Nazwa implementacji: Nauka języka Python - klasy

Autor: Piotr Fiorek

Opis implementacji: Poznanie czym są klasy i jak dzięki nim tworzyć efektywniejsze programy.

Klasy są najtrudniejszym koncepcyjnie z zagadnień w Pythonie. W dużym uproszczeniu – wszystkie typy danych, które do tej pory poznaliśmy to osobne typy danych, a tworząc zmienną danego typu, tworzymy obiekt typu danej klasy. Zatem jeśli tworzymy zmienną przetrzymującą cyfrę, tworzymy obiekt typu klasy „int", a tworząc zmienną przechowującą listę, tworzymy obiekt typu klasy „list". Obiekty danych klas mają tę właściwość, że poza przetrzymywaniem danych mają też własne, prywatne funkcje do operowania na zmiennych, które przechowują. Python jak wszystkie języki obiektowe pozwala na tworzenie własnych klas. Składniowo jest to bardzo proste i wygląda tak:

```
class NaszaNowaKlasa:

pass
```

W ten sposób stworzyliśmy nową klasę o nazwie „NaszaNowaKlasa". Tak jak w przypadku przykładu funkcji, klasa ta składa się jedynie ze słowa kluczowego „pass", a więc nic nie robi i nie jest do niczego przydatna. Aby stworzyć klasę, która może nam się do czegoś przydać trzeba dodać do niej funkcje, które będą wykonywały dla nas operacje na danych które umieścimy w naszej klasie. Na początek stworzymy klasę, która będzie służyła do przetrzymywania informacji o figurze geometrycznej, jaką jest prostokąt. Aby opisać prostokąt, potrzebujemy tylko dwóch informacji – długości obydwu boków. Każda klasa, która ma być do czegoś przydatna, potrzebuje specjalnej funkcji nazywanej konstruktorem. Konstruktor jest to specjalna funkcja, która jest automatycznie wywoływana w momencie tworzenia nowego obiektu danej klasy. Funkcja ta przyjmuje jako parametry dane, które chcemy umieścić w obiekcie naszej klasy. W Pythonie, konstruktor ma specjalną nazwę „

__init__

":

```
>>> class Kwadrat:

... def __init__(self, x, y):

... self.jeden_bok = x

... self.drugi_bok = y
```

Jak widać, nasz konstruktor tworzymy tak samo, jak każdą inną funkcję tylko, że wewnątrz klasy. Funkcja ta tak naprawdę przyjmuje trzy parametry, ale pierwszy z nich to specjalny parametr zwyczajowo nazywany „self". Nie należy się nim przejmować, ale należy pamiętać, żeby zawsze umieścić go jako pierwszy parametr każdej funkcji wewnątrz tworzonej klasy. Pozostałe dwa parametry to boki naszego kwadratu. Wewnątrz konstruktora zmienne podane jako parametry zapisujemy jako prywatne zmienne obiektu typu tej klasy, który będzie tworzony przy użyciu tego konstruktora. Tworzenie obiektów typu „Kwadrat":

```
>>> a = Kwadrat(3, 4)

>>> b = Kwadrat(7, 11)

>>> a.jeden_bok

3

>>> a.drugi_bok

4

>>> b.jeden_bok
```


KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
ORE
FWIOO
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt "Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych" współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

```
7
```

```
>>> b.drugi_bok
```

```
11
```

„a” oraz „b” są obiektami klasy „Kwadrat”. Dane podawane im podczas ich tworzenia są przez nie zapamiętywane i są prywatne dla każdego z nich. Dodajmy do naszej klasy funkcję liczącą pole kwadratu:

```
def pole(self):
```

```
pole = self.jeden_bok * self.drugi_bok
```

```
return pole
```

Oczywiście funkcję należy wpisać jako część klasy „Kwadrat”, a więc odpowiednio wciętą (w interpreterze trzeba jeszcze raz zadeklarować całą klasę, razem z nową funkcją). Wywołuje się ją tak jak, już wcześniej widzieliśmy np. przy obiektach typu listy, czyli nazwa obiektu, potem kropka, a potem wywołanie naszej funkcji. Czyli jeśli jeszcze raz zdefiniujemy naszą klasę wraz z nową metodą i utworzymy obiekt „a” z parametrami o wartościach „3” i „4” to wywołanie wyglądać będzie tak:

```
>>> print(a.pole())
```

```
12
```

Funkcja nie przyjmuje żadnych parametrów (poza „self”, ale on jest ważny tylko podczas deklarowania funkcji), więc wpisujemy jej nazwę oraz puste nawiasy. A gdybyśmy chcieli cały nasz obiekt podać jako parametr funkcji „print”? Jeśli zrobimy to z naszą klasą w takiej formie w, jakie jej teraz to dostaniemy mniej więcej taki wynik:

```
>>> print(a)
```

```
<__main__.Kwadrat object at 0x7fc0324d6610>
```

Nie jest to to, czego się spodziewaliśmy i czego chcieliśmy. Aby funkcja „print” wiedziała, jak poprawnie wydrukować nasz obiekt, musimy wcześniej w klasie zadeklarować specjalną funkcję o nazwie „__str__”. Jest ona później automatycznie wywoływana przez „print” za każdym razem, kiedy chcemy wyświetlić nasz obiekt.

```
def __str__(self):
```

```
opis = 'jeden bok: ' + str(self.jeden_bok) + '\ndrugi bok: ' + str(self.drugi_bok)
```

```
return opis
```

