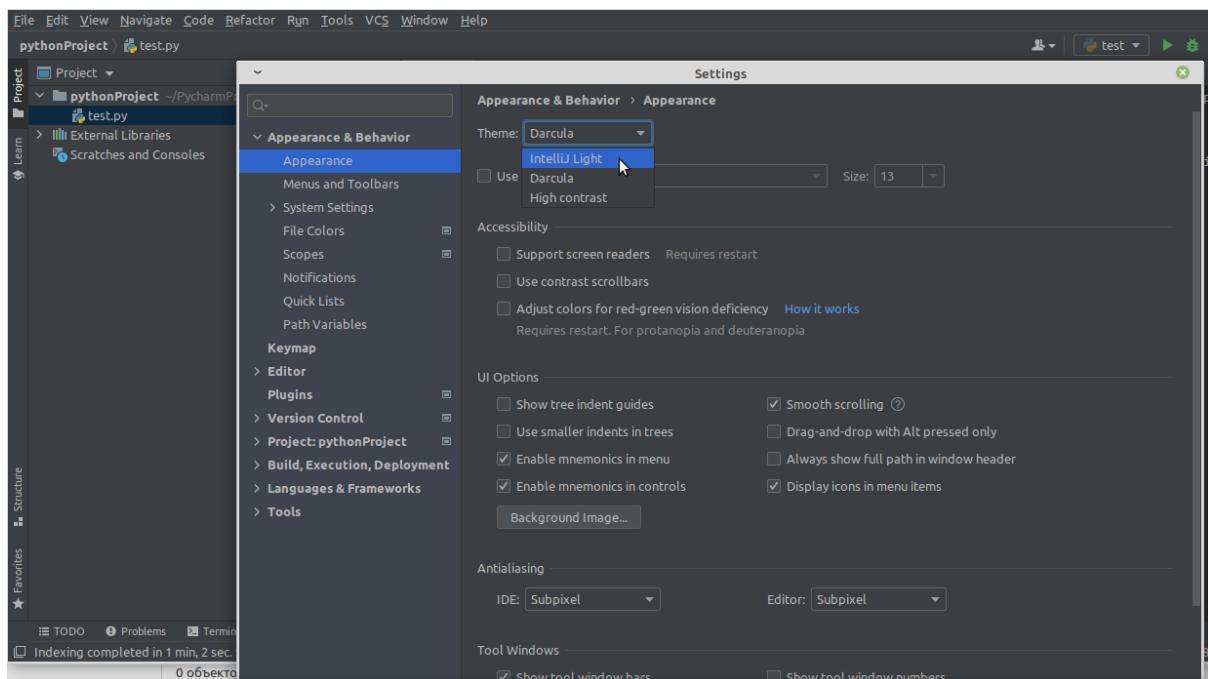
Нажав Enter, вы увидите файл на панели Project. Также он будет открыт в центральной части окна PyCharm. После того, как исходный код написан, чтобы первый раз запустить программу, проще всего нажать Ctrl+Shift+F10 (Windows) или Ctrl + Shift + R (MacOS). Внизу откроется вкладка Run, в которой отобразится результат выполнения:



Иногда удобнее, чтобы панель выполнения программы открывалась не снизу, а, например, справа. В этом случае в настройках панели (справа значок похожий на гайку) следует выбрать Move to → Right Top:



Внешний вид среды и множество других её свойств, поведение настраиваются в окне Settings (меню File → Settings):



Нажатие Ctrl + D (Windows) или Cmd + D (MacOS) дублирует строку, в которой находится курсор.

Выделенный участок можно сдвинуть вправо (сделать вложенным), нажав Tab (на Windows и MacOS). Смещение влево (на внешний уровень) выполняется комбинацией Shift + Tab (на Windows и MacOS).

Поднять/опустить (поменять местами с предшествующей/нижестоящей) строку или выделенный участок можно с помощью сочетаний Shift + Ctrl + стрелка вверх или стрелка вниз клавиатуры (Windows) или Cmd + Shift + стрелка вверх или стрелка вниз клавиатуры (MacOS).

