**Exercise: 3**

**Name: Pekka Lehtola**

**How many tasks did you do: 9**

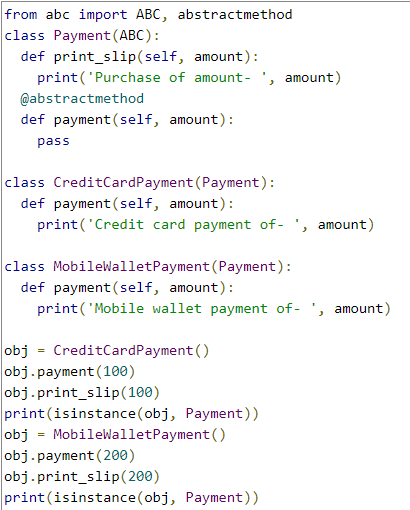
**Were the tasks easy, ok, difficult: The tasks were challenging this time, especially 3.7 produced problems.**

**Do you need help/comments in any task (if yes, to which ones):**

**No, but go easy with 3.7, I tried to do it for several hours and had to settle for an easier way because I didn’t get it to work with another class at all.**

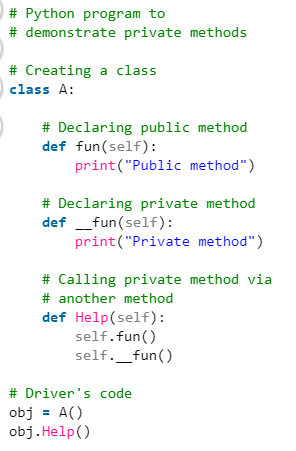
1. Explain the following terms:

a. Abstraction (in programming)

* In object-oriented programming, abstraction is one of three central principles (along with encapsulation and inheritance). Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.
* What I read online about abstraction its like creating a private class that cant have objects, but with it other classes can be created.
* In the code ABC (Abstract Base Classes) is used to crate abstract class.

b. Accessor and mutator methods

* Accessor and mutator are basically get and set methods used in python.
* Accessor(get) returns copy of a private variable.
* Mutator(set) is used to modify private variables.



c. Public and private methods

* Private methods are basically the same as private attributes, they cant be called directly.
* They can be called with other classes methods or with the help of name mangling

d. \_\_str\_\_ method (in Python)

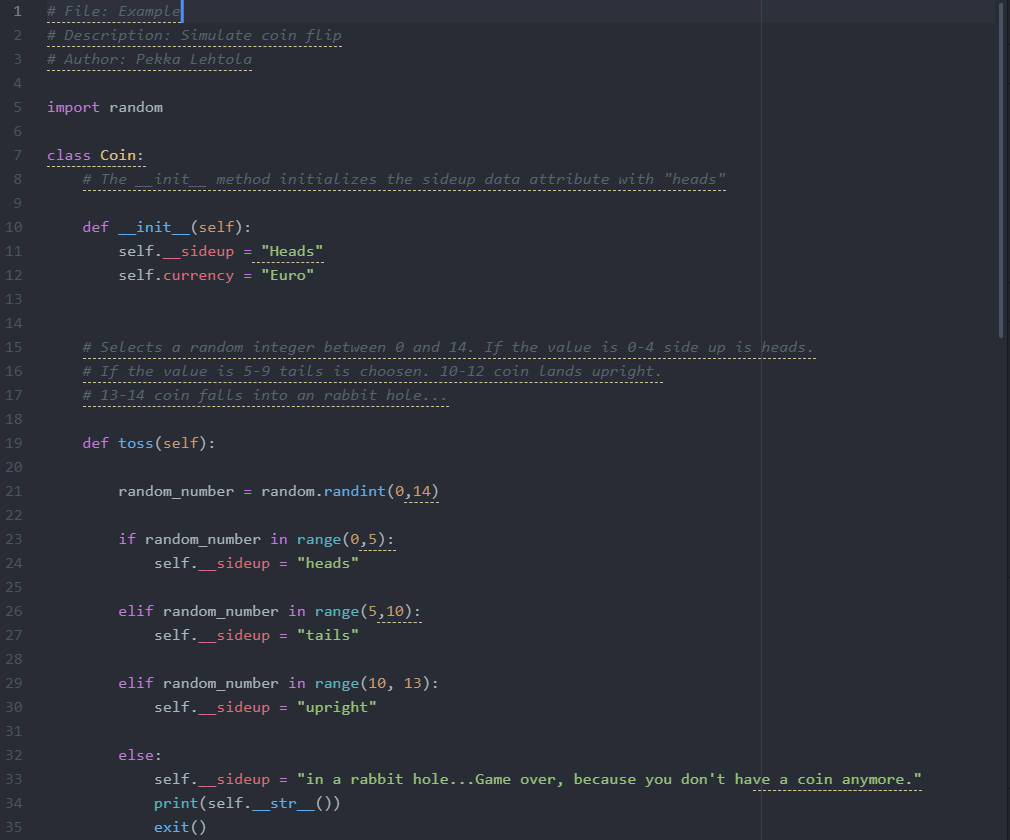
* Used for defining output print of an object. With out it if you try to print an object, output would look like ( <\_\_main\_\_.Test object at 0x00000286441D0A48> )

