**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Програмування**

**Лабораторна робота №6**

**«Словники та функції користувача»**

Виконала:

студентка групи ІО-82

Тарасенко Юлія

Залікова книжка № 8222

Перевірив асистент Шевело

Київ 2018 р.

**Тема**: «Основи об’єктно-орієнтованого програмування. Модулі та пакети».

**Мета**: вивчити способи створення та підключення модулів та пакетів. Основи ООП. Методи і атрибути класів та робота з ними. Побудова програми у стилі ООП.

**Загальне завдання**:

1. Вивчити матеріал лекцій 18, 19, 20 та 21.
2. Виконати індивідуальне завдання лабораторної роботи, вибране відповідно до варіанту.

**Теоретичні відомості**

**Модулі й пакети**

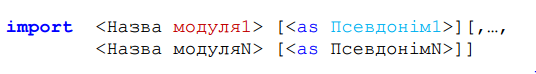
Модулем у мові Python називають будь-який файл з програмним кодом.

Кожний модуль може імпортувати інший модуль, одержуючи, таким чином, доступ до атрибутів (змінних, функцій і класів), оголошених усередині імпортованого модуля.

Слід відмітити, що імпортований модуль може містити програму не тільки мовою Python – так, можна імпортувати скомпільований модуль, написаний мовою С. Усі програми, які ми запускали раніше, були розташовані в модулі з назвою "\_\_main\_\_".

Одержати ім'я модуля дозволяє визначений атрибут \_\_name\_\_.

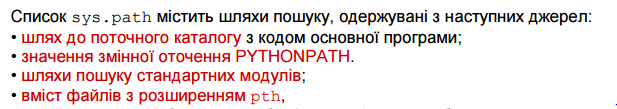
Імпортувати модуль дозволяє інструкція **import**.



Доступ до атрибутів модуля здійснюється за допомогою точкової нотації Наприклад, звернутися до

константи pi, розташованої усередині модуля math, можна так:





Пакети

Пакетом називають каталог з модулями, у якому розташований файл ініціалізації \_\_init\_\_.py. Файл ініціалізації може бути порожнім або містити код, який буде виконаний при першому доступі до пакета.

pip – використовується для установки і управління програмними пакетами, написаними на Python. Для версії Python 3.5, pip постачається разом з інтерпретатором Python.

**Об'єктно-орієнтоване програмування**

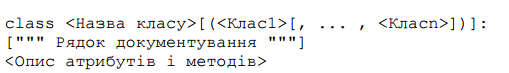
Об'єктно-орієнтоване програмування (ООП) – це спосіб організації програми, що дозволяє використовувати той самий код багаторазово. На відміну від функцій і модулів, ООП дозволяє не тільки розділити програму на фрагменти, але й описати предмети реального світу у вигляді зручних сутностей – об'єктів, а також організувати зв'язки між цими об'єктами

Основною «цеглинкою» ООП є клас. Клас – це складний тип даних, що включає набір змінних і функцій для керування значеннями, що зберігаються в цих змінних.

Змінні називають атрибутами, а функції – методами. Клас є фабрикою об'єктів, тобто дозволяє створити необмежену кількість екземплярів, заснованих на цьому класі.

ООП ґрунтується на трьох основних концепціях розробки: інкапсуляція, спадкування й поліморфізм.

Клас описують за допомогою ключового слова class за наступною схемою:



Щоб використовувати атрибути й методи класу, необхідно створити екземпляр класу згідно з наступним синтаксисом



При доступі до методів класу використовують такий формат: 

**Інкапсуляція** - це механізм, який об'єднує дані і методи, що маніпулюють цими даними, і захищає і те і інше від зовнішнього втручання або неправильного використання. Коли методи і дані об'єднуються у такий спосіб, створюється об'єкт.

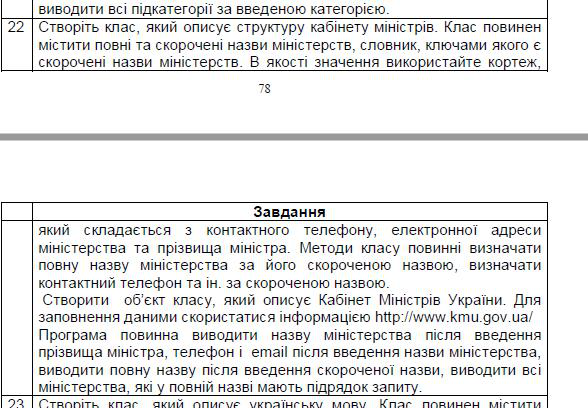
***Наслідування***– процес, завдяки якому один об’єкт може придбати властивості іншого, тобто наслідувати властивість іншого обєкту і додавати риси характерні тільки для нього самого.