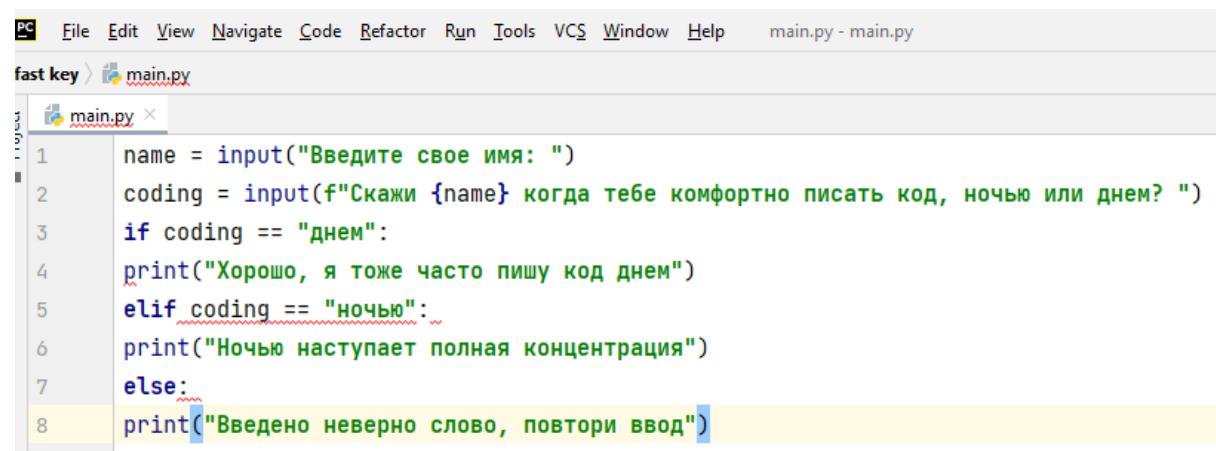
Кодирование

в PyCharm есть Intelligent Coding Assistance — интеллектуальный ассистент кодирования, который делает автодополнение кода, проверяет синтаксис, сообщает об ошибках и даёт рекомендации по их исправлению.

В частности, если написать main и нажать Tab, Space или Enter (для Windows и MacOS), PyCharm автоматически полностью завершит всю конструкцию main. Также если перед if поставить точку .if и нажать Tab, то PyCharm полностью напишет конструкцию if. То же самое верно для True.while — работает PyCharm's Postfix completions (постфиксное дополнение кода).

Форматирование кода

Команда Ctrl + Alt + L (Windows) или Cmd + Opt + L (MacOS) исправляет ошибки форматирования в коде, добавляет отступы.



The screenshot shows the PyCharm interface with the following details:

- Menu Bar:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** fast key, main.py
- Code Editor:** A Python file named "main.py" is open. The code contains a conditional statement:

```
name = input("Введите свое имя: ")
coding = input(f"Скажи {name} когда тебе комфортно писать код, ночью или днем? ")
if coding == "днем":
    print("Хорошо, я тоже часто пишу код днем")
elif coding == "ночью":_
    print("Ночью наступает полная концентрация")
else:
    print("Введено неверно слово, повтори ввод")
```

The code uses Python 3 syntax, including f-strings and a trailing underscore in the elif line.
- Status Bar:** main.py - main.py

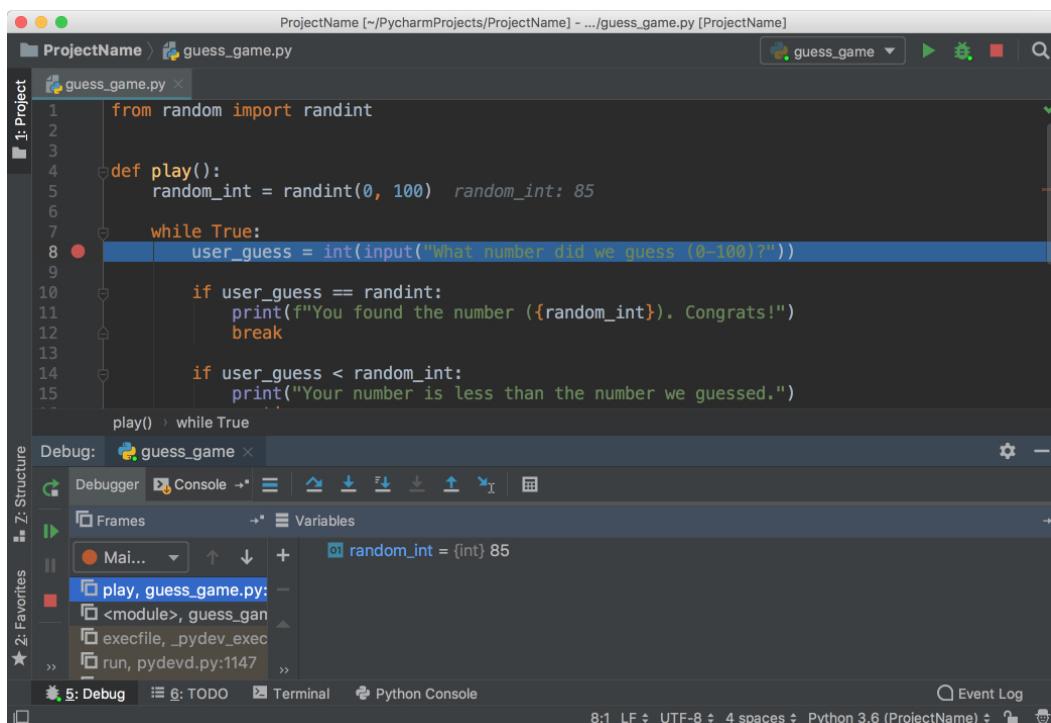
```
name = input("Введите свое имя: ")
coding = input(f"Скажи {name} когда тебе комфортно писать код, ночью или днем? ")
if coding == "днем":
    print("Хорошо, я тоже часто пишу код днем")
elif coding == "ночью":
    print("Ночью наступает полная концентрация")
else:
    print("Введено неверно слово, повтори ввод")
```

