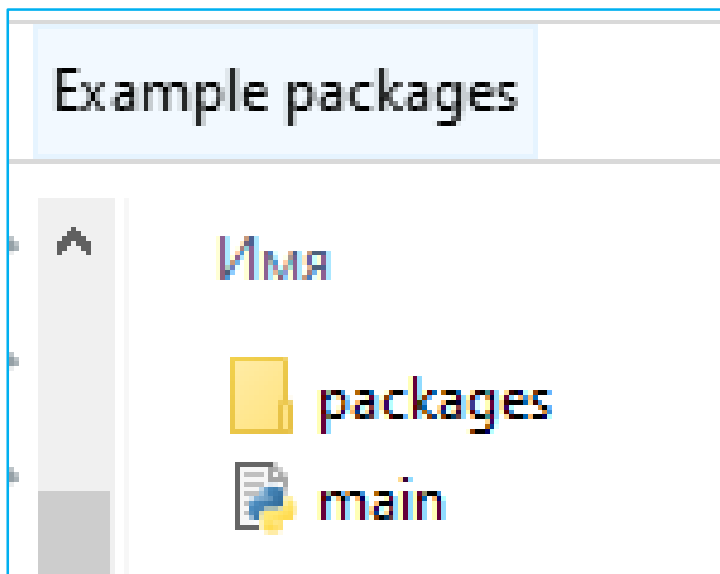


#2 Как создать пакет в Python Home work 7_2



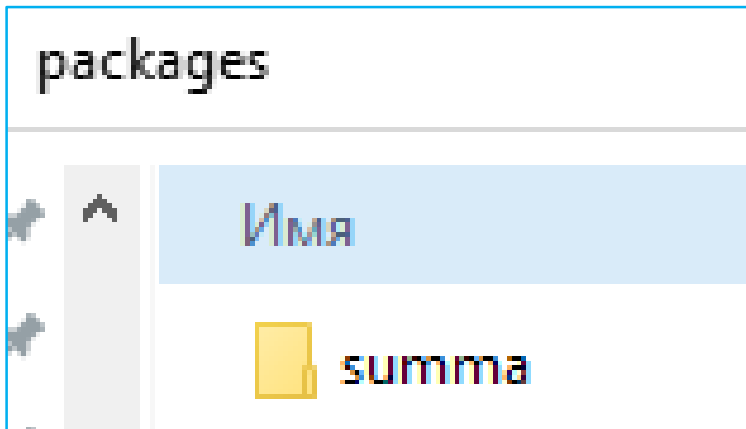
Stefan Zhauryd
Instructor



Шаг 1) Создать папку **Example packages**

Шаг 2) Внутри папки **Example packages**
создайте основной файл вашей программы **main.py**
и создайте папку **packages**

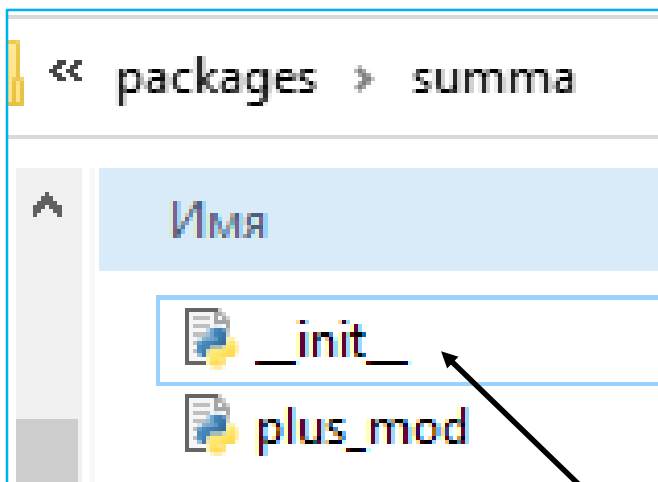




Шаг 3) Создать в папке **packages** папку для пакета с названием **summa**



Пакеты – это просто каталоги **с модулями** и специальным файлом **`__init__.py`**, который показывает Python, что этот каталог особый, так как содержит модули Python.



