Programowanie w Pythonie
Klasy cd. cz 2

dr Agnieszka Zbrzezny

Ćwiczenia sprawdzające podstawową wiedzę z klas

1. Napisz program importujący wbudowany moduł `array` i wyświetlający przestrzeń nazw tego modułu.

2. Napisz program, który utworzy klasę i wyświetli przestrzeń nazw wspomnianej klasy.

3. Napisz program, który utworzy instancję określonej klasy i wyświetli przestrzeń nazw tej instancji.

4. Moduł `builtins` zapewnia bezpośredni dostęp do wszystkich 'wbudowanych' identyfikatorów Pythona. Napisz program, który zaimportuje funkcję `abs()` za pomocą wbudowanego modułu, wyświetli dokumentację funkcji `abs()` i znajdzie wartość bezwzględną z -155.

5. Zdefiniuj klasę `Student` (atrybuty: `nazwa_ucznia`, `klasa_ucznia`, `student_id`). Używając atrybutów funkcji wyświetlaj nazwy wszystkich argumentów.

6. Napisz metodę `student_data()`, która wyświetli identyfikator ucznia (`student_id`). Jeśli użytkownik przekaże argument `nazwa_ucznia` lub `klasa_ucznia`, funkcja wyświetli nazwę ucznia i klasę.

7. Napisz prostą klasę o nazwie `Student` i wyświetl jej typ. Wyświetl także klucze atrybutów `__dict__` i wartość atrybutu `__module__` klasy `Student`.

8. Napisz program, w którym utworzysz dwie puste klasy, `Student` i `Marks`. Teraz utwórz kilka instancji i sprawdź, czy są to instancje wspomnianych klas, czy nie. Sprawdź również, czy wspomniane klasy są podklasami wbudowanej klasy `Object`, czy nie.

9. Napisz klasę nazwie `Student` z dwoma atrybutami `student_name`, `marks`. Zmodyfikuj wartości atrybutów wspomnianej klasy i wydrukuj oryginalne i zmodyfikowane wartości wspomnianych atrybutów.

10. Napisz klasę o nazwie `Student` z dwoma atrybutami `student_id`, `student_name`. Poza klasą dodaj nowy atrybut `student_class` i wyświetl cały atrybut wraz z wartościami wspomnianej klasy. Teraz usuń atrybut `student_name` i wyświetl cały atrybut z wartościami.

    Przykładowe wyjście:

    ```
    Original attributes and their values of the Student class:
    student_id V10
    student_name Adam Nowak
    After adding the student_class, attributes and their values with
    the said class:
    ```

```
student_id V10
student_name Adam Nowak
student_class V
After removing the student_name, attributes and their values
from the said class:
student_id V10
student_class V
```

11. Napisz klasę o nazwie `Student` i utwórz dwie instancje `student_1`, `student_2` i przypisz podane wartości do wspomnianych atrybutów instancji. Wydrukuj wszystkie atrybuty instancji `student_1`, `student_2` z ich wartościami w podanym formacie.

Niewiele trudniejsze ćwiczenia sprawdzające podstawową wiedzę

1. Napisz klasę, która ma dwie metody `get_string` i `print_string`. `get_string` akceptuje ciąg znaków od użytkownika, a `print_string` drukuje ciąg wielkimi literami.
2. Napisz klasę o nazwie `Rectangle` skonstruowaną przez długość i szerokość oraz metodę, która obliczy pole prostokąta.
3. Napisz klasę o nazwie `Circle` skonstruowaną przez promień i dwie metody, które obliczą pole i obwód koła.
4. Napisz program, który uzyska nazwę klasy instancji w Pythonie.