Отладка

Есть три способа начать отладку:

1. Нажать Ctrl + Shift + D на Mac или Shift + Alt + F9 в Windows или Linux.
2. Щелкнуть фон правой кнопкой мыши и выбрать Debug.
3. Щелкнуть маленьющую зелёную стрелку слева от предложения `__main__` и выбирать оттуда Debug.

После этого вы увидите открытое окно Debug внизу:



Запуск

Есть три способа запуска программы:

1. Использование клавиши Ctrl + Shift + R на Mac или Ctrl + Shift + F10 на Windows или Linux.
2. Нажать правую кнопку мыши в поле редактирования и в меню выбрать Run.
3. Если в программе есть предложение `__main__`, то щёлкнуть на маленьющую зелёную стрелку слева от фразы `__main__` и выбрать Run.

Любой из этих вариантов приведёт к запуску программы, и появится панель «Run Tool» в нижней части окна, с выводом кода.

```
from random import randint

def play():
    random_int = randint(0, 100)

    while True:
        user_guess = int(input("What number did we guess (0-100)?"))

        if user_guess == random_int:
            print(f"You found the number ({random_int}). Congrats!")
            break

        if user_guess < random_int:
            print("Your number is less than the number we guessed.")
            continue
```

Компиляция

Компиляцию можно провести с помощью модуля Pyinstaller.

В Terminal нужно перейти в папку проекта и ввести ‘`pyinstaller название_проекта.py`’.

```

## 
from tkinter import *
import time
import random
Game_Running = True
game_width = 500
game_height = 500
snake_item = 20
snake_color1 = "red"
snake_color2 = "yellow"
virtual_game_x = game_width // snake_item
virtual_game_y = game_height // snake_item
snake_x = virtual_game_x // 2
snake_y = virtual_game_y // 2
snake_x_nav = 0
snake_y_nav = 0