`__init__.py` - скажет питону
что папка является пакетом

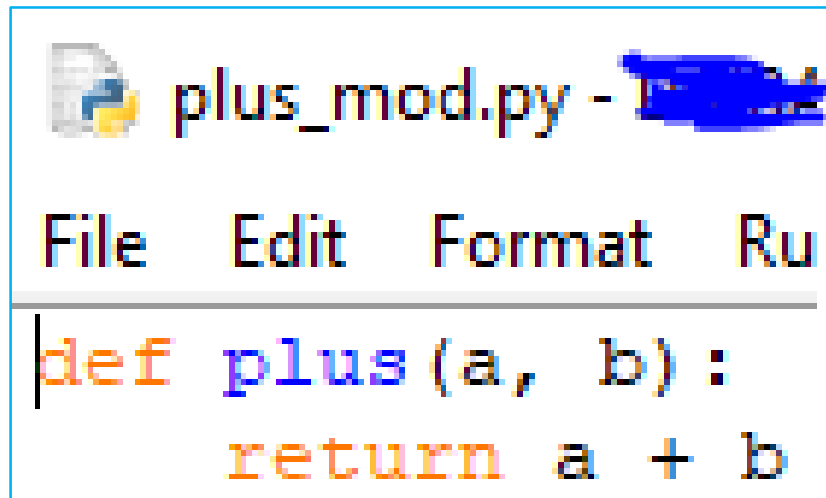
Шаг 4) Создать в папке пакета **summa** файл **`__init__.py`**

Шаг 5) Создать в папке пакета **summa** файл
`plus_mod.py`

`plus_mod.py` - это модуль с какими-то функциями, который входит в пакет **summa**



Шаг 6) Наполним модуль необходимыми функциями



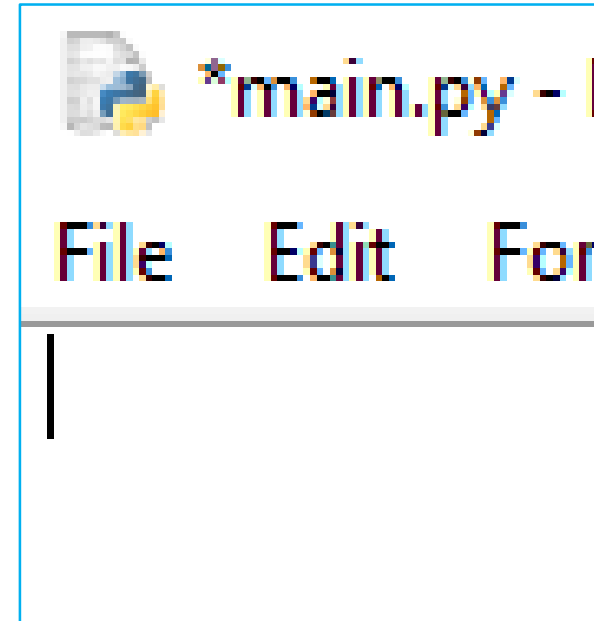
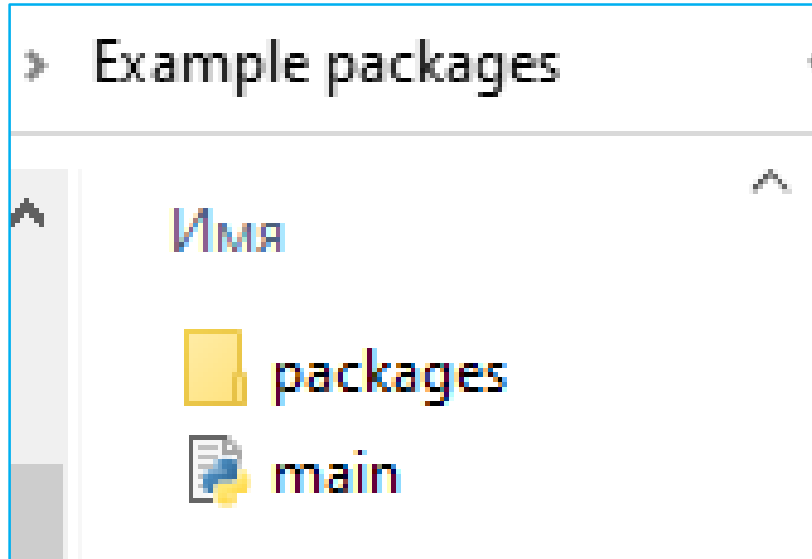
```
plus_mod.py - [redacted]  
  
File Edit Format Ru  
  
def plus(a, b):  
    return a + b
```

В файл `plus_mod.py` положим функцию 'plus'