Klasy

1. Zdefiniuj klasę o nazwie `American`, która ma statyczną metodę o nazwie `print_nationality`. (Użyj dekoratora staticmethod, aby zdefiniować metodę statyczną klasy.)
2. Zdefiniuj klasę o nazwie `American` i jej podklasę `NewYorker`.
3. Zdefiniuj klasę o nazwie `Shape` i jej podklasę `Square`. Klasa `Square` posiada inicjalizator, który jako argument przyjmuje długość. Obie klasy mają funkcję `area`, która może zwrócić kształtu, który domyślnie wynosi 0.
4. W pliku `polynomial.py` zaimplementuj własną (Nie korzystając z analogicznych klas z bibliotek Pythona!) klasę `Polynomial`, reprezentującą wielomian pojedynczej zmiennej `x`. Klasa powinna zawierać metody:
   - `__init__(self, coefficients)` – inicjalizator tworzący wielomian. Atrybut `coefficients` to lista współczynników wielomianu stojących kolejno przy coraz większych potęgach `x`, np. `coefficients = [5, 4, 3, 0, 1]` reprezentuje wielomian $1 * x^4 + 0 * x^3 + 3 * x^2 + 4 * x + 5$.
   - `deg(self)` – zwraca stopień wielomianu.
   - `__str__(self)` – zwraca napis reprezentujący self.
   - `__neg__(self)` – zwraca wielomian (instancję Polynomial) odpowiadający `-self`.

- `__add__(self, other_poly)` – zwraca wielomian odpowiadający `self + other_poly`. Zastanów się czy w przypadku dodawania, odejmowania i mnożenia można sobie ułatwić zadanie pisząc metode pomocniczą. Zaimplementuj ją. (Przetestuj dla `pol_1.coefficients = [3, 1, 1]` i `pol_2.coefficients = [5, 2]`).
- `__sub__(self, other_poly)` – zwraca wielomian odpowiadający `self - other_poly`.
- `__mul__(self, other_poly)` – zwraca wielomian odpowiadający `self * other_poly`.
- `__eq__(self, other_poly)` – zwraca `True`, gdy wielomiany `self` i `other_poly` są równe i `False` w przeciwnym przypadku.
- `__call__(self, x)` – zwraca wartość wielomianu w punkcie `x`.

Następnie przetestuj każdą z tych metod w funkcji `main()` w pliku `test_polynomial.py`.

5. Napisz klasę `Vector`. Zaimplementuj metody: `__init__`, `__repr__`, `__add__`, `__sub__`, `__mul__`, `__eq__`, `__len__`, `__getitem__`, `__str__` oraz metody `norm` - zwraca normę (długość, wielkość) wektora, `inner` – zwraca iloczyn skalarny (iloczyn skalarny) siebie i innego wektora. Przetestuj te metody w osobnym pliku w funkcji `main`.

6. Wykorzystując atrybut `__slots__` zaimplementuj w pliku `osoba.py` klasę `Osoba` z właściwościami `nazwisko` oraz `rok_urodzenia`. Dla obu własności zdefiniuj setter oraz deleter. Zdefiniuj także metodę specjalną `__str__`. Dodaj do klasy `Osoba` metodę klasową o nazwie `get_ile(cls)`, która zwraca wartość zmiennej `cls._ile` oraz metode `zwieksz_pobory(ile_procent)`. Wykorzystując atrybut `__slots__` zaimplementuj w pliku `osoba.py` klasę `Pracownik` dziedziczącą po klasie `Osoba` z własnościami `rok_zatrudnienia` oraz `pobory`. Dla obu własności `nazwisko` oraz `rok_urodzenia` zdefiniuj setter oraz deleter. Zdefiniuj także metodę specjalną `__str__`.

   W pliku test_osoba.py w funkcji main():

   - Na rzecz klasy `Osoba` wywołaj metodę `get_ile()`.
   - Utwórz jedną instancję klasy `Osoba` i przetestuj każdą z metod tej klasy w tym metodę `get_ile()`.
   - Utwórz jedną instancję klasy `Pracownik` i przetestuj każdą z metod tej klasy w tym metodę `get_ile()`.
   - Usuń utworzone instancje i na rzecz klasy `Pracownik` wywołaj metodę `get_ile()`.