Лабораторная работа 2.13 Модули и пакеты

Цель работы: *приобретение навыков по работе с модулями и пакетами языка программирования Python версии 3.x.*

Ход работы

Модули и пакеты значительно упрощают работу программиста. Классы, объекты, функции и константы, которыми приходится часто пользоваться можно упаковать в модуль, и, в дальнейшем, загружать его в свои программы при необходимости. Пакеты позволяют формировать пространства имен для работы с модулями.

Модули в Python

Что такое модуль в Python?

Под модулем в *Python* понимается файл с расширением *.py*. Модули предназначены для того, чтобы в них хранить часто используемые функции, классы, константы и т. п. Можно условно разделить модули и программы: программы предназначены для непосредственного запуска, а модули для импортирования их в другие программы. Стоит заметить, что модули могут быть написаны не только на языке *Python*, но и на других языках (например *C*).

Как импортировать модули в Python?

Самый простой способ импортировать модуль в *Python* это воспользоваться конструкцией:

```
import имя_модуля
```

Импорт и использование модуля *math*, который содержит математические функции, будет выглядеть вот так.

```
>>> import math
>>> math.factorial(5)
120
```

За один раз можно импортировать сразу несколько модулей, для этого их нужно перечислить через запятую после слова *import*:

```
import имя_модуля1, имя_модуля2
```

```
>>> import math, datetime
>>> math.cos(math.pi/4)
0.707106781186547
>>> datetime.date(2017, 3, 21)
datetime.date(2017, 3, 21)
```

Если вы хотите задать псевдоним для модуля в вашей программе, можно воспользоваться вот таким синтаксисом:

```
import имя_модуля as новое_имя
```

```
>>> import math as m
>>> m.sin(m.pi/3)
0.866025403784438
```

Используя любой из вышеперечисленных подходов, при вызове функции из импортированного модуля, вам всегда придется указывать имя модуля (или псевдоним). Для того, чтобы этого избежать делайте импорт через конструкцию *from ... import...*

```
from имя_модуля import имя_объекта
```

```
>>> from math import cos
>>> cos(3.14)
0.999998731727539
```

При этом импортируется только конкретный объект (в нашем примере: функция *cos*), остальные функции недоступны, даже если при их вызове указать имя модуля.

```
>>> from math import cos
>>> cos(3.14)
-0.999998731727539
>>> sin(3.14)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    sin(3.14)
NameError: name 'sin' is not defined
>>> math.sin(3.14)
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    math.sin(3.14)
NameError: name 'math' is not defined
```

Для имортирования нескольких функций из модуля, можно перечислить их имена через запятую.

```
from имя_модуля import имя_объекта1, имя_объекта2
```

```
>>> from math import cos, sin, pi
>>> cos(pi/3)
0.500000000000000
>>> sin(pi/3)
0.866025403784438
```

Импортируемому объекту можно задать псевдоним.

```
from имя_модуля import имя_объекта as псевдоним_объекта
```

```
>>> from math import factorial as f
>>> f(4)
24
```

Если необходимо импортировать все функции, классы и т. п. из модуля, то воспользуйтесь следующей формой оператора `from … import …*`

```
from имя_модуля import *
```

```
>>> from math import *
>>> cos(pi/2)
6.123233995736766e-17
>>> sin(pi/4)
0.707106781186547
>>> factorial(6)
720
```

Пакеты в Python

Что такое пакет в Python?

Пакет в *Python* – это каталог, включающий в себя другие каталоги и модули, но при этом дополнительно содержащий файл `__init__.py`. Пакеты используются для формирования пространства имен, что позволяет работать с модулями через указание уровня вложенности (через точку).

Для импортирования пакетов используется тот же синтаксис, что и для работы с модулями.

Использование пакетов в Python

Рассмотрим следующую структуру пакета:

```
fincalc
|-- __init__.py
|-- simper.py
|-- compper.py
|-- annuity.py
```

Пакет *fincal* содержит в себе модули для работы с простыми процентами (simper.py), сложными процентами (compper.py) и аннуитетами (annuity.py).

Для использования фукнции из модуля работы с простыми процентами, можно использовать один из следующих вариантов:

```
import fincalc.simper
fv = fincalc.simper.fv(pv, i, n)
import fincalc.simper as sp
fv =sp.fv(pv, i, n)
from fincalc import simper
fv = simper.fv(pv, i, n)
```

Файл `__init__.py` может быть пустым или может содержать переменную `__all__`, хранящую список модулей, который импортируется при загрузке через конструкцию

```
from имя_пакета import *
```

Например для нашего случая содержимое `__init__.py` может быть вот таким:

```
__all__ = ["simper", "compper", "annuity"]
```

Указания по технике безопасности

При работе на ЭВМ без разрешения руководителя занятия запрещается:

- подавать (снимать) напряжение на ПЭВМ и электрические розетки с распределительного щита;
- включать и выключать блоки питания ПЭВМ и мониторы;
- извлекать ПЭВМ из защитного кожуха;
- устранять неисправности, возникшие в ходе выполнения лабораторной работы.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Выполните клонирование созданного репозитория.
4. Дополните файл `.gitignore` необходимыми правилами для работы с IDE PyCharm.
5. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.
6. Создайте проект PyCharm в папке репозитория.
7. Выполните индивидуальные задания. Приведите в отчете скриншоты работы программ решения индивидуального задания.
8. Зафиксируйте сделанные изменения в репозитории.
9. Добавьте отчет по лабораторной работе в *формате PDF* в папку *doc* репозитория. Зафиксируйте изменения.
10. Выполните слияние ветки для разработки с веткой *master/main*.
11. Отправьте сделанные изменения на сервер GitHub.
12. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Индивидуальные задания

Задание 1

Выполнить индивидуальное задание лабораторной работы 2.11, оформив все функции программы в виде отдельного модуля. Разработанный модуль должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Номер варианта уточнить у преподавателя.

Задание 2

Выполнить индивидуальное задание лабораторной работы 2.8, оформив все классы программы в виде отдельного пакета. Разработанный пакет должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Настроить соответствующим образом переменную `__all__` в файле `__init__.py` пакета. Номер варианта уточнить у преподавателя.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется электронно в формате PDF, должен содержать ответы на контрольные вопросы, ссылку на репозиторий с которым выполнялась работа, скриншоты IDE PyCharm, скриншоты результатов работы программ.

Вопросы для защиты работы

1. Что является модулем языка Python?
2. Какие существуют способы подключения модулей в языке Python?
3. Что является пакетом языка Python?
4. Каково назначение файла `__init__.py`?
5. Каково назначение переменной `__all__` файла `__init__.py`?