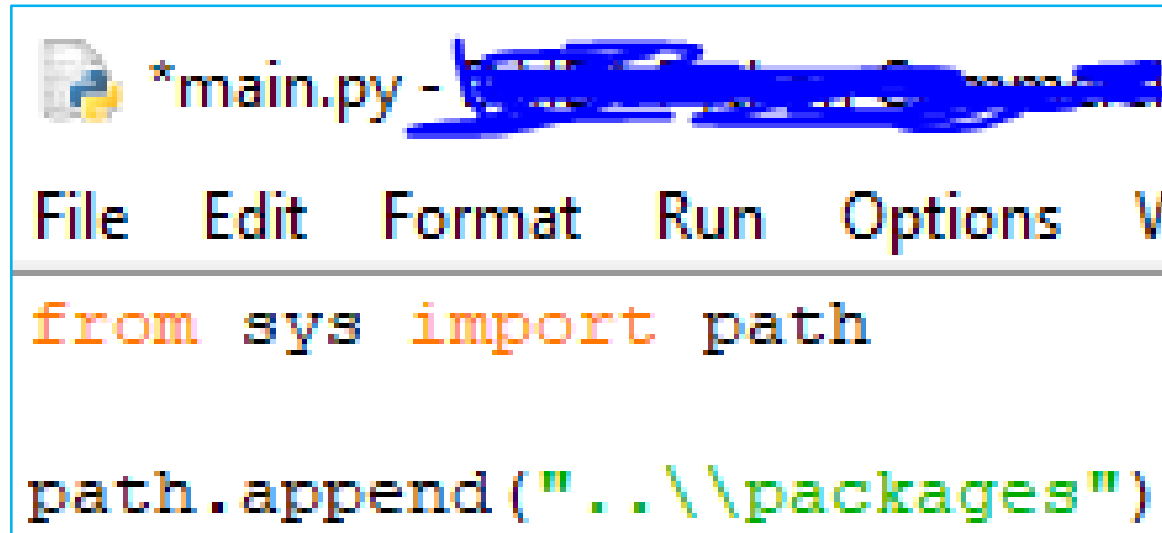
```
if __name__ == "__main__":  
    print("I prefer to be a module")  
else:  
    print("I like to be a module")
```

← At the
end of
each file

Шаг 7) Теперь откроем файл нашей программы **main.py**



Шаг 8) Добавим папку **packages** в качестве пути по умолчанию, для поиска модулей и пакетов.



```
*main.py - [redacted]
File Edit Format Run Options V
from sys import path
path.append("../packages")
```

```
if __name__ == "__main__":
    print("I prefer to be a module")
else:
    print("I like to be a module")
```

← At the
end of
each file

Шаг 9) Воспользуемся функционалом определенном в нашем пакете **summa**

```
*main.py - D:\IBA Python Commercial 07_02-14_03 2022\I
File Edit Format Run Options Window Help
from sys import path

path.append("../\packages")

from packages.summa.plus_mod import plus

a = 100
b = 80

print(plus(a, b))
```

либо так

```
File Edit Format Run Options Window Help
from sys import path

path.append("../\packages")

from packages.summa import plus_mod

a = 100
b = 80

print(plus_mod.plus(a, b))
```

from (И)з:

- 1 - папки **packages** в которой лежит
- 2 - **пакет summa** в котором находится
- 3 - **модуль plus_mod** импортировать
- 4 - **функцию 'plus'**

```
if __name__ == "__main__":
    print("I prefer to be a module")
else:
    print("I like to be a module")
```

← At the end of each file

Шаг 10) Запустив программу, В результате увидим что всё сработало:

```
main.py
File Edit Format Run Options
from sys import path
path.append("../package")
from packages.summa.plus import plus
a = 100
b = 80
print(plus(a, b))
```

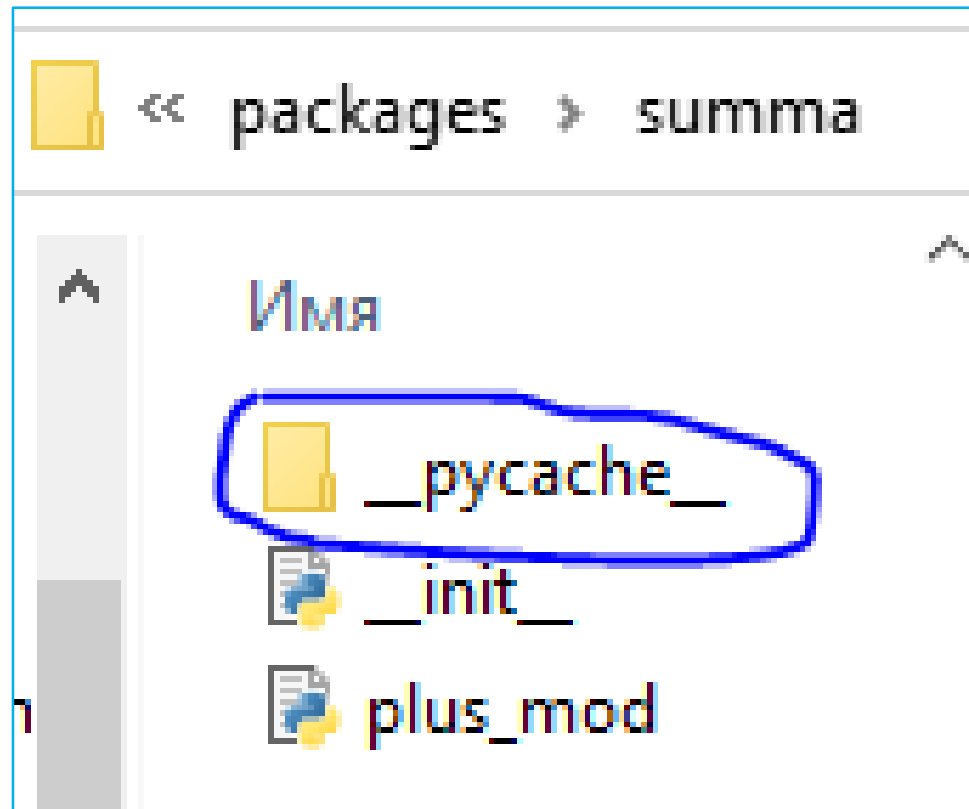
```
File
Python 3.7.4 Shell
D64)
Type
>>>
= RE
- 1
180
>>>
```