Jest to po prostu funkcja o specjalnej nazwie, nieprzyjmująca żadnych parametrów, która zwraca napis. To właśnie ten napis będzie drukowany przez funkcję „print”. Rzeczą, na którą warto zwrócić uwagę w tym przykładzie jest to, że długości boków podajemy jako parametry funkcji „str” i dopiero potem sklejamy przy użyciu plusa z innymi napisami. Robimy tak ponieważ funkcja „str” zamienia liczby na napisy (a tak naprawdę tworzy obiekty klasy „string”, których nie przypisujemy do żadnych zmiennych, ale wykorzystujemy, aby dodać do innych napisów) i dopiero te napisy możemy dodawać do innych napisów. Gdybyśmy pominęli przekazanie długości boków do funkcji i od razu próbowali je sklejać z tekstem to, Python zwróciłby błąd mówiący, że nie wie, w jaki sposób dodać cyfrę do napisu. W Pythonie jest wiele takich specjalnych funkcji, które mają swoje zastosowania i są wykorzystywane przez samego Pythona w różnych sytuacjach. Pełny spis tych funkcji, wraz z wyjaśnieniami można znaleźć w dokumentacji pod adresem:http://http://docs.python.org/3/reference/datamodel.html#special-method-names. Gotowe klasy służące do wielu przydatnych rzeczy można znaleźć w tak zwanych modułach. Opis wszystkich standardowych modułów jest pod adresem: http://docs.python.org/3/library/index.html. Znajduje się tam lista modułów, a pod linkiem do każdego z nich opis klas, oraz funkcji z tych klas, które dostarcza. Aby użyć jakiegoś modułu w naszym programie, wystarczy na początku dopisać „import <nazwa-modułu”. Każdy moduł wystarczy zaimportować tylko raz, na początku pliku z kodem. Jeśli byśmy np. chcieli zaimportować moduł

KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI ORE FWIOO UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt “Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych” współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

„calendar”

i przy użyciu funkcji o takiej samej nazwie wydrukować kalendarz na rok 2014 wyglądałoby to tak:

```
>>> import calendar

>>> print(calendar.calendar(2014))
```

```
                                  2014

      January                   February                  March
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
                   1                1  2  3  4  5                1  2  3  4
 2  3  4  5  6  7  8       6  7  8  9 10 11 12       5  6  7  8  9 10 11
 9 10 11 12 13 14 15      13 14 15 16 17 18 19      12 13 14 15 16 17 18
16 17 18 19 20 21 22      20 21 22 23 24 25 26      19 20 21 22 23 24 25
23 24 25 26 27 28 29      27 28 29                  26 27 28 29 30 31
30 31

      April                     May                       June
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
                   1             1  2  3  4  5  6                1  2  3
 2  3  4  5  6  7  8       7  8  9 10 11 12 13       4  5  6  7  8  9 10
 9 10 11 12 13 14 15      14 15 16 17 18 19 20      11 12 13 14 15 16 17
16 17 18 19 20 21 22      21 22 23 24 25 26 27      18 19 20 21 22 23 24
23 24 25 26 27 28 29      28 29 30 31               25 26 27 28 29 30
30

       July                    August                  September
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
                   1             1  2  3  4  5                   1  2
 2  3  4  5  6  7  8       6  7  8  9 10 11 12       3  4  5  6  7  8  9
 9 10 11 12 13 14 15      13 14 15 16 17 18 19      10 11 12 13 14 15 16
16 17 18 19 20 21 22      20 21 22 23 24 25 26      17 18 19 20 21 22 23
23 24 25 26 27 28 29      27 28 29 30 31            24 25 26 27 28 29 30
30 31

      October                   November                  December
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
 1  2  3  4  5  6  7                1  2  3  4                   1  2
 8  9 10 11 12 13 14       5  6  7  8  9 10 11       3  4  5  6  7  8  9
15 16 17 18 19 20 21      12 13 14 15 16 17 18      10 11 12 13 14 15 16
22 23 24 25 26 27 28      19 20 21 22 23 24 25      17 18 19 20 21 22 23
29 30 31                  26 27 28 29 30            24 25 26 27 28 29 30
                                                    31
```

Aby użyć czegoś z modułu, musimy nazwę „tego czegoś” poprzedzić nazwą moduły i kropką, dokładnie tak jak robiliśmy z funkcjami prywatnymi klasy. Robimy tak ponieważ w tej chwili wykorzystaliśmy funkcję prywatną modułu. Zadania:

Napisać klasę reprezentującą człowieka (zmienne takie jak imię, nazwisko, płeć, wiek, wzrost), posiadającą kilka podstawowych funkcji i wykorzystać ją w programie.

Napisz klasę bazową reprezentującą samochód i dwie klasy dziedziczące po niej – jedną reprezentującą samochód osobowy z dodatkowymi metodami do „obsługiwania” liczby pasażerów i i drugą reprezentującą samochód ciężarowy z metodami do obsługi typu i masy ładunku.

KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
ORE
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt “Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych” współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.