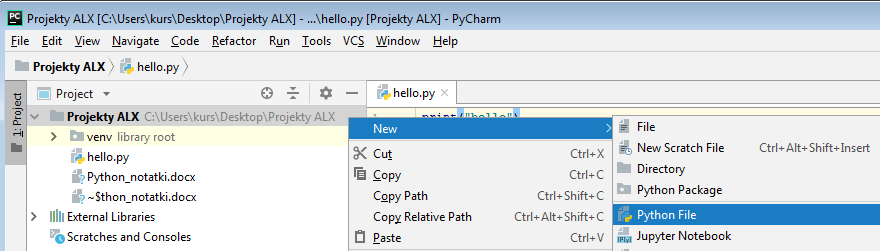
Uruchamianie nowego pliku



Zadanie 01:

*"""cena = 100  
wiek = 2.5  
temat = "Kurs Python"  
pytanie = "?"  
  
print(cena)  
print(wiek)  
print(temat)  
print(pytanie)  
print(type(cena))  
print(type(wiek))  
print(type(temat))  
print(type(pytanie))  
print(cena+wiek)  
print("czeko", end="")"""*ile = 15  
print(**"Grupa liczy "**, end=**""**)  
print(ile)  
print(**"Grupa liczy"**,ile)  
  
print(**"Programowanie w Python"**,**"Programowanie"**,100)  
print(**"Programowanie w Python"**, **"Programowanie"**, 100) *# spacje robi przecinek, ale nie jesteśmy w stanie go tu usunąć*

Zadanie 02:

cena = 100  
wiek = 2.5  
temat = **"Kurs Python"**pytanie = **"?"**print(cena)  
print(wiek)  
print(temat)  
print(pytanie)  
print(cena, wiek, temat, pytanie)

zadanie 03:

*"""imie = "Robert"  
print("czesc" + imie)  
print("czesc " + imie)  
  
ile = 20  
print("do zaplaty " + str(ile) + " zł") # str(ile) to konwersja liczby na stringa, bo wystarczy ze w konkatenacji jest jeden string, to domaga się aby cała reszta byla stringiem  
"""*zmienna1 = **"poznajemy"**zmienna2 = **"Pythona"**zmienna3 = **"Dzisiaj"**zmienna4 = **"wiele"***# napisz zdanie "//Dzisiaj poznajemy możliwości Pythona, których jest wiele"*print(**"//"** + zmienna3 + **" "** + zmienna1 + **" możliwości "** + zmienna2 + **", których jest "** + zmienna4 + **"."**) *# konkatenacja tylko z "+"*print(**"//"** + zmienna3, zmienna1, **"możliwości"**, zmienna2 + **","**, **"których jest"**, zmienna4 + **"."**) *# łączenie mieszane z "," i "+"*

Zadanie 04:

*"""  
a = 1  
print(a)  
  
a = 12  
print(a) # modyfikacja zmiennej  
  
a = 1 # tworzenie zmiennej  
b = 100 # tworzenie zmiennej  
b = a # teraz b = a, a nie 100  
print(b)  
"""*a = 10  
b = 1.5  
c = **"test"***# print(a, b, c)*print(a)  
print(b)  
print(c)  
  
a = 5.5  
b = **"abc"**c = 0  
print(a)  
print(b)  
print(c)  
  
b = c  
a = 13  
print(a)  
print(b)  
print(c)

