
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki		
Przedmiot	Skryptowe języki programowania		
Prowadzący	mgr inż. Martyna Tarczewska		
Temat	<i>Obiektość Pythona</i>		
Student	Marcin Ogórkiewicz		
Nr ćw.	4	Data wykonania	29.10.2023
Ocena		Data oddania spr.	29.10.2023

Zadanie 1

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
from random import randint
from typing import List

# stałe i zmienne globalne

# funkcje

class Student:
    def __init__(self):
        self.name = ""
        self.last_name = ""
        self.index = ""
        self.marks = []
        self.avg = float

    def give_name(self, name: str, last_name: str) -> None:
        self.name = name
        self.last_name = last_name

    def give_index(self) -> None:
        self.index = str(randint(100000, 1000000))

    def give_mark(self, mark: int) -> None:
        self.marks.append(mark)

    def get_marks(self) -> List[int]:
        return self.marks
```

```

def get_avg(self) -> float:
    amount = len(self.marks)
    sum_of_grades = 0
    for i in range(amount):
        sum_of_grades += self.marks[i]
    self.avg = sum_of_grades/amount
    return self.avg

def say_hello(self) -> None:
    print("Hello! I'm " + self.name + " " + self.last_name + " " +
self.index)

def main() -> None:
    s = Student()
    s.give_index()
    s.give_name("Jane", "Doe")
    s.give_mark(5) # wywołanie sposób 1
    Student.give_mark(s, 3) # wywołanie sposób 2
    print(s.get_marks(), s.get_avg())
    s.say_hello()

main()

```

```

C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab04\Zadanie1.py
[5, 3] 4.0
Hello! I'm Jane Doe 925725

Process finished with exit code 0

```

Zadanie 2

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

class Vehicle:

    def __init__(self, owner: str, table: str):
        self.owner = owner
        self.table = table

    def get_owner(self) -> str:
        return self.owner

    def get_sound(self):
        print("vehicle's brum brum")

```

```
class Car(Vehicle):

    def get_sound(self) -> None:
        print("car's brum brum")

def main() -> None:
    vehi = Vehicle("Andrzej", "PY001KY")
    print(vehi.get_owner())
    vehi.get_sound()
    cr = Car("Marcin", "CB001KY")
    print(cr.get_owner())
    cr.get_sound()

main()

"""Dla instancji klasy Car() wykonuje się metoda get_sound zawarta w klasie
Car(),
podobnie dla Vehicle() wykonuje się metoda get_sound zawarta w klasie
Vehicle(). Metody get_owner nie dało się
wykonać dla klasy Vehicle(), ponieważ ta metoda należy do klasy Car().
Można to naprawić przenosząc metodę __init__ do klasy nadrzędnej Vehicle(),
wraz z metodą get_owner."""
```

```
C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab04\Zadanie2.py
Andrzej
vehicle's brum brum
Marcin
car's brum brum

Process finished with exit code 0
```

Zadanie 3

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
from random import randint
from typing import List

# stałe i zmienne globalne

# funkcje

class Student:
    quantity = 0

    def __init__(self):
        self.name = ""
        self.last_name = ""
        self.index = ""
        self.marks = []
        self.avg = float
```

```

        Student.quantity += 1

    def give_name(self, name: str, last_name: str) -> None:
        self.name = name
        self.last_name = last_name

    def give_index(self) -> None:
        self.index = str(randint(100000, 1000000))

    def give_mark(self, mark: int) -> None:
        self.marks.append(mark)

    def get_marks(self) -> List[int]:
        return self.marks

    def get_avg(self) -> float:
        amount = len(self.marks)
        sum_of_grades = 0
        for i in range(amount):
            sum_of_grades += self.marks[i]
        self.avg = sum_of_grades/amount
        return self.avg

    def say_hello(self) -> None:
        print("Hello! I'm " + self.name + " " + self.last_name + " " +
self.index)

def main() -> None:
    a = Student()
    b = Student()
    c = Student()
    d = Student()
    e = Student()
    print(Student.quantity)

main()
"""Obiekty nie mają dostępu do zmiennej, ten ma jedynie klasa Student()"""

```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab04\Zadanie3.py
5
Process finished with exit code 0

```

Zadanie 4

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
from random import randint
from typing import List

# stałe i zmienne globalne

```

```

# funkcje

class Student:
    quantity = 0

    def __init__(self):
        self.name = ""
        self.last_name = ""
        self.index = ""
        self.marks = []
        self.avg = float
        Student.quantity += 1

    def give_name(self, name: str, last_name: str) -> None:
        self.name = name
        self.last_name = last_name

    def give_index(self) -> None:
        self.index = str(randint(100000, 1000000))

    def empty_name_check(self) -> bool:
        if self.name or self.last_name == "":
            return True
        else:
            return False

    def give_mark(self, mark: int) -> None:
        self.marks.append(mark)

    def get_marks(self) -> List[int]:
        return self.marks

    def get_avg(self) -> float:
        amount = len(self.marks)
        sum_of_grades = 0
        for i in range(amount):
            sum_of_grades += self.marks[i]
        self.avg = sum_of_grades/amount
        return self.avg

    def say_hello(self) -> None:
        print("Hello! I'm " + self.name + " " + self.last_name + " " +
self.index + " " + str(self.empty_name_check()))

def main() -> None:
    a = Student()
    b = Student()
    c = Student()
    d = Student()
    e = Student()
    a.say_hello()
    print(Student.quantity)
    print(a.empty_name_check())

main()

```

```
C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab04\Zadanie4.py
Hello! I'm    True
5
True
Process finished with exit code 0
```

Zadanie 5

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

class Item:

    def get_sound(self) -> None:
        print("item's sound")

class Element:

    def get_sound(self) -> None:
        print("element's sound")

class Thing(Item, Element):

    def say_hello(self) -> None:
        print("hello!")

def main() -> None:
    it = Item()
    el = Element()
    th = Thing()
    it.get_sound()
    el.get_sound()
    th.get_sound()

main()
"""Różnica polega na tym, że klasa Thing() będzie dziedziczyła atrybuty od
klasy wymienionej jako pierwsza. """
```

```
C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab04\Zadanie5.py
item's sound
element's sound
item's sound
Process finished with exit code 0
```

