# BME 646/ ECE695DL: Homework 1

Chengjun Guo

01.17.2022

## 1 Introduction

This project needs two classes one called geocountry derived from country. Country class has two instance variables called capital and population. It has a init function and a net population which returns the current net. The extended class geocountry has two more instance variables called area and density. It has three more functions including two density calculators and a net density. The net population in country class is overwritten by geocountry.

## 2 Methodology

### 2.1 OOP

In this project, I used Object Oriented Programming skill.

## 3 Implementation and Results

```
class Countries():

    def __init__(self, capital, population):
        self.capital = capital
        self.population = population #birth,death,last_count

    def net_population(self):
        current_net = self.population[0] - self.population[1] + self.population[2]
        return current_net

class GeoCountry(Countries):

    def __init__(self,capital, population, area):
        super(GeoCountry,self).__init__(capital, population)
        self.area = area
        self.density = 0

    def density_calculator1(self):
        if len(self.population) == 3:
            self.density = (self.population[0] - self.population[1] + self.population[2]) / self.area
        if len(self.population) == 4:
            self.density = (self.population[0] - self.population[1] + (self.population[2] + self.population[3]) / 2) / self.area

    def density_calculator2(self):
        if len(self.population) == 3:
            self.population[2] = self.population[2] - self.population[0] + self.population[1]
            self.density = (self.population[0] - self.population[1] + self.population[2]) / self.area
        if len(self.population) == 4:
            self.population[3] = (self.population[3] - self.population[0] + self.population[1]) * 2 - self.population[2]
            self.density = (self.population[0] - self.population[1] + (self.population[2] + self.population[3]) / 2) / self.area


    def net_density(self,choice):
        if choice == 1:
            return self.density_calculator1
        if choice == 2:
            return self.density_calculator2

    def net_population(self):
        if len(self.population) == 3:
            x = self.population[0] - self.population[1] + self.population[2]
            self.population.append(x)
            return self.population[0] - self.population[1] + (self.population[2] + self.population[3]) / 2
        if len(self.population) == 4:
            self.population[3] = self.population[0] - self.population[1] + (self.population[2] + self.population[3]) / 2
            return self.population[3]

if __name__ == "__main__":
    task2 = Countries("Piplipol", [40,30,20])
    task5 = GeoCountry("Polpip", [55,10,70], 230)

    #test
#    ob1 = GeoCountry(’YYY’, [20,100, 1000],5)
#    print(ob1.density)#0
#    print(ob1.population)#[20,100,1000]
#    ob1.density_calculator1()
#    print(ob1.density)#184.0
#    ob1.density_calculator2()
#    print(ob1.population)#[20, 100, 1080]
#    print(ob1.density)#200.0
#    ob2 = GeoCountry(’ZZZ’, [20, 50, 100], 12)
#    fun = ob2.net_density(2)
#    print(ob2.density)#0
#    fun()
#    print("{:.2f}".format(ob2.density))#8.33
#    print(ob1.population)#[20,100, 1080]
#    print(ob1.net_population())#960.0
#    print(ob1.population)#[20,100,1080,1000]
#    print(ob1.density)#200.0 (the value of density still uses the previous value of population population)
#
#    ob1.density_calculator1()
#    print(ob1.population)#[20, 100, 1080, 1000]
#    print(ob1.density)#192.0
#    ob1.density_calculator2()
#    print(ob1.population)#[20, 100, 1080, 1080]
#    print(ob1.density)#200
```

Listing 1: My Code

## 4 Lessons Learned

The hurdle that I faced would be taking short cut in the density calculator. I used net population function to avoid typing the formula in the density calculator. However, this would cause error with the task moving on. It takes a while to debug it.

## 5 Suggested Enhancements

Currently I don't have much comments. I only commented the population in country class in case I forget the variables in list. The enhancement I would take is adding more comments to make the code more readable.