```

Terminal: Local +

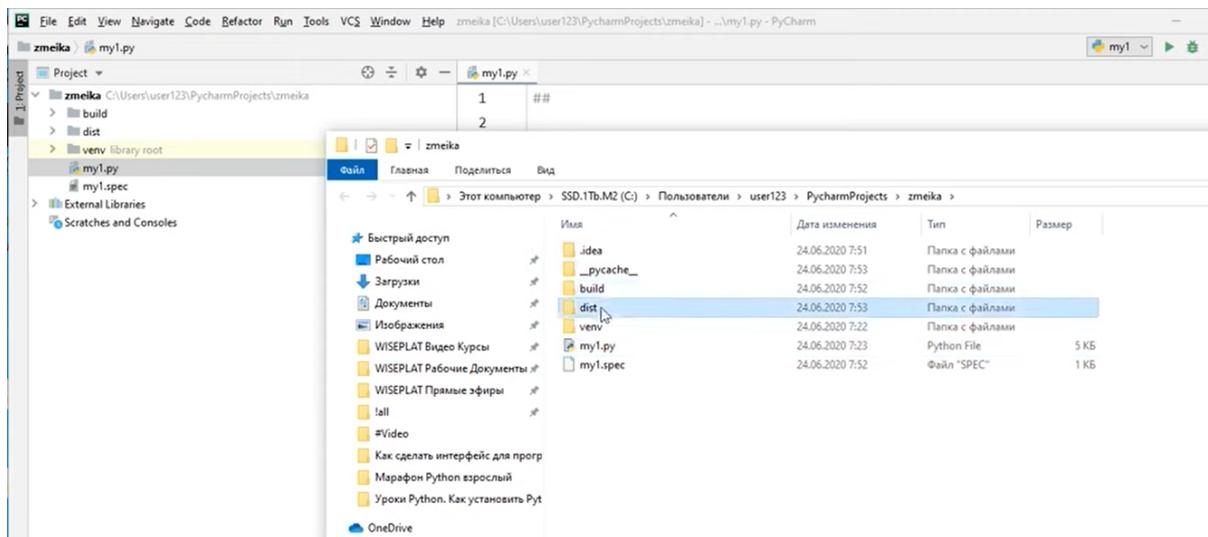
```

100% |██████████| 829kB 1.2MB/s
Building wheels for collected packages: pyinstaller
  Building wheel for pyinstaller (PEP 517) ... done
    Stored in directory: C:\Users\user123\AppData\Local\pip\Cache\wheels\62\fe\62\4c0f196d1e0dd689e097449bc81d7d585a7de7dd86b081b80b
Successfully built pyinstaller
Installing collected packages: future, pefile, pywin32-ctypes, altgraph, pyinstaller
  Running setup.py install for future ... done
  Running setup.py install for pefile ... done
Successfully installed altgraph-0.17 future-0.18.2 pefile-2019.4.18 pyinstaller-3.6 pywin32-ctypes-0.2.0
(venv) C:\Users\user123\PycharmProjects\zmeika>pyinstaller my1.py

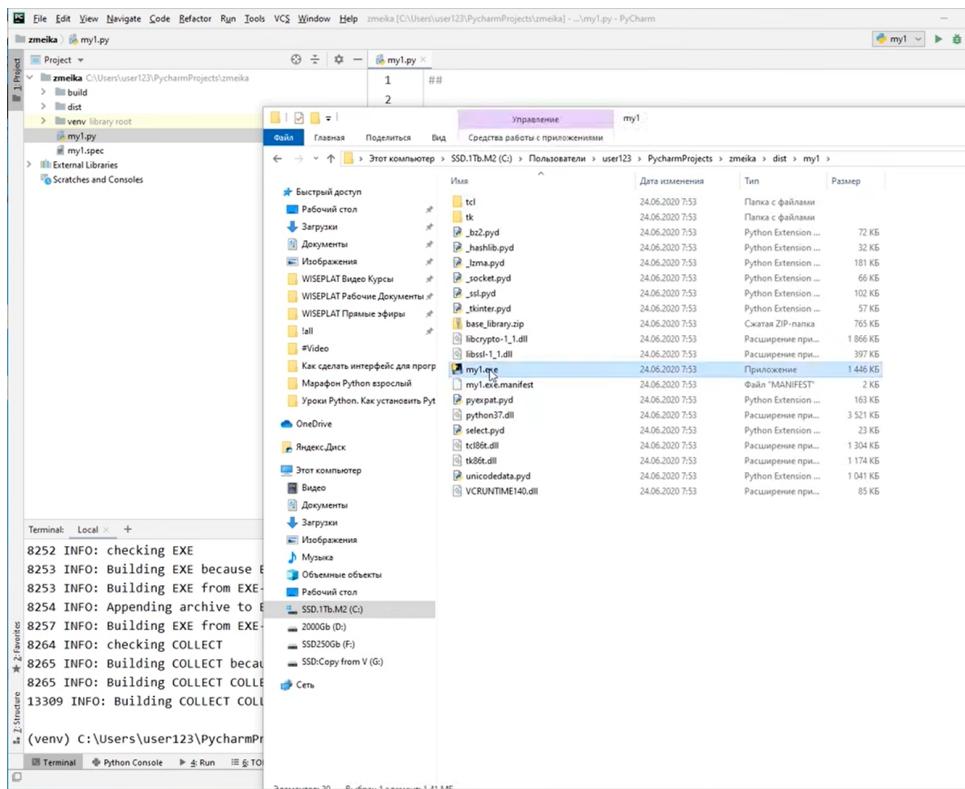
```

10:18 UTF-8 4 spaces Python 3.7 (zme)

После нажатия Enter в папке с проектом появится несколько новых папок, в том числе dist, внутри которой и будет файл с программой формата .exe.