Zadanie 05:

zm1 = **"Ciekawe"**zm2 = **"Programowanie"**zm3 = **"Jest"**zm4 = **"Wciągające"**zm5 = **"I"***#Programowanie jest ciekawe i wciagajace*pomocnicza = zm1  
zm1 = zm2  
zm2 = pomocnicza  
  
pomocnicza = zm2  
zm2 = zm3  
zm3 = pomocnicza  
  
pomocnicza = zm4  
zm4 = zm5  
zm5 = pomocnicza  
  
print(zm1, zm2, zm3, zm4, zm5)

zadanie 06:

zm1 = **"Ciekawe"**zm2 = **"Programowanie"**zm3 = **"Jest"**zm4 = **"Wciągające"**zm5 = **"I"***#Programowanie jest ciekawe i wciagajace*pomocnicza = zm1  
zm1 = zm2  
zm2 = pomocnicza  
  
pomocnicza = zm2  
zm2 = zm3  
zm3 = pomocnicza  
  
pomocnicza = zm4  
zm4 = zm5  
zm5 = pomocnicza  
  
print(zm1, zm2, zm3, zm4, zm5)

zadanie 07:

imie = **"Patrycja"**nazwisko = **"Matalewska"**miejsceUrodzenia = **"Katowice"**dataUrodzenia = **"15/03/1984"**pesel = 12345  
  
print(imie + **" "** + nazwisko + **" urodzona "** + miejsceUrodzenia + **" w dniu "** + dataUrodzenia + **" o peselu "** + str(pesel) + **"."**)

zadanie 07:

imie = **"Patrycja"**nazwisko = **"Matalewska"**miejsceUrodzenia = **"Katowice"**dataUrodzenia = **"15/03/1984"**pesel = 12345  
  
print(imie + **" "** + nazwisko + **" urodzona "** + miejsceUrodzenia + **" w dniu "** + dataUrodzenia + **" o peselu "** + str(pesel) + **"."**)

zadanie 08:

*"""  
and = 8  
print(and)  
  
zm = False  
print(type(zm))  
  
  
zm = 3.14159  
print(int(zm))  
print(round(zm,0))  
print(round(zm,2))  
"""*zm1 = bool(0) *# fałsz*zm2 = bool(**""**)  
print(zm1, zm2)

Zadanie 09:

zm1 = 100  
zm2 = 3.14  
zm3 = **"Patrycja"**zm4 = **True**print(zm1, zm2, zm3, zm4)  
  
print(type(zm1))  
print(type(zm2))  
print(type(zm3))  
print(type(zm4))  
  
zm1 = 4.447  
print(type(zm1))  
  
zm2 = **"poiuyt"**print(type(zm2))  
  
zm3 = **False**print(type(zm3))  
  
zm4 = 200  
print(type(zm4))

zadanie 10:

a = **"test"**print(a\*20) *# testtesttest, napis zostanie powielony*

zadanie 11:

chleb = 1.99  
mleko = 2.50  
cukierki = 12.99  
  
zamowienie = (2 \* chleb) + (0.5 \* mleko) + (0.4 \* cukierki)  
print(round(zamowienie,2))

zadanie 15:

imie = **"Adam"**nazwisko = **"Kowalski"**wiek = 35  
pensja = 4000  
stanowisko = **"mechanik"**print((**"Pan "** + imie + **" "** + nazwisko + **" (wiek: "** + str(wiek) + **" lat) pracuje na stanowisku: "** + stanowisko + **" (pensja: "** + str(pensja) + **" brutto). \n"**)\*10)  
  
print(**"brutto\n"** \* 3)

zadanie 15\_1:

*# pole kwadratu o boku 2 [cm]*a = 2  
pole\_kwadratu = a \*\* 2  
print(pole\_kwadratu)  
  
*# obwód trójkąta o bokach 3, 4, 6 [cm]*tr1 = 3  
tr2 = 4  
tr3 = 6  
obwod\_trojkata = tr1 + tr2 + tr3  
print(obwod\_trojkata)  
  
*# pole koła o promeniu 3 [cm]*pi = 3.14  
r = 3  
pole\_kola = pi \* r \*\* 2  
print(pole\_kola)

