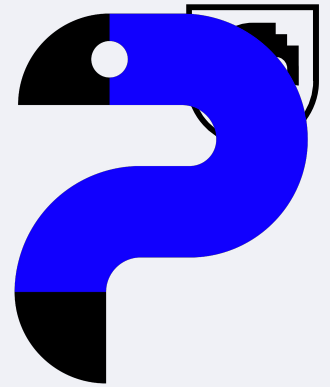




**Code
Academy**



1 LYGIS

11 paskaita. Get Set metodai, debug



Šiandien išmoksite

01

Kas yra Get ir Set metodai

02

Debugint programą



```
class Darbuotojas:
    def __init__(self, vardas, pavarde, atlyginimas):
        self.vardas = vardas
        self.pavarde = pavarde
        self.atlyginimas = atlyginimas
```

```
domas = Darbuotojas("Domas", "Rutkauskas", 1200)
domas.atlyginimas = 1500
print(domas.atlyginimas)
```

```
domas = Darbuotojas("Domas", "Rutkauskas", 1200)
domas.atlyginimas = -150
print(domas.atlyginimas)
```

Kada gali prireikti get/set metodų



```
class Darbuotojas:
    def __init__(self, vardas, pavarde, atlyginimas):
        self.vardas = vardas
        self.pavarde = pavarde
        self.__atlyginimas = atlyginimas

    def set_atlyginimas(self, naujas):
        if naujas < 0:
            print("Atlyginimas negali būti neigiamas")
        else:
            self.__atlyginimas = naujas

domas = Darbuotojas("Domas", "Rutkauskas", 1200)
domas.set_atlyginimas(-1200)

# Atlyginimas negali būti neigiamas

print(domas.atlyginimas)

# AttributeError: 'Darbuotojas' object has no attribute 'atlyginimas'
```

Sprendimo variantas 1



Sprendimo variantas 2

```
class Darbuotojas:
    def __init__(self, vardas, pavarde, atlyginimas):
        self.vardas = vardas
        self.pavarde = pavarde
        self.__atlyginimas = atlyginimas

    def set_atlyginimas(self, naujas):
        if naujas < 0:
            print("Atlyginimas negali būti neigiamas")
        else:
            self.__atlyginimas = naujas

    def get_atlyginimas(self):
        return self.__atlyginimas

domas = Darbuotojas("Domas", "Rutkauskas", 1200)
print(domas.get_atlyginimas())

# 1200

domas.atlyginimas = 500
print(domas.get_atlyginimas())

# 1200
```



```
class Darbuotojas:
    def __init__(self, vardas, pavarde, atlyginimas):
        self.vardas = vardas
        self.pavarde = pavarde
        self.__atlyginimas = atlyginimas

    @property
    def atlyginimas(self):
        return self.__atlyginimas

    @atlyginimas.setter
    def atlyginimas(self, naujas):
        if naujas < 0:
            print("Atlyginimas negali būti neigiamas")
        else:
            self.__atlyginimas = naujas

domas = Darbuotojas("Domas", "Rutkauskas", 1200)
print(domas.atlyginimas)
# 1200

domas.atlyginimas = 1500
print(domas.atlyginimas)
# 1500

domas.atlyginimas = -150
# Atlyginimas negali būti neigiamas

print(domas.atlyginimas)
# 1500
```

Dekinatorius @Property



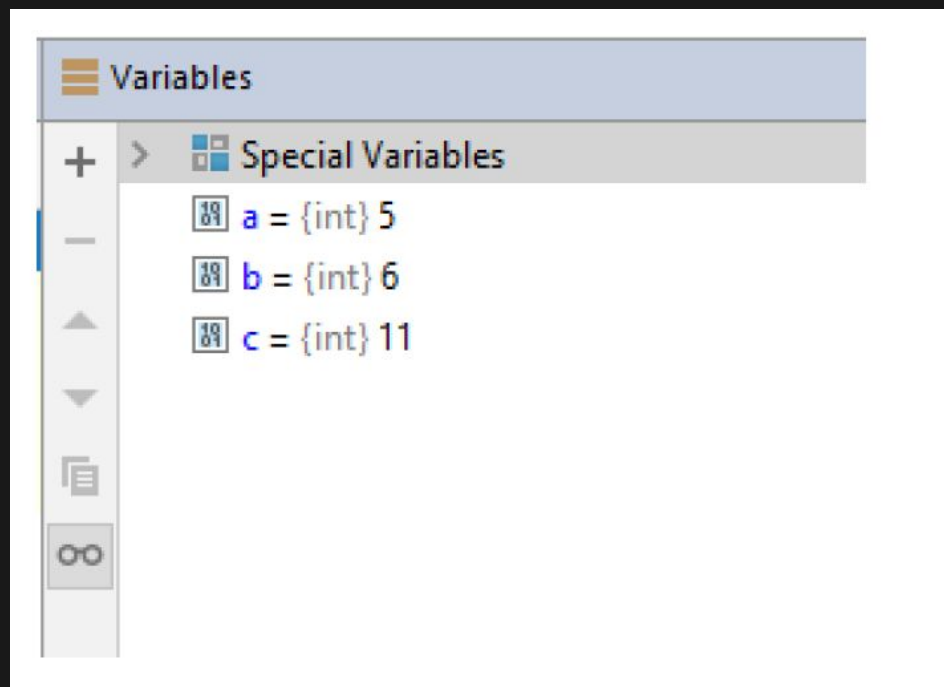
```
1 a = 5
2 b = 6
3 c = a + b
4
```

```
1 a = 5 a: 5
2 b = 6 b: 6
3 c = a + b c: 11
4
5 pass
```