***Поліморфізм***– властивість, яка дозволяє одне і те саме ім’я використовувати для вирішення декількох технічно різних задач, тобто основною метою поліморфізму є використання одного імені для задання загальних класу дій. На практиці це значить спроможність об’єктів вибирати внутрішній метод або процедуру, виходячи із типу даних, прийнятих в повідомленні.

**Алгоритм розв`язування завдання**

* Для того щоб виконати завдання потрібно спочатку створити клас
* В середині класу створити методи та забезпечити вивід даних
* В основній програмі створити екземпляри
* ПІсля вводу міста, перевірити чи в програмі є інформація про це місто
* Якщо інформація присутня – вивести дані
* Якщо інформація відсутня – сповістити про це користувача

**Завдання**

****

**фрагменту тексту програми, який написаний індивідуально.**

**import** lab5  
  
**class Ministry:  
 def \_\_init\_\_(**self, name, shrt\_name, telephone, email, surname**):** self.name **=** name  
 self.shrt\_name **=** shrt\_name  
 self.telephone **=** telephone  
 self.email **=** email  
 self. surname **=** surname  
 **def** \_\_str\_\_**(**self**):  
 return** str**(**self.name**)**, str**(**self.telephone**)**, str**(**self.shrt\_name**)**, str**(**self.email**)**, str**(**self.surname**)**short\_min **= ['Ministry\_of\_Agrarian\_Policy'**, **'Ministry\_of\_Internal\_Affairs'**, **'Ministry\_of\_Environmental\_Protection'**,  
 **'Ministry\_of\_Economic\_Development\_and\_Trade'**, **'Minenergugol'**, **'MZS'**, **'MIP'**,**'Mininfrastructure'**,  
 **'Minculture'**, **'Ministry\_of\_Minority of Life'**, **'Ministry of Defense'**, **'MES'**, **'MoH'**,**'Minregion'**,  
 **'Minsotspolitiki'**,**'Ministry of Finance'**, **'Ministry of Justice'**, **'MTOT']**globl\_list **= [**i **for** i **in** list**(**zip**(**short\_min, lab5.ministry, lab5.short\_min, lab5.telephone, lab5.email, lab5.surname**))]**# print(globl\_list)  
  
**for** name, val1, val2, val3, val4, val5 **in** globl\_list**:** globals**()[**name**] =** Ministry**(**val1,val2, val3, val4, val5**)**# print(Ministry\_of\_Agrarian\_Policy.name)  
# print(globals())  
  
**class Gov:  
 def \_\_init\_\_(**self, **\***gov**):** self.governments **=** list**(**gov**)  
 def addGov(**self, gov**):** self.governments.append**(**gov**)  
 def showAll(**self**):  
 for** gov **in** self.governments**:** print**(**gov**)  
 def fullname(**self**):** fullinput **=** input**("Введите сокращенное название министерства: ")  
 for** i **in** self.governments**:  
 if** fullinput **==** i.shrt\_name**:** print**("Полное название министерства: " +** i.name**)  
 def fullname2(**self**):** input2 **=** input**("Введите фамилию министра: ")  
 for** i **in** self.governments**:  
 if** input2 **==** i.surname**:** print**("Данный министр работает в этом министерстве: " +** i.name**)  
 def inf(**self**):** input3 **=** input**("Введите сокращенное название министерства: ")  
 for** i **in** self.governments**:  
 if** input3 **==** i.shrt\_name**:** print**("Телефон министерства: " +** i.telephone**)** print**("E-mail министерства: " +** i.email**)  
 def search(**self**):** input4 **=** input**("Найти все совпадения с: ")  
 for** i **in** self.governments**:  
 if** input4 **in** i.name**:** print**(**i.name**)**Government **=** Gov**()**d **=** globals**()**.copy**()**# print(d)  
**for** key **in** d**:  
 if** key **in** short\_min**:** Government.addGov**(**d**[**key**])**# print(Government.governments[0].name)  
# print(Government.governments[0].\_\_dict\_\_)  
# Government.fullname()  
# Government.fullname2()  
# Government.inf()  
# Government.search()

**Аналіз результатів та висновки.**

Під час виконання роботи проблем не виникло