PODSTAWY PROGRAMOWANIA W PYTHON

PO 12 ZAJĘCIACH:

1. Omawiane zagadnienia:
    a. pola i metody pseudo-prywatne – zmienne i metody klasy możemy ukrywać przed dostępem poza klasą. W tym celu nazwy metod i zmiennych zaczynamy od dwóch podkreślników, np.: **__imie**, **__sprawdz_numer()** – gdy tak zrobimy to Python zmieni i ukryje te nazwy – IDE nie podpowie nam ich. Pamiętajmy, że to jest tylko ukrycie – w łatwy sposób możemy sprawdzić jak te nazwy są zmienione – w tym celu możemy wyświetlić namespace (przestrzeń nazw – czyli wszystkie atrybuty do których instancja ma dostęp) – np.: **pracownik.__dict__**
    Python jest językiem otwartym, jak widzimy nawet pseudoprywatność nie ochroni naszych zmiennych – nie musimy się tym jednak przejmować – dobrą praktyką jest zawsze korzystać z udostępnionych przez programistę metod i właściwości do manipulowania obiektami.
    b. właściwości – properties – są to metody, które definiujemy w celu kontrolowania dostępu do zmiennych – metody te „udają" zmienne – korzystamy z nich bez nawiasów.
        i. getter – dekorator: **@property** – służy do zwracania wartości, kontrolujemy co i jak użytkownik naszego kodu może odczytać
        ii. seter – dekorator **@nazwa_property.setter** – aby utworzyć seter musimy najpierw zdefiniować getter. Settery służą do walidacji i odpowiedniej obróbki przekazanych przez nie wartości, zanim je zapiszemy do zmiennej
        iii. deleter - **@nazwa_property.deleter** – służy do wyczyszczenia w zdefiniowany przez nas sposób property.
2. **testy jednostkowe** – https://docs.python.org/3/library/unittest.html

a. moduł **unittest** – pozwala na tworzenie testów naszego kodu. Zobaczyliśmy, że niektóre sposoby implementowania funkcjonalności nastręczają trudności w testowaniu – patrz atrybuty prywatne, funkcje używające print()
b. stworzyliśmy helper – dzięki któremu przekierowaliśmy print() do bufora w pamięci, a następnie jego wartość zwracaliśmy – dzięki temu możemy w łatwy sposób przechwytywać print().
c. funkcje możemy przypisywać do zmiennych, a następnie te zmienne wywoływać:

```
x = funkcja()     # -> użycie z nawiasami wykona funkcję
x = funkcja       # -> użycie bez nawiasów - przypisuje funkcję do zmiennej
x()               # -> możemy wywołać x jak funkcję - z nawiasami !!!
```

3. Filmiki do obejrzenia:
   a. klasy w Python: https://www.youtube.com/playlist?list=PL-osiE80TeTsqhIuOqKhwlXsIBIdSeYtc
   b. testy: https://www.youtube.com/watch?v=nmBbR97Vsv8
4. **Zadanie domowe dla tych czujących się pewniej** – umiemy już pisać kod obiektowy, zachowując przy tym hermetyczność (in. enkapsulację) – czyli teraz nasza baza będzie odporna na nieuprawnione zmiany – dajemy możliwość manipulowania danymi we wskazany przez nas sposób. Wszystkie metody i klasy są opisane, potrafimy walidować dane. Możemy wprowadzić pola klasy i metody klasy, które będą zmieniać informacje wspólne dla wszystkich instancji.