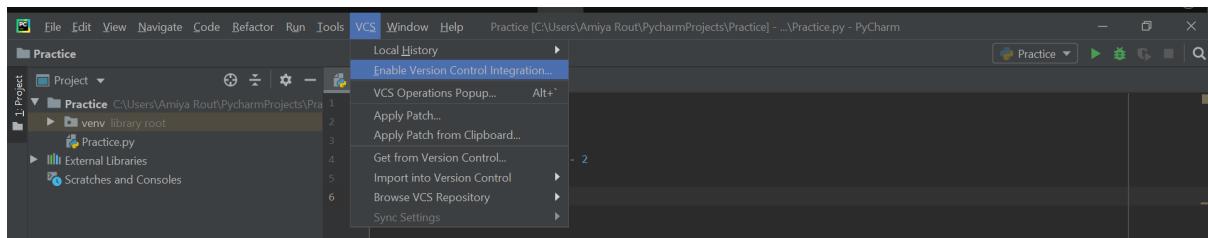
Его можно запустить:



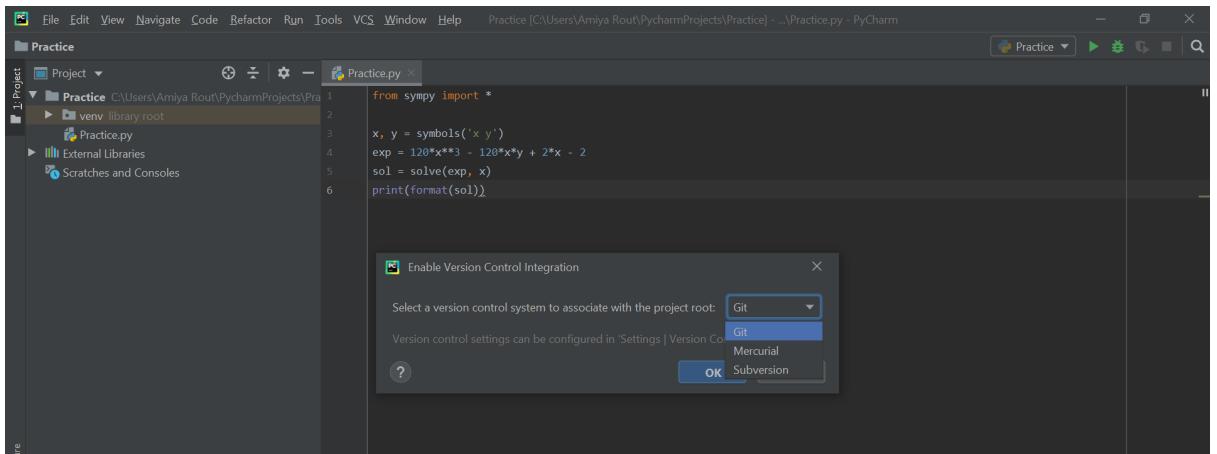
Публикация в репозитории на GitHub (для этого нужна учётная запись)

Алгоритм:

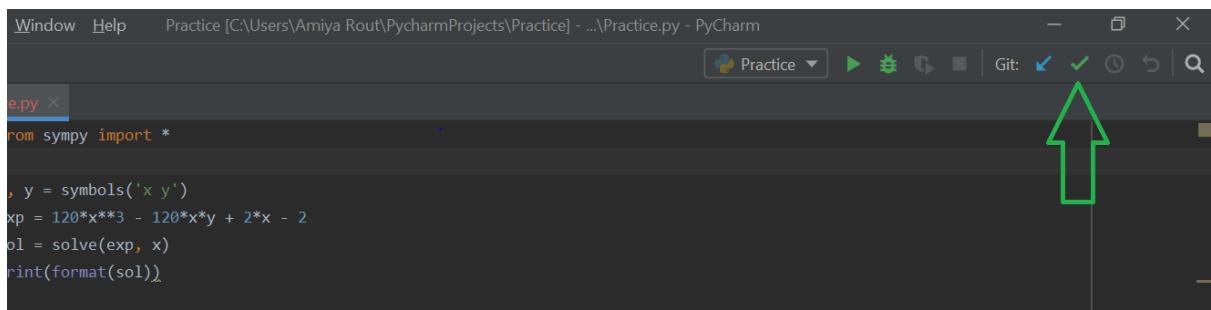
- Перейти на панель VCS, которая находится в верхней части pycharm, и нажать на неё. После нажатия выбрать «Включить интеграцию с системой управления версиями» («Enable Version Control Integration»).



После нажатия кнопки «Включить интеграцию с системой управления версиями» появится всплывающее окно. Нужно выбрать Git из выпадающего меню и нажать OK. Это инициализирует репозиторий git.