Klaidų taisymas (Debug)



Kintamųjų langas



11 paskaita. Get Set metodai, debug

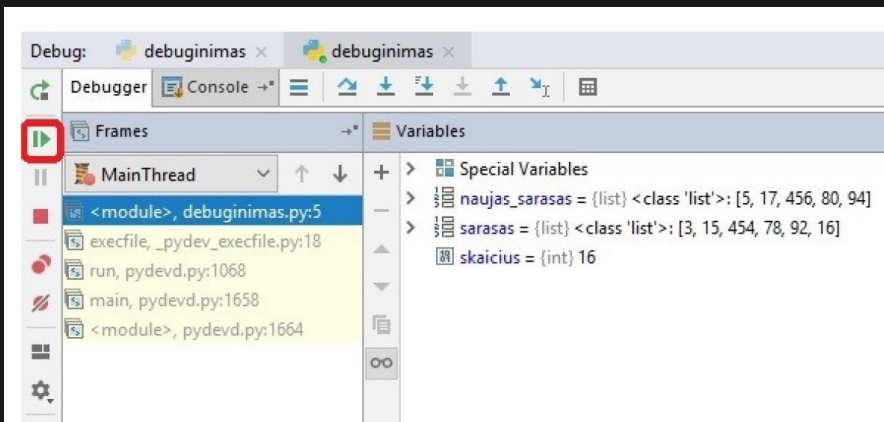


```
sarasas = [3, 15, 454, 78, 92, 16]
naujas_sarasas = []

for skaicius in sarasas:
    naujas_sarasas.append(skaicius + 2)
```

```
1 sarasas = [3, 15, 454, 78, 92, 16] sarasas: <class 'list': [3, 15, 454,
2 naujas_sarasas = [] naujas_sarasas: <class 'list': [5, 17, 456, 80, 94]
3
4 for skaicius in sarasas: skaicius: 16
5 naujas_sarasas.append(skaicius + 2)
```

Sąrašo debuginimas



11 paskaita. Get Set metodai, debug

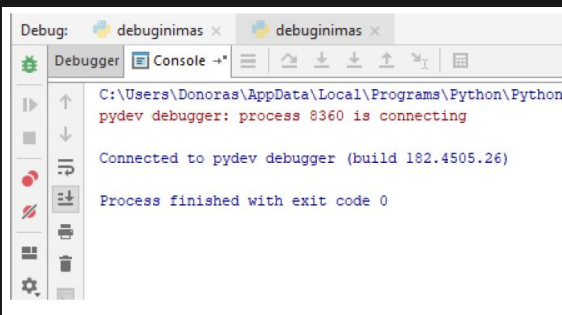


Kaip patikrinti, ar kodas veikia:

```
def patikrinti_skaiciu(skaicius):  
    if skaicius < 0:  
        print("Skaičius teigiamas")  
    if skaicius == 0:  
        print("Skaičius lygus 0")  
    if skaicius < 0:  
        print("Skaičius neigiamas")  
  
patikrinti_skaiciu(20)
```

Nieko nespausdina!

```
1 def patikrinti_skaiciu(skaicius):  
2     if skaicius < 0:  
3         print("Skaičius teigiamas")  
4     if skaicius == 0:  
5         print("Skaičius lygus 0")  
6     if skaicius < 0:  
7         print("Skaičius neigiamas")  
8  
9 patikrinti_skaiciu(20)
```



Kaip patikrinti ar kodas veikia 1



```
def patikrinti_skaiciu(skaicius):  
    if skaicius > 0:  
        print("Skaičius teigiamas")  
    if skaicius == 0:  
        print("Skaičius lygus 0")  
    if skaicius < 0:  
        print("Skaičius neigiamas")
```

```
patikrinti_skaiciu(20)  
    Skaičius teigiamas
```

Kaip patikrinti ar kodas veikia 2

11 paskaita. Get Set metodai, debug



```
1 class Automobilis():
2     def __init__(self, modelis, metai):
3         self.__modelis = modelis
4         self.__metai = metai
5
6     toyota = Automobilis("Toyota Auris", 2015)  toyota: <
7     tesla = Automobilis("Tesla Y", 2020)
```

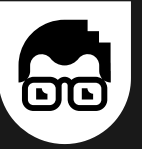
Variables

+ > Special Variables

▼ **toyota** = {Automobilis} <_main_.Automobilis object at 0x0000023841A36780>

- 181 _Automobilis__metai = {int} 2015
- 181 _Automobilis__modelis = {str} 'Toyota Auris'

Kaip patikrinti objektus



Užduotis nr. 1

Parašyti klasę "Namas", kuri turėtų savybę "plotas" ir "verte". Padaryti taip, kad savybė "verte" koreguojama tik įvedus skaičių. Programoje naudoti dekoratorių `@property`.



Namų darbas

Užbaigti klasėje nepadarytas užduotis