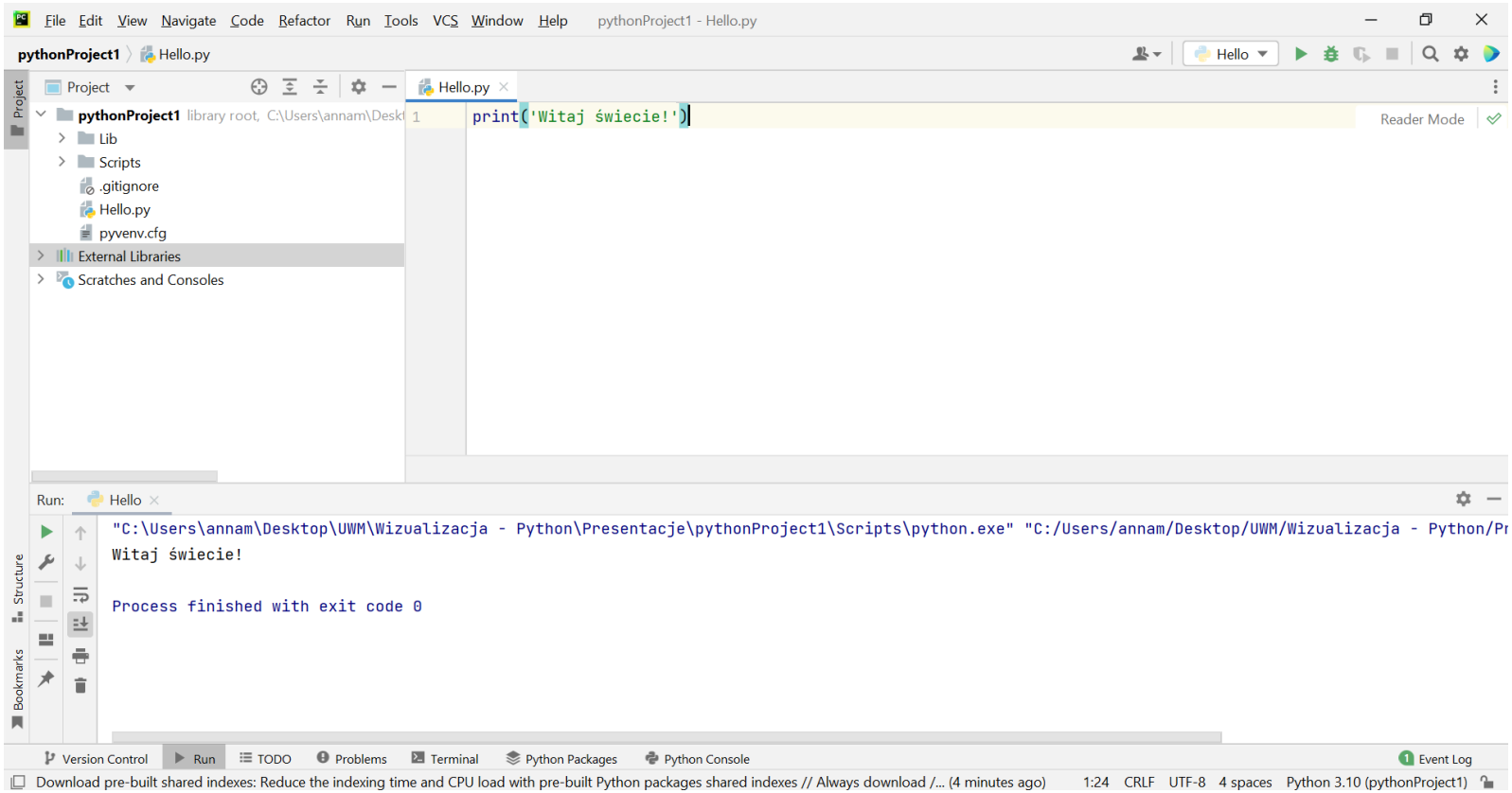
Wizualizacja danych w Python, semestr letni 2022

Anna Muranova

Ćwiczenie 1.

Najprostszy program

Stwórz project `pythonProject1` na dysku D w folderze o numerze twojego indeksu Stwórz w tym projekcie program `Hello.py` wyświetlający na konsoli napis „Witaj świecie!”



Zmień program tak, żeby tam była funkcja `hello_world`.