2. Modify the Coin class (see Exercise2) so that in addition to sideup you have another data attribute called currency. Add a function generating the currency (Euro, Pound, Dollar, Ruble, Yen). Use a random generator to get the currency (=similar to tossing the coin). Add a function to print out the currency.

3. Add a method that can change the currency of the coin. Test that your coin still works.

4. Change the Coin’s sideup attribute to private. Test that your coin still works. What happens, if you now try to change the attribute’s value from the main function? Try it out...!

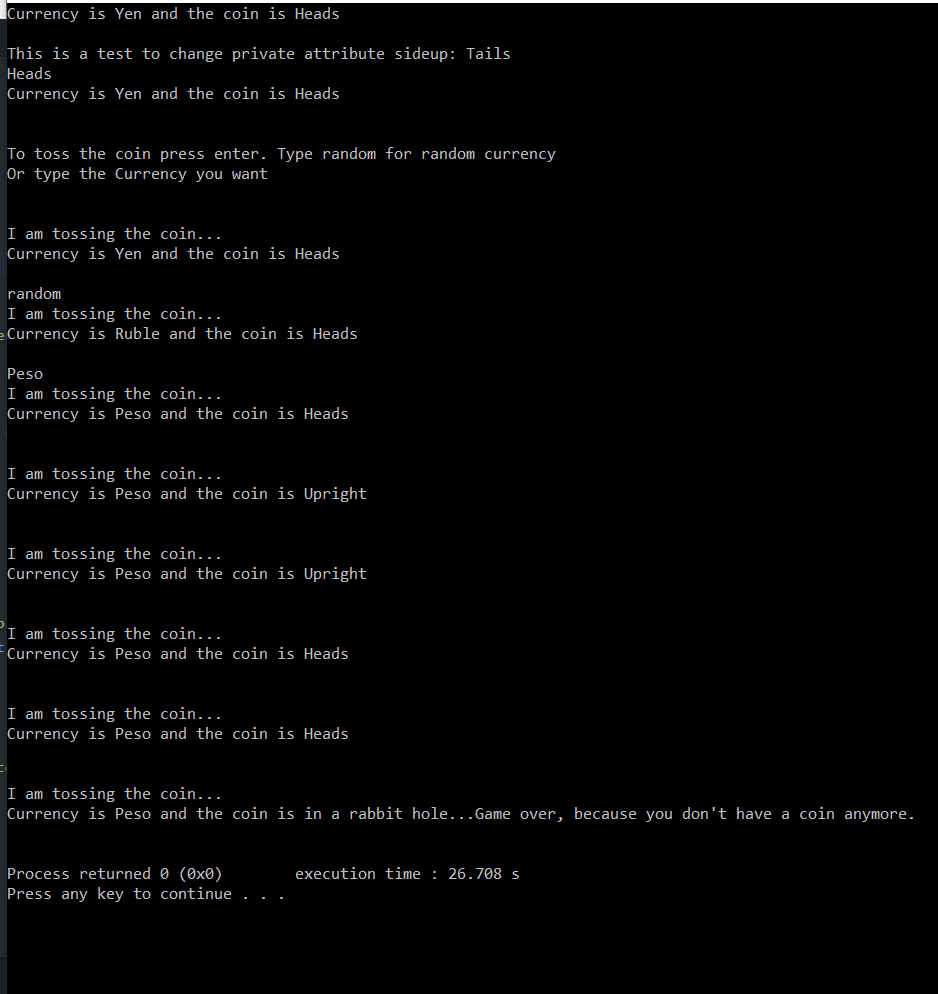
Screen capture of Task 2/3/4







Screen capture of the output of Task **2/3/4**



Tämä oli yhdistelmä tehtävistä 2,3,4

Alussa yritys vaihtaa yksityisen atribuutin arvoksi Tails, joka ei toiminut.

5. Create a class Dice and make an object of it. You shall be able to roll the dice, get the result (number between 1 – 6) and get its color. Add at least 1 extra feature. Design your program using pseudocode. Document your code properly (with good comments) and pay attention to the clarity of the output prints.

Program starts

Initialize Dice:

Color = White

Side up = 6

Sum of throws = 0

Defining dice roll:

random number = random number between 1-6

color list = (1 = Red, 2 = Blue, 3 = Green, 4 = Yellow, 5 = Black, 6 = White)

Sum of throws = Sum of throws + random number

Dices side up = random number

Dices color = color list( random number )

Defining get side up:

print Dices side up

print sides color

print sum of throws

Defining restart game:

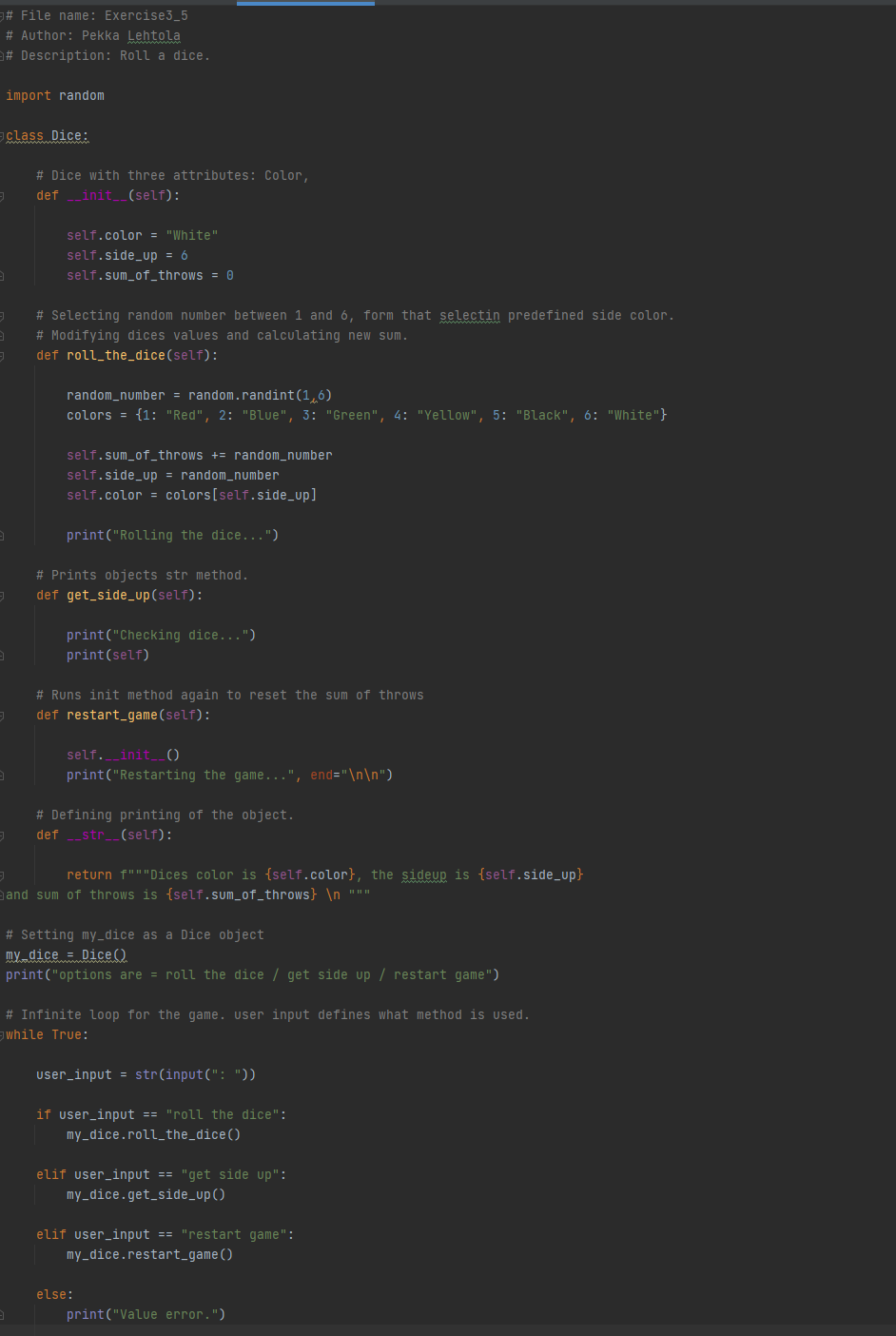
Initialize Dice again

Infinite Loop:

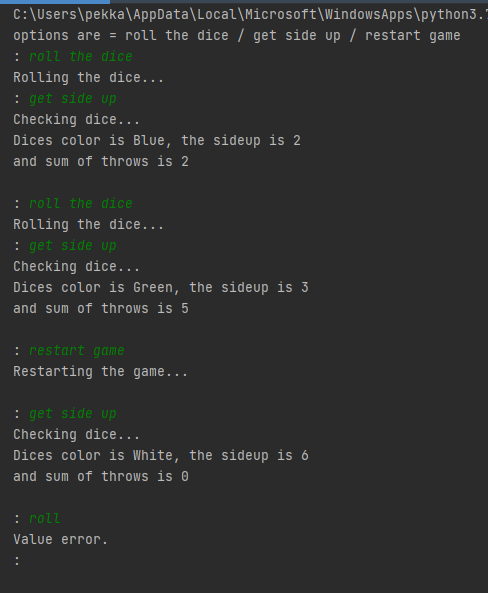
take user input

from that user input run:

dice roll , get side up or restart game

Screen capture of Task 5

Screen capture of the output of Task 5

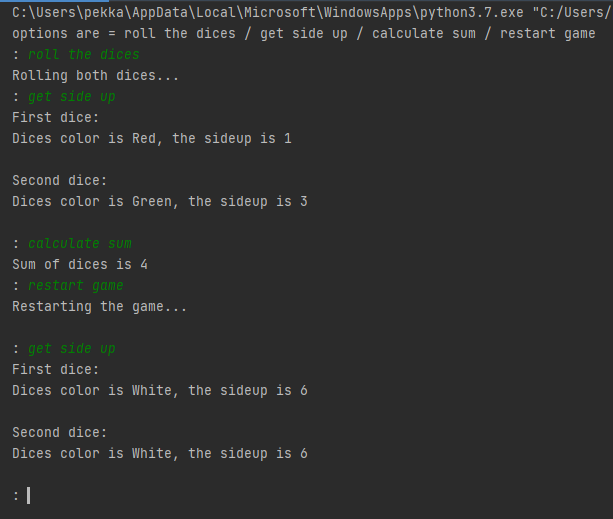
****

6. Create two Dice objects and roll them both. Sum the result and print to screen.

Screen capture

of Task 6

Screen capture of the output of Task 6



7. Design first using pseudocode, then code this: Create a Dice rolling game of three players (three Dice objects). On first round everybody rolls their dice, lowest number loses and is out of game. On second round the two remaining contestants roll a dice and higher number wins. Use proper output prints of the situation all the time. If on either round there is a tie between 2 or 3 dices, then the tied dices are rolled again.