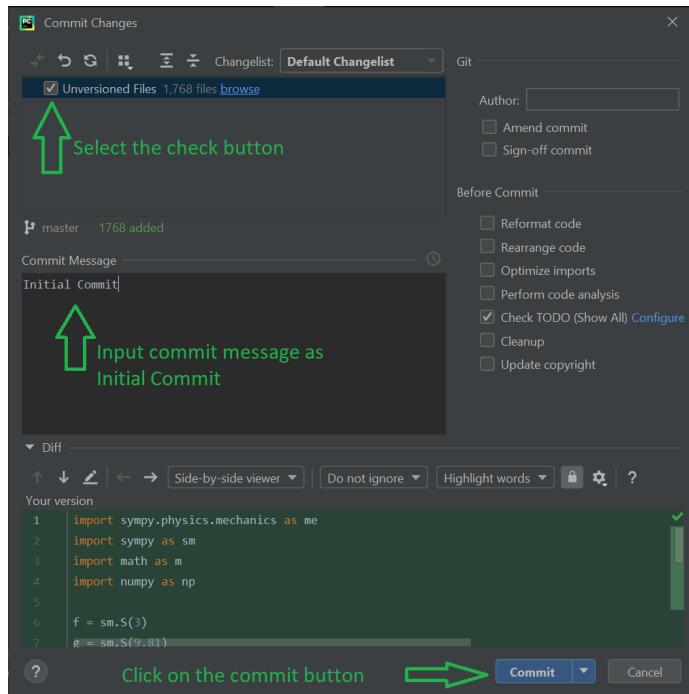


2. Следующий шаг — нажать на зелёную галочку, присутствующую в верхней части pycharm. По сути, этот символ обозначает Commit, и он собирает все «Unversioned» файлы и подготавливает их к обновлению на GitHub.

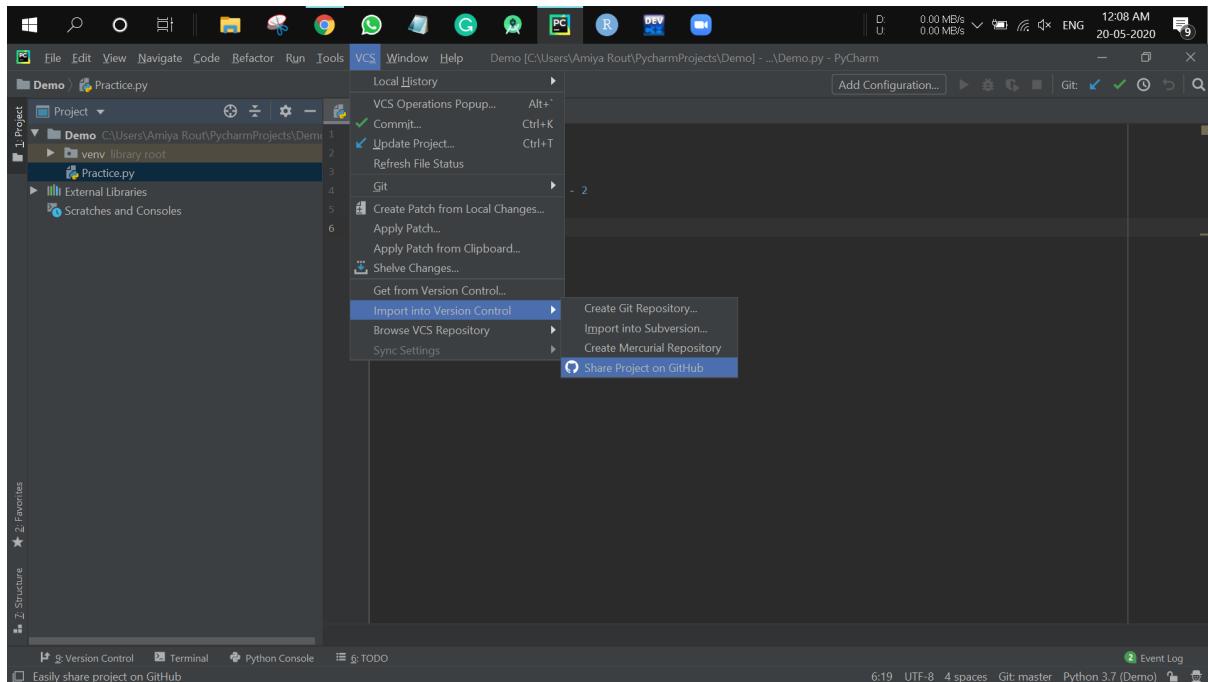


После нажатия появится новый экран. Сначала нужно выбрать Unversioned Files, затем в окно Commit Message ввести «Initial Commit», затем нажать на кнопку Commit.

Теперь проект готов к загрузке на GitHub.