результат сложения
равен 180



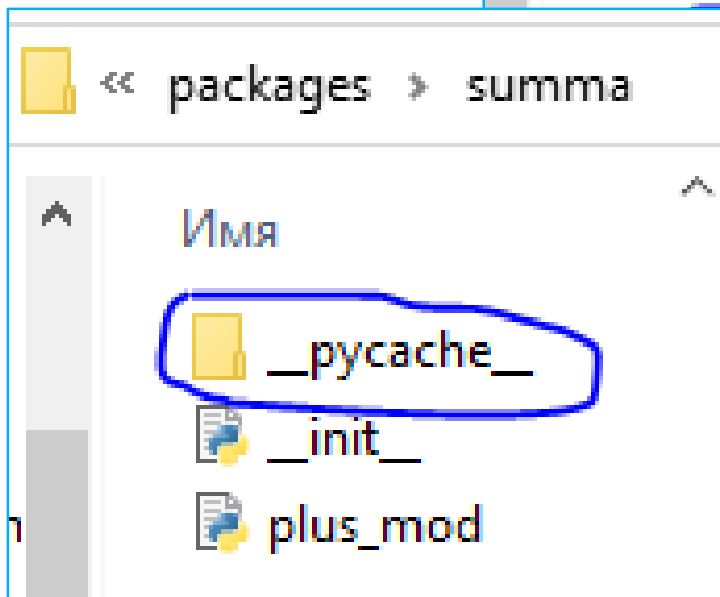
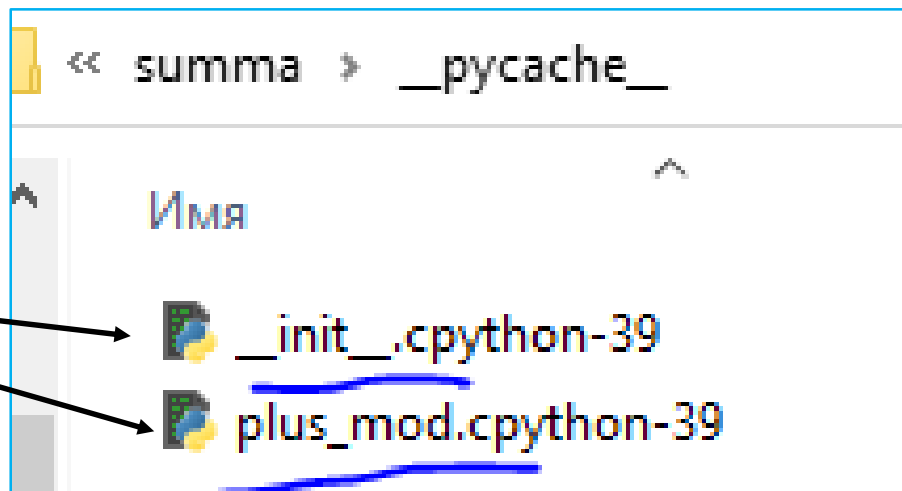
Шаг 11) Проверим нашу папку `packages/summa`

Как видно появилась папка `__pycache__`



Шаг 12) Заглянем в папку `__pycache__`

это байткод с расширением `.pyc`



Импорт пакета или модуля – относительно дорогостоящее мероприятие, поэтому Python предпринимает некоторые трюки для ускорения этого процесса. Один из способов – создать байт-компилированные файлы (или байткод) с расширением `.pyc`, которые являются некой промежуточной формой, в которую Python переводит программу. Такой файл `.pyc` полезен как при импорте модулей так и при импорте пакетов с их модулями в следующий раз в другую программу – это произойдёт намного быстрее, поскольку значительная часть обработки, требуемой при импорте модуля, будет уже проделана. Этот байткод также является платформо-независимым.



Важно

Обращайте внимание на имена файлов,
имена функций и переменных. Это важно
при использовании модулей