Zadanie 6

```
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

class Vehicle:

    def __init__(self, owner: str, table: str):
        self.owner = owner
        self.table = table

    def __repr__(self) -> str:
        return self.owner + " " + self.table

    def __str__(self) -> str:
        return self.owner + " " + self.table

    def __eq__(self, o: object) -> bool: # funkcja do sprawdzania równości
(==)
        return self.owner == o.owner

    def __ne__(self, o: object) -> bool: # funkcja zastępująca !=
        return self.owner != o.owner

    def __lt__(self, o: object) -> bool: # funkcja zastępująca <
        return self.owner < o.owner

    def __gt__(self, o: object) -> bool: # funkcja zastępująca >
        return self.owner > o.owner

    def __le__(self, o: object) -> bool:
        return self.owner <= o.owner

    def __ge__(self, o: object) -> bool:
        return self.owner >= o.owner

    def get_owner(self) -> str:
        return self.owner

    def get_sound(self):
        print("vehicle's brum brum")

class Car(Vehicle):

    def get_sound(self) -> None:
        print("car's brum brum")

def main() -> None:
    vehi = Vehicle("Andrzej", "PY001KY")
    cr = Car("Marcin", "CB001KY")
    print(vehi.__eq__(cr))
    print(vehi.__ne__(cr))
    print(vehi.__lt__(cr))
    print(vehi.__gt__(cr))
```

```
print(vehi.__le__(cr))
print(vehi.__ge__(cr))
```

```
main()
```

```
C:\Users\koku\PycharmProjects\SkrypoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrypoweJęzykiProgramowania\Lab04\Zadanie6.py
False
True
True
False
True
False

Process finished with exit code 0
```

Zadanie 7

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
from random import randint
from typing import List

# stałe i zmienne globalne

# funkcje

class Student:
    def __init__(self, name: str, last_name: str, index: int):
        self.name = name
        self.last_name = last_name
        self.index = index

    def __repr__(self) -> str:
        return self.name + " " + self.last_name

    def __str__(self) -> str:
        return self.last_name + " " + self.name

    def __eq__(self, o: object) -> bool: #funkcja do sprawdzania równości
(==)
        return self.index == o.index

    def __ne__(self, o: object) -> bool: #funkcja zastępująca !=
        return self.index != o.index

    def __lt__(self, o: object) -> bool: #funkcja zastępująca <
        return self.index < o.index

    def __gt__(self, o: object) -> bool: #funkcja zastępująca >
        return self.index > o.index

s1 = Student('Joe', 'Doe', 111111)
print(repr(s1))
s2 = Student('Jane', 'Key', 222222)
```



```

print(str(s2))
if s1 == s2:
    print("objects equal!")
else:
    print("not equal..")

"""Bez przeciążenia funkcja nie działa poprawnie."""

```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab04\Zadanie7.py
Joe Doe
Key Jane
not equal..

Process finished with exit code 0

```

Zadanie 8

```

#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing
from random import randint
from typing import List

# stałe i zmienne globalne

# funkcje

class Student:
    def __init__(self, name: str, last_name: str, index: int):
        self.name = name
        self.last_name = last_name
        self.index = index

    def __repr__(self) -> str:
        return self.name + " " + self.last_name

    def __str__(self) -> str:
        return self.last_name + " " + self.name

    def __eq__(self, o: object) -> bool: #funkcja do sprawdzania równości
(==)
        return self.index == o.index

    def __ne__(self, o: object) -> bool: #funkcja zastępująca !=
        return self.index != o.index

    def __lt__(self, o: object) -> bool: #funkcja zastępująca <
        return self.index < o.index

    def __gt__(self, o: object) -> bool: #funkcja zastępująca >
        return self.index > o.index

s1 = Student('Joe', 'Doe', 111111)

```

```
s2 = Student('Jane', 'Key', 222222)
print(s1.__dict__)

"""Funkcja __dict__ przekształca obiekt klasa na słownik."""
```

```
C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\Lab04\Zadanie8.py
{'name': 'Joe', 'last_name': 'Doe', 'index': 111111}

Process finished with exit code 0
```

Zadanie 9

```
#!C:\Users\koku\AppData\Local\Programs\Python\Python310

# importy
import typing

# stałe i zmienne globalne

# funkcje

class Vehicle:

    def __init__(self, owner: str, table: str):
        print("Vehicle constructor")
        self.owner = owner
        self.table = table

    def get_owner(self) -> str:
        return self.owner

    def get_sound(self):
        print("vehicle's brum brum")

class Car(Vehicle):

    def __init__(self, owner: str, table: str):
        print("Car constructor")
        self.table = table
        super().__init__(owner, table)

    def get_sound(self) -> None:
        print("car's brum brum")

def main() -> None:
    cr = Car("Marcin", "CB001KY")

main()

"""W klasie dziedziczącej, najpierw zostaje wywołany konstruktor własny, a
potem konstruktor z klasy wyższej."""
```

```
C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts\python.exe C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\Lab04\Zadanie9.py
Car constructor
Vehicle constructor

Process finished with exit code 0
```

Wnioski

Zadania były dla mnie zrozumiałe; pozwoliły mi zaznajomić się z działaniami na klasach w języku Python.