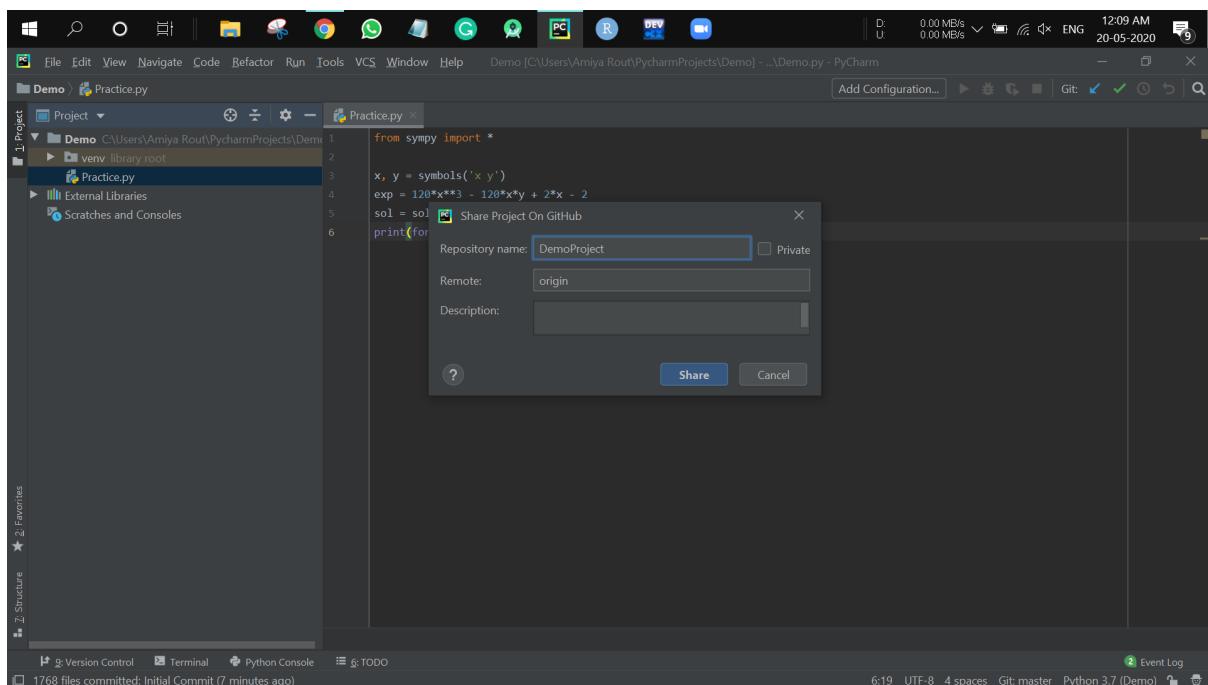
3. Последний шаг, который необходимо выполнить, - перейти в VCS, затем выбрать Import into Version Control, а затем нажать на Share Project on GitHub.



После нажатия появится всплывающее окно, в котором пользователь должен ввести свой идентификатор входа в GitHub и пароль. После успешного входа в свою учётную запись появится ещё одно

всплывающее окно, где пользователь должен указать название и описание репозитория. Затем нужно нажать на кнопку Share и готово.

Примечание: не ставьте пробелы или специальные символы при указании имени репозитория.



Дополнительные функции

Навигация

Умный поиск позволяет быстро перейти к любому классу, файлу или символу, а также к нужному окну или действию IDE.

Горячие клавиши навигации на Windows и MacOS:

Действие	MacOS	Windows
Сдвинуть курсор к предыдущему слову	Opt ←	Ctrl ←
Сдвинуть курсор к следующему слову	Opt →	Ctrl →

Сдвинуть курсор к началу строки	Cmd ←	Home
Сдвинуть курсор к концу строки	Cmd →	End
Сдвинуть курсор к соответствующей скобке	Ctrl M	Ctrl Shift M
Сдвинуть курсор к началу блока кода	Cmd Opt [Ctrl [
Сдвинуть курсор в конец блока кода	Cmd Opt]	Ctrl]
Следующий метод	Ctrl Shift ↓	Alt ↓
Предыдущий метод	Ctrl Shift ↑	Alt ↑
Сдвинуть курсор в начало страницы	Cmd Pg Up	Ctrl Pg Up
Сдвинуть курсор в конец страницы	Cmd Pg Dn	Ctrl Pg Dn
Сдвинуть курсор на страницу наверх	Pg Up	Pg Up
Сдвинуть курсор на страницу вниз	Pg Dn	Pg Dn
Сдвинуть курсор в начало текста	Cmd Home	Ctrl Home
Сдвинуть курсор в конец текста	Cmd End	Ctrl End
Перейти к строчке/столбцу...	Cmd L	Ctrl G
Переключатель	Ctrl Tab	Ctrl Tab