Zadanie 16:

print(2+2)  
print(10-7)  
print(4\*5)  
print(12/4) *# wynik dzielenia zawsze jest float*print(20%8)  
print(3\*\*2)

zadanie 17:

a = 1  
b = 2  
c = 3  
d = 4  
e = 5  
  
a = a \* e  
d = d - b  
c = c \* a  
a = (b + d + e) / 3  
c = c / a  
b = b + b  
  
a = a + a + c  
e = b \* c \* a  
d = (a \* b) + (c \* e)  
c = c - (a + e)  
b = 3 \* b  
d = b  
wynik = a + b + c + d + e  
  
print(wynik)

zadanie 17\_1:

*"""  
Samochód na 100 km spala 8 l paliwa. Ile spali paliwa po przejechaniu 382 km?  
"""*spalanie = 8  
trasa = 100  
droga = 382  
  
ile\_spali = (spalanie/trasa) \* droga  
  
print(round(ile\_spali, 2))  
print(**"Samochód, ktory spala "** + str(spalanie) + **" litrów na "** + str(trasa) + **" km, spala średnio "** + str(ile\_spali) + **" na dystansie "** + str(droga) + **" km."**)

Zadanie 18:

kwota\_brutto = 1000  
vat1 = 1.03  
vat2 = 1.07  
vat3 = 1.23  
  
kwota\_netto1 = kwota\_brutto / vat1  
kwota\_netto2 = kwota\_brutto / vat2  
kwota\_netto3 = kwota\_brutto / vat3  
  
print(kwota\_netto1)  
print(kwota\_netto2)  
print(kwota\_netto3)  
  
*#print("%.3f" % kwota\_netto2)*print(round(kwota\_netto1,3))  
print(round(kwota\_netto2,3))  
print(round(kwota\_netto3,3))

zadanie 19:

wynagrodzenie\_netto = 5500  
godziny\_pracy = 168  
  
stawka\_godzinowa\_netto = wynagrodzenie\_netto / godziny\_pracy  
print(stawka\_godzinowa\_netto) *# bez zaokrąglenia*print(round(stawka\_godzinowa\_netto, 2)) *# z zaokrągleniem*podatek\_dochodowy = 18  
  
stawka\_godzinowa\_brutto = stawka\_godzinowa\_netto \* (1 + podatek\_dochodowy/100)  
print(stawka\_godzinowa\_brutto) *# bez zaokrąglenia*print(round(stawka\_godzinowa\_brutto, 2)) *# z zaokrągleniem*

zadanie 19\_1:

*"""znajdź klucz dla zmiennej a=2, zaprojektuj algorytm i podaj wartosci dla x, y, z  
  
2 6 14 30 62 x y z"""*a = 2  
x = 62  
  
x = x \* a + a  
print(x)  
  
x = x \* a + a  
print(x)  
  
x = x \* a + a  
print(x)

Zadanie 20:

*"""  
Firma X w styczniu miala dochod 700 zł ....  
"""*dochod = 700  
koszty = 500 *# koszty są stałe w każdym miesiącu*zysk = dochod - koszty  
  
dochod2 = dochod \* 1.5  
zysk2 = dochod2 - koszty  
  
dochod3 = dochod2 \* 1.5  
zysk3 = dochod3 - koszty  
  
dochod4 = dochod3 \* 1.5  
zysk4 = dochod4 - koszty  
  
dochod5 = dochod4 \* 1.5  
zysk5 = dochod5 - koszty  
  
dochod6 = dochod5 \* 1.5  
zysk6 = dochod6 - koszty  
  
print(round(zysk, 2))  
print(round(zysk2, 2))  
print(round(zysk3, 2))  
print(round(zysk4, 2))  
print(round(zysk5, 2))  
print(round(zysk6, 2))  
  
**""" wynik:  
200  
500  
1075  
1862.50  
3043.75  
4815.62  
"""**

Zadanie 20\_1:

*"""  
posiadamy 2 drukarki czarnobiale i kolorowa, kolora drukuje 2 kartki na 5 minyt, czarnobiala 8 kartek na 2 minuty.  
mamy 45 minut czasu na druki  
ile wydrukujemy w sumie kartek w kolorze i czarnobieli  
jaki jest wynik  
  
wynik to 198  
kolorowych 18  
czarnobialych 180  
"""*czarno\_biala = 2/5  
kolorowa = 8/2  
czas = 45  
  
print(czarno\_biala\*czas + kolorowa\*czas)