`https://www.learnpython.org/pl/Funkcje`

`https://www.w3schools.com/python/python_functions.asp`

Konsolę Pythona i najprostszy operacje

Uruchom konsolę Pythona w Pycharm. Poćwicz deklarację zmiennych i podstawowe operacje (arytmetyczne, porównania ith).

- Dodawanie, odejmowanie:
  $2 + 3$, $8 - 3$, itd.
- Dzielenie:
  %, /, //
- Iloczyn:
  $5 * 3$
- Porównanie:
  $>$, $<$, $<=$, $>=$, $!=$, $==$
- Potęgowanie:
  $3 * *5$, $9 * *(1/2)$

Spróbuj:

```
> > import Hello
> > Hello.hello_world()
```

PEP8 – zbiór „reguł”, jak pisać kody, aby były jak najbardziej czytelne.

Poczytaj o PEP8

- `https://kamil.kwapisz.pl/gramatyka-dla-programisty/`
- `https://analityk.edu.pl/jak-pisac-kod-w-python-aby-byl-czytelny-pep8/`

Pełny opis:

- `https://www.python.org/dev/peps/pep-0008/`
- `https://legacy.python.org/dev/peps/pep-0008/`

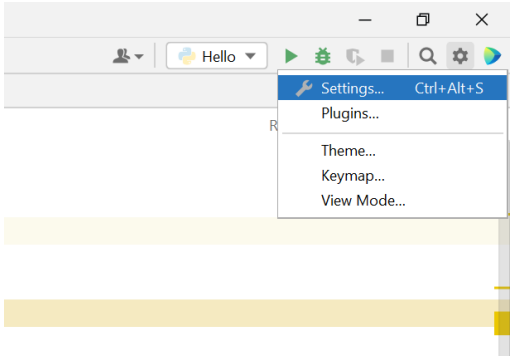
Dodaj do `Hello.py` poniższy kod:

```
a=2.5
b=3
a*b
```

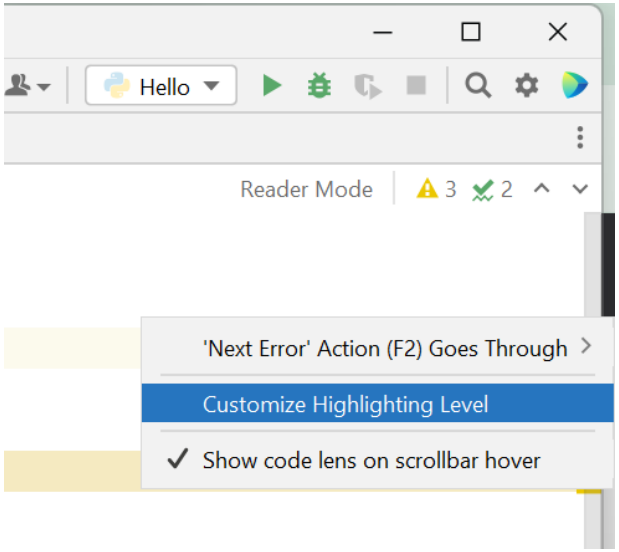
- Jak wyświetlić wynik dziania a*b?
- Spróbuj 'Run File in Python konsole'. Wyświetl na konsole sumą a i b.
- Dodaj do pliku jeszcze kilka zmiennych liczbowych i wyświetl wyniki operacje arytmetycznych na nich (każda operacja co najmniej raz). Spróbuj dzielenie operacje na niezdefiniowanych zmiennych oraz dzielenie przez 0.
- W końcu napraw program tak, żeby on działał bez błędów.