Выбрать в...	Opt F1	Alt F1
Недавние файл	Cmd E	Ctrl E
Место последнего изменения	Cmd Shift Backspace	Ctrl Shift Backspace
Перенести курсор назад	Cmd [Ctrl Alt ←
Перенести курсор вперед	Cmd]	Ctrl Alt →
Добавить анонимную закладку	F3	F11
Добавить закладку с нумерацией	Ctrl Shift [цифра]	Ctrl Shift [цифра]
Добавить закладку с мнемоникой	Opt F3	Ctrl F11
Показать все закладки	Cmd F3	Shift F11
Перейти к закладке с цифрой...	Ctrl [цифра]	Ctrl [цифра]
Показать мнемонические закладки	Cmd Opt F3	Ctrl Shift F11
Показать окно закладок	Cmd 2	Alt 2
Показать окно структуры	Cmd 7	Alt 7
Показать окно поиска	Cmd 3	Alt 3

Следующее появление	Cmd Opt ↓	Ctrl Alt ↓
Предыдущее появление	Cmd Opt ↑	Ctrl Alt ↑

Быстрые и безопасные рефакторинги

PyCharm предоставляет широкие возможности реорганизации кода с помощью рефакторингов Rename и Delete, Extract Method, Introduce Variable, Inline Variable, Inline Method и многих других. Рефакторинги учитывают особенности конкретного языка или фреймворка, помогая вносить изменения по всему проекту.

Веб-фреймворки Python

PyCharm обеспечивает поддержку популярных веб-фреймворков, таких как Django, Flask, Google App Engine, Pyramid и web2py. Вы можете создавать и отлаживать Django-шаблоны, работать с утилитами manage.py и appcfg.py, а также использовать специфичные для фреймворков автодополнение и навигацию.

Поддержка научных библиотек

PyCharm поддерживает Pandas, Numpy, Matplotlib и другие библиотеки для научных вычислений. IDE обеспечивает умное редактирование, позволяет просматривать наборы данных в виде графиков и в табличной форме.

Необходимое программное и аппаратное обеспечение

Официальные системные требования последней версии PyCharm выглядят так:

ОС:

Windows — Microsoft Windows 10 64-бит или Windows 8 64-бит;
macOS — macOS 10.13 или выше;
Linux — среда GNOME или KDE.

RAM: не менее 2 ГБ, но рекомендуется 8 ГБ;

Место на диске: установка потребует 2,5 ГБ, рекомендуется использование SSD;

Разрешение экрана: не менее 1024x768 пикселей;

Python: Python 2.7, Python 3.5 или более поздняя версия.

Плагины для расширения функционала PyCharm

1. Tabnine.

- автоматически дополняет код;
- предлагает закончить функцию при её объявлении;
- генерирует коды блока по запросу из комментариев.

2. PyLint

- Проверяет код на соответствие стандартам (длина строки, имена переменных);
- Подсвечивает ошибки в коде, анализирует его без запуска.

3. SonarLint

В реальном времени подсвечивает ошибки, предоставляет комментарий с причиной ошибки, риском и возможностями её устранения.

4. Rainbow Brackets

Красит скобки в различные цвета, чтобы было проще отслеживать их иерархию.

5. Python Smart Execute

Позволяет запустить отдельный блок кода в консоли.

Источники:

1. Возможности PyCharm // Jet Brains URL:
<https://www.jetbrains.com/ru-ru/pycharm/features/> (дата обращения: 11.09.2023).
2. Основы PyCharm // Хабр URL: <https://habr.com/ru/articles/720480/> (дата обращения: 10.09.2023).
3. Постолит А. В. П63 Python, Django и PyCharm для начинающих.
-СПб.: БХВ-Петербург, 2021. -464 с.: ил. -(Для начинающих) ISBN 978-5-9775-6779-4 в (дата обращения: 10.09.2023).
4. Работа в IDE PyCharm (FAQ) + полезные фишки // Pythonchik URL:
<https://pythonchik.ru/osnovy/faq-po-rabote-s-pycharm> (дата обращения: 11.09.2023).
5. PyCharm — эффективная разработка на Python // Записки Преподавателя URL:
<https://waksoft.susu.ru/2019/11/07/pycharm-effektivnaya-razrabotka-na-python/#writing-code-in-pycharm> (дата обращения: 10.09.2023).
6. PyCharm Community. Основы работы // Лаборатория Линуксоида URL: <https://younglinux.info/python/pycharm> (дата обращения: 10.09.2023).
7. Set up a Git repository // PyCharm URL:
<https://www.jetbrains.com/help/pycharm/set-up-a-git-repository.html> (дата обращения: 10.09.2023).