

1. Täydennä seuraava koodi toimivaksi. Kirjoita koodia vain harmaisiin kohtiin. Hyödynnä periytymissuhdetta eli älä kirjoita samaa koodia useaan kertaan. Ohjelman tulee toimia samoin kuin esimerkkitulosteessa.

```
class Person:
    def __init__(self, first_name, last_name):
```

```
        self.first_name = first_name;
        self.last_name = last_name;
```

```
    def plot(self):
```

```
        print(f"-- My name is {self.first_name} {self.last_name}");
        return;
```

```
class Student(Person):
    def __init__(self, first_name, last_name, student_nr):
        super().__init__(first_name, last_name);
        self.student_nr = student_nr;

    def plot(self):
        super().plot();
        print(f"and my student number is {self.student_nr}")
```

```
# pääohjelma
```

```
p1 = Person("James", "Bond");
s1 = Student("Johnny", "English", 321);
p1.plot();
s1.plot();
```

Ohjelman tuottama tuloste:

```
-- My name is James Bond
-- My name is Johnny English
and my student number is 321
```

2. Määrittele muutamalla lauseella seuraavat olio-ohjelmoinnin käsitteet

a) luokka / class

- luokan avulla voidaan luoda olioita
- määrittelee olioiden yhteiset ominaisuudet ja toiminnot
- 'sapluuna' olioiden luomiseksi

b) olio / object

- luokan konkreettinen ilmentymä eli instanssi
- sisältää luokan kaikki ominaisuudet
- osaa kaikki luokassa määritellyt toiminnot

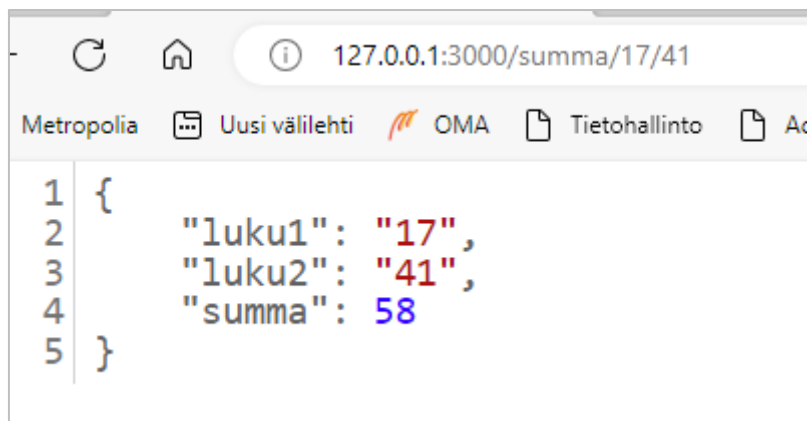
c) periytyminen

- luokasta voidaan periyttää uusia aliluokkia
- aliluokat omaavat automaattisesti ylläluokan kaikki ominaisuudet ja toiminnot
- aliluokat voivat sisältää uusia ominaisuuksia tai toimintoja

d) assosiaatio

- kahden luokan välinen yhteys/ suhde
- suhde voi olla pysyvä tai tilapäinen.
- pysyvässä suhteessa luokka tietää/ tuntee toisen luokan koko ajan

3. Alla on kuva selaimesta lähetetystä API-kutsusta ja siihen saadusta vastauksesta. Täydennä ohjelmassa olevat varjostetut kohdat siten, että API tuottaa selaimen vastaavat tiedot kuin kuvassa. Virheisiin ei tarvitse varautua. (max. 8 p.)



```
from flask import Flask, Response
```

```
app = Flask(__name__)
@app.route('/summa/<luku1>/<luku2>')
def kaiku(luku1, luku2):
```

```
    summa = float(luku1) + float(luku2)
```

```
    vastaus = {
        "luku1": luku1,
        "luku2": luku2,
        "summa": summa
    }
```

```
    return vastaus
```

```
if __name__ == '__main__':
    app.run(use_reloader=True, host='127.0.0.1', port=3000)
```