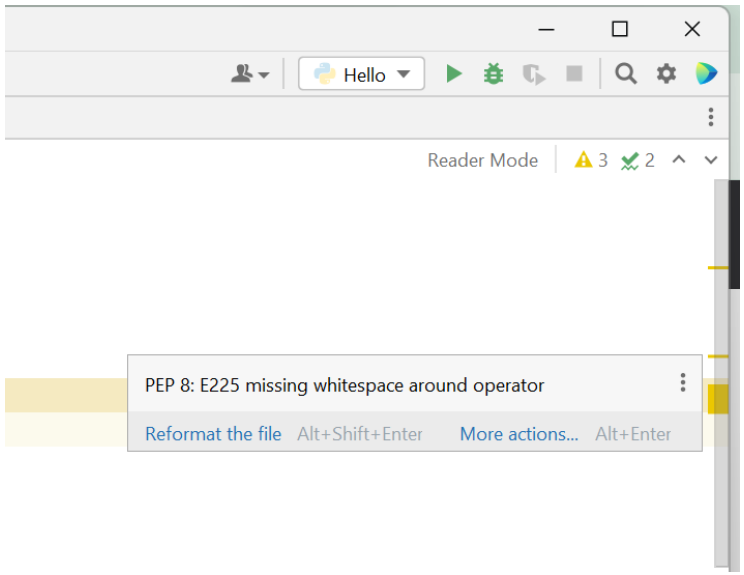
PEP8 w PyCharm

Jak sprawdzić, by kod spełniał standard PEP8?
`https://www.jetbrains.com/help/pycharm/tutorial-code-quality-assistance-tips-and-tricks.html#6966da55`



Settings -> Editor -> Indpections -> Python -> PEP8 ...
Dodać podpowiedzi:





Dodaj do `Hello.py` poniższy kod:

```
var1 = 2
var2 = 3
var1 += 2
var3 = var1
var1 *= 3
var4 = var1/var2
```

- Na trzeciej linijce ustaw breakpointa i uruchom tryb debug. Sprawdź wartości poszczególnych zmiennych w kolejnych etapach wykonywania programu.
- Spróbuj polecenie `print (var1+=2)`. Co się dzieje?
- Dodaj

```
 print(var1, var2,var3, var4)
var4 /=2
print('var4 = ', var4)
```

program11.py

Napisz program, który pobierze z klawiatury liczbą całkowitą n i wyświetla na konsoli tabliczkę mnożenia od 0 do n.
Dodaj opcje, która wyświetla tabliczkę tylko jeżeli $0 < n < 100$, inaczej wyświetla "n is too large".

program11.py

Zakomentuj poprzednią cześć w pliku.
Napisz program, który pobierze z klawiatury dwie liczby całkowitych a, b i wyświetla liczby p, q taki że $\frac{a}{b} = \frac{p}{q}$ i $\frac{p}{q}$ jest ułamkiem nieskracalnym.
Wskazówka: aby znaleźć największy wspólny dzielnik skorzystaj z funkcji math.gcd() i dołącz bibliotekę math.
Zmień program tak, żeby tam była funkcja, która bierze a, b i zwraca p, q.

https://www.learnpython.org/pl/Funkcje
https://www.w3schools.com/python/python_functions.asp

Metoda Newtona dla pierwiastka

program12.py

Napisz funkcje `sqrt`, która oblicza pierwiastek x danej liczby a metodą Newtona z dokładnością $|x^2 - a| < \epsilon$, gdzie a, ϵ, x_0 są parametrami funkcja. Domyślnie $x_0 = 1, \epsilon = 0,0001$

Porównaj obliczony x z math.sqrt(a).

`https://pl.wikipedia.org/wiki/Metoda_Newtona`

Zmodyfikuj funkcje tak, żeby warunkiem zakończenia było $|x_{n+1} - x_n| < \epsilon$.

Zmodyfikuj funkcje tak, żeby szukała pierwiastek równania $x^3 + ax^2 + bx + c = 0$ do $|x_{n+1} - x_n| < \epsilon$, ale nie więcej niż 100 iteracji (a, b, c podaję się z klawiatury). W przypadku zakończenia z powodu przekroczenia 100 iteracji na konsoli musi być wyświetlano „`no convergence`”
Sprawdzić dla $x^3 + 5x^2 + 4x + 2$ i $x^3 - 2x + 2$.

(Pierwiastki równania można sprawdzić przy pomocy strony `https://www.wolframalpha.com/`)

pd1.py

Napisz funkcje, która bierze liczbą całkowitą n jako parametr i oblicza

- $$e1 = \left(1 + \frac{1}{n}\right)^n$$
- $$e2 = \sum_{k=0}^{n} \frac{1}{k!}$$

  (silnią k obliczyć przy pomocy cyklu, nie używać biblioteki math).

pd1.py

- Wyświetlić $e1, e2$ dla różnych n.
- Porównaj $e1, e2$ z math.e (Wyświetlić $e1 - math.e, e2 - math.e, |e1 - math.e|, |e2 - math.e|$).

pd2.py

Napisz funkcję *największy wspólny dzielnik* (gcd) dla dwóch liczb całkowitych.