Zadanie 24:

*"""  
napis = input("Wprowadź jakikolwiek napis: ")  
print((napis + "\n" ) \* 5)  
"""*napis = input(**"Wprowadź jakikolwiek napis: "**)  
print(napis)  
ile = input(**"Wprowadź ilość powtórzeń: "**)  
print((napis + **"\n"**) \* int(ile))

Zadanie 25:

*"""  
Program, który oblicy pole trojkata, pod warunkiem ze uzytkownik poda wysokosc i dlugosz podstawy tego trojkata.  
Uwzglednij, ze wysokosc i dlugosc podstawy moga byc liczbami niecalkowitymi.  
"""*podstawa = float(input(**"Podaj podstawę trójkąta: "**)) *# zamiana na float może być również później (w obl. pola trójkąta), ale dobrą praktyka jest dawanie tego juz tutaj*wysokosc = float(input(**"Podaj wysokość trójkąta: "**))  
  
pole\_trojkata = 0.5 \* podstawa \* wysokosc  
  
print(**"Pole trójkąta wynosi: "** + str(pole\_trojkata) + **"."**)

Zadanie 26:

liczba1 = int(input(**"Podaj liczbę nr 1: "**))  
liczba2 = int(input(**"Podaj liczbę nr 2: "**))  
liczba3 = int(input(**"Podaj liczbę nr 3: "**))  
  
srednia = (liczba1 + liczba2 + liczba3) / 3  
  
print(**"Średnia z podanych liczb wynosi: "** + str(srednia) + **"."**)

zadanie 27:

podstawa = float(input(**"Podaj podstawę potęgi: "**))  
wykladnik = int(input(**"Podaj wykladnik potęgi: "**))  
  
potega = round(podstawa \*\* wykladnik, 3)  
  
print(**"Wynik potęgowania to: "** + str(potega) + **"."**)

zadanie 28\_1:

*"""  
program ktory wczyta od uzytkowanika dowolny tekst  
program ma za zadanie wypisac:  
- ile wprowadzono znakow w wprowadzonych tekscie,  
- ile jest spacji we wprowadzonym tekscie  
"""*tekst = input(**"Podaj dowolny tekst: "**)  
ilosc\_znakow = len(tekst)  
znak2 = tekst.replace(**" "**, **""**)  
ilosc\_znakow2 = len(znak2)  
  
print(tekst)  
print(znak2)  
  
print(**"W tekście jest "** + str(ilosc\_znakow) + **" znaków."**)  
print(**"W tekście jest "** + str(ilosc\_znakow - ilosc\_znakow2) + **" spacji."**)  
  
**""" nieudana proba:  
ilosc\_spacji = (tekst.count(" ")) # zlicza spacje  
print("W podanym tekście jest: " + str(ilosc\_znakow) + " sztuk znaków.")  
print("W podanym tekście jest: " + str(ilosc\_spacji) + " sztuk spacji.")  
print(tekst.replace(" ", "")) # zamienia znaki  
print(tekst.count("0")) # zlicza podane znaki  
print(ilosc\_spacji)  
"""**

Zadanie 29:

chleb = 6.50  
sok = 4  
paczek = 5.50  
  
suma\_chleb = int(input(**"Wpisz, ile sztuk chleba chcesz kupić? "**))  
suma\_sok = int(input(**"Wpisz, ile litrów soku chcesz kupić? "**))  
suma\_paczek = int(input(**"Wpisz, ile sztuk pączków chcesz kupić? "**))  
print(**"Twoje zamówienie to:"**)  
print(**" - "** + str(suma\_chleb) + **" chleby po "** + str(chleb) + **" zł za bochenek,"**)  
print(**" - "** + str(suma\_sok) + **" soki po "** + str(sok) + **" zł za litr