Program start

Initialize Dice:

Side up = 1

Name = user input

Defining dice roll:

if player side up = 7:

Skip

else:

Side up = random number between 1 – 6

Defining check duplicates:

if all players have the same side\_up number:

player\_1 dice roll

player\_2 dice roll

player\_3 dice roll

else If player\_1 and player\_2 has the same side\_up number but not 7:

player\_1 dice roll

player\_2 dice roll

else If player\_1 and player\_3 has the same side\_up number but not 7:

player\_1 dice roll

player\_3 dice roll

else If player\_2 and player\_3 has the same side\_up number but not 7:

player\_2 dice roll

player\_3 dice roll

Defining check winner:

create list containing players dice\_up numbers

remove from list players that has dice\_up value of 7

If list lenght = 1:

Declare remaining player as winner.

Exit code

else:

select the player with the smalles dice\_up value from the list

set dice\_up value as 7

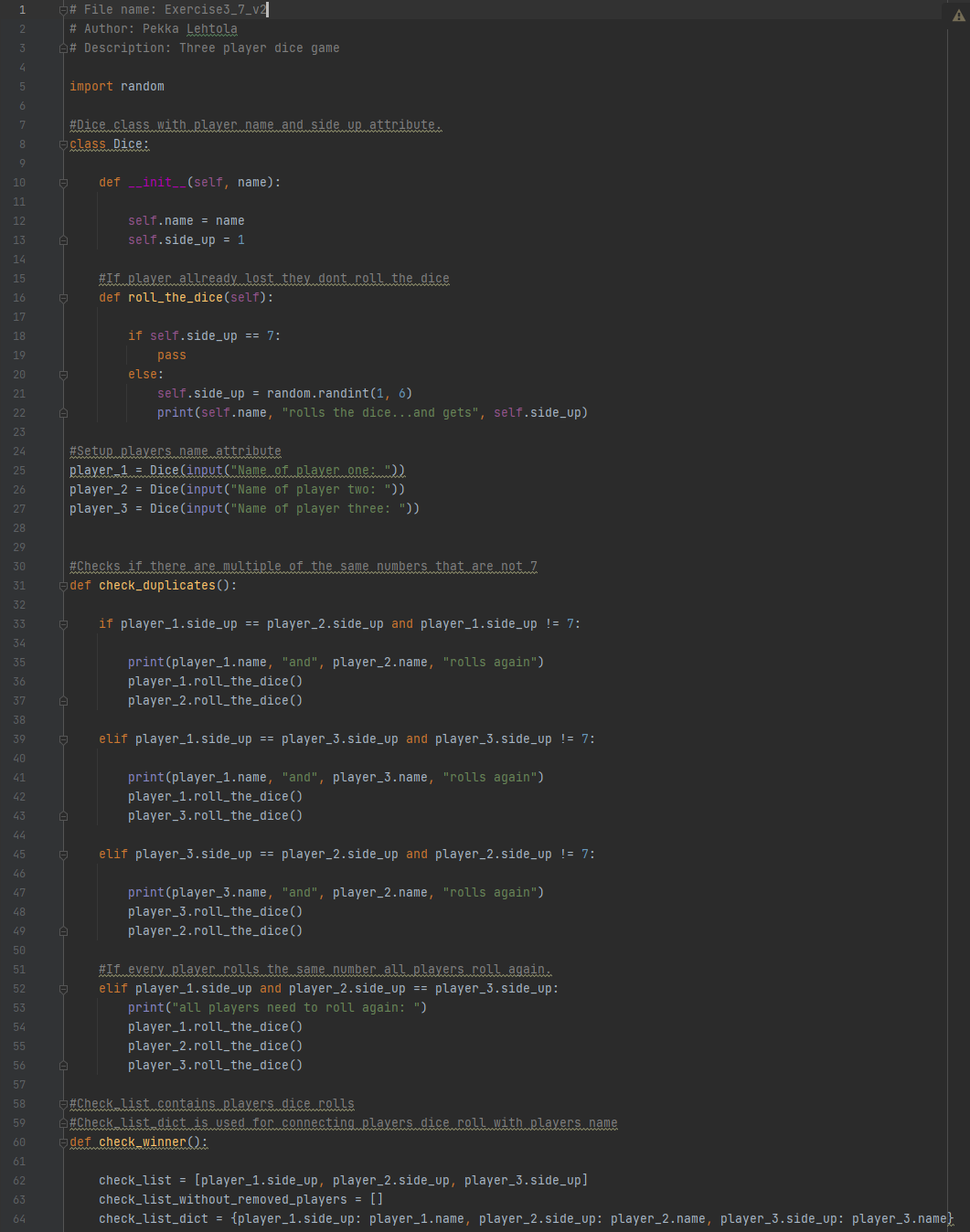
Defining main:

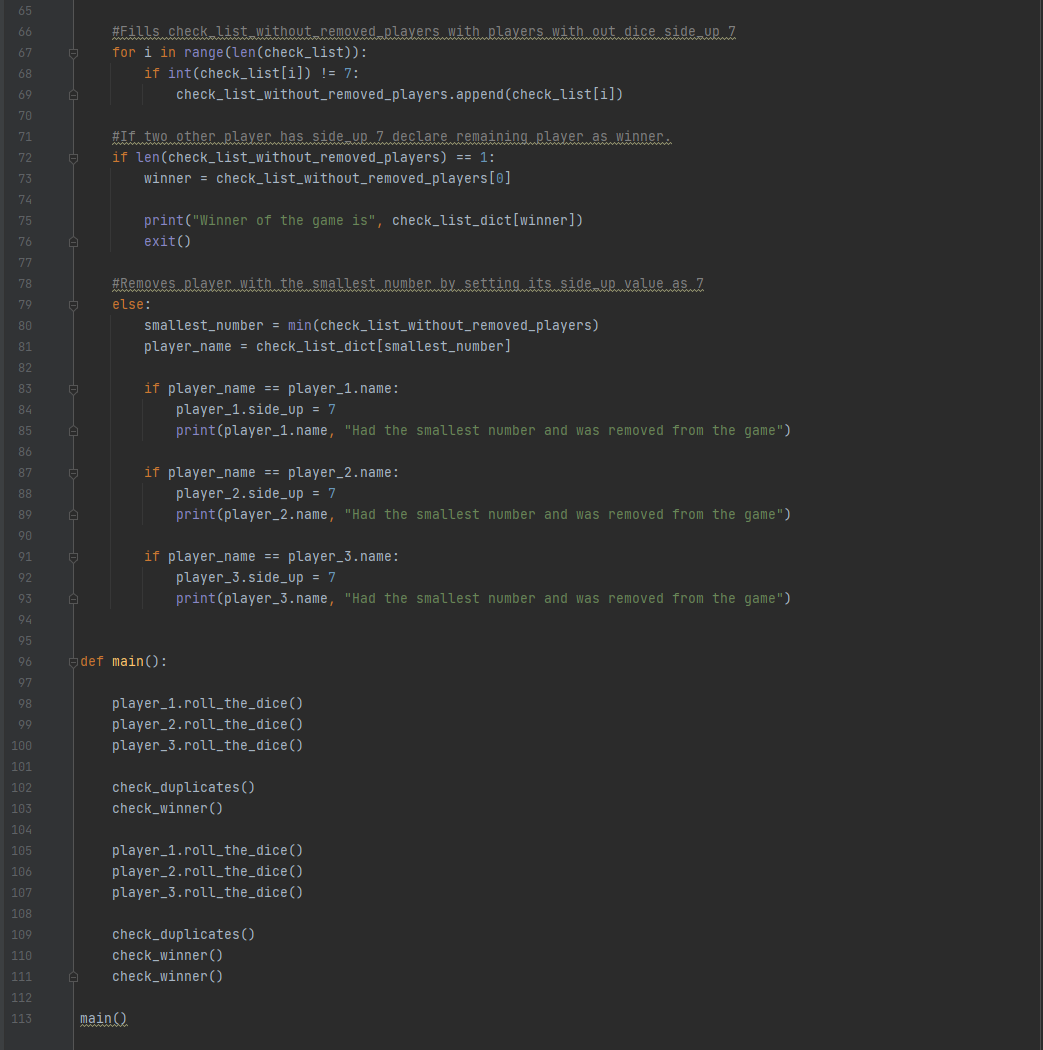
Infinite loop:

All players roll the dice

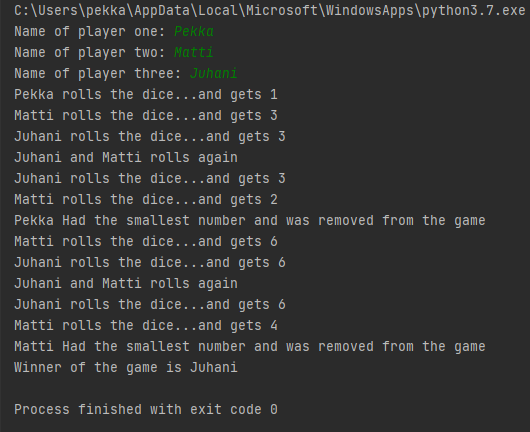
check for duplicates

check winner

Screen capture of Task 7



Screen capture of the output of Task 7



Design first using pseudocode, then code this: Create a CellPhone Class. Write a program that will design a class that represents a cell phone. The data attributes are manufact (Manufacter), model (Model) and retailPrice (Retail price). The class will also have the following methods:

a. \_\_init\_\_

b. set Manufact

c. set Model

d. setRetailPrice

e. getManufact

f. getModel

g. getRetailPrice

Program start

Defining Cellphone:

Defining initializing:

manufacturer = “ “

model = “ “

retail price = 0

Defining set manufact:

Cellphone manufacturer = user input

Defining set model:

Cellphone model = user input

Defining set retail price:

Cellphone retail price = user input

Defining get manufacturer:

return “Manufacturer: “ + Cellphone manufacturer

Defining get model:

return “Model number: “ + Cellphone model number

Defining get retail price:

return “Retail price: “ + Cellphone retail price

Defining main:

my cellphone = Cellphone

my cellphone set manufact

my cellphone set model

my cellphone set retail price

print “ Here is the data that you provided : “

my cellphone get manufacturer

my cellphone get model:

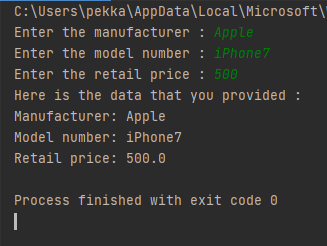
my cellphone get retail price:

Start main

Screen capture of Task 8



Screen capture of the output of Task 8



9. Take a look at the CellPhone Class/Object: where are these concepts (or are they there) (take a screen capture and indicate a line)?

a. Object?

b. Encapsulation?

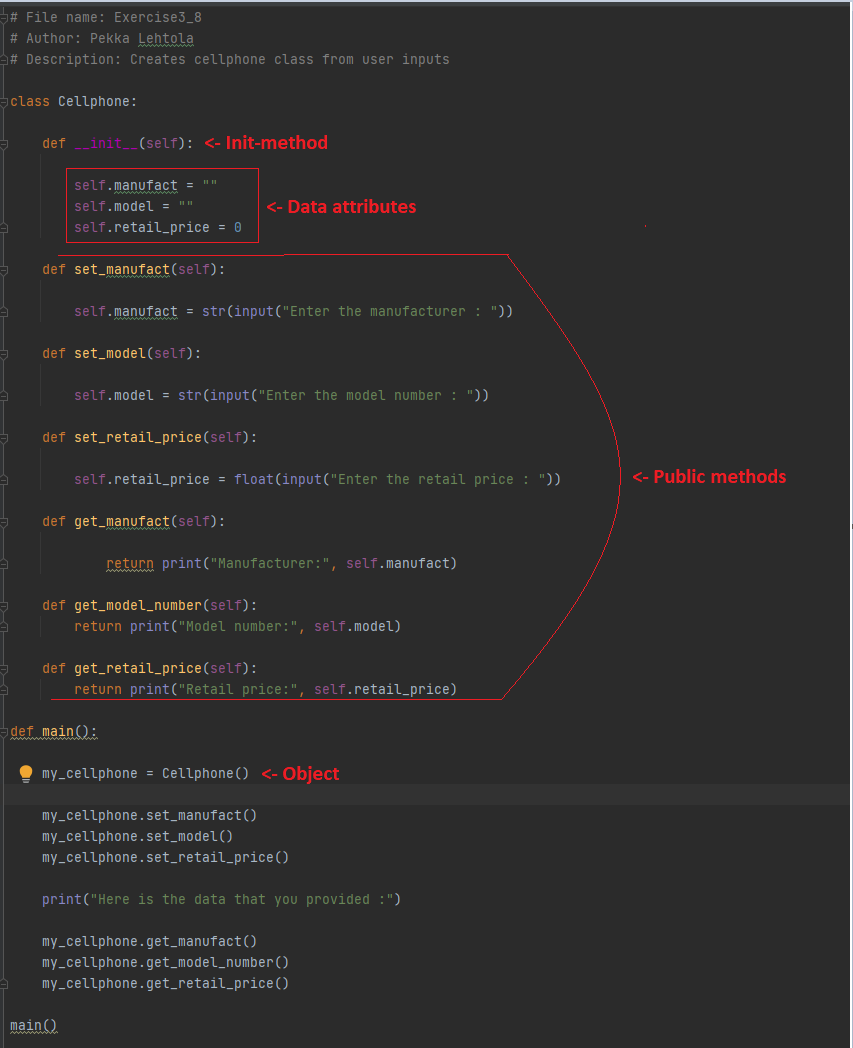
c. Data attributes?

d. Hidden attributes?

e. Public methods?

f. Private methods?

g. Init-method?

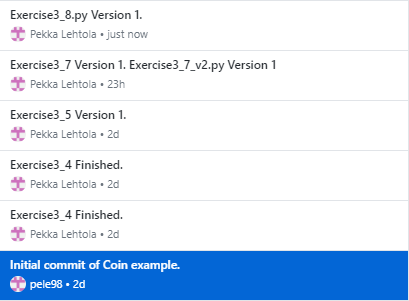


No Encapsulation in this code, that would require private data attributes like \_model or \_\_manufact

No Hidden attributes

No private methods

Screen capture of git log (showing that you made a commit after every task).



Huom. Tehtävät 2-4 samassa koska oli teknisiä ongelmia…

Self-assessment:

This exercise was easy/difficult/ok/etc. for me because…

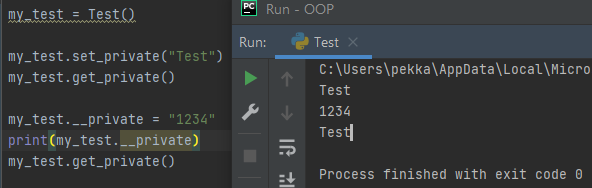
Näiden tehtävien kanssa jostain syystä koin todellisia tuskia, varsinkin tehtävässä 7, jotenkin en saanut päätäni toimimaan kun yritin toteuttaa usealla luokalla.

Doing this exercise, I learned…

Edellisissä kotitehtävissä miettisin \_\_str\_\_ metodin tarkoitusta ja näiden tehtävien jälkeen tarkoitus on selkeä.

I am still wondering…

En ihan ymmärrä mitä tässä tapahtuu…



I understood/did not understand that… ; I did/did not know that… ; I did/did not manage to do…

Tosiaan subclassit jäi laajankin kokeilujen jälkeen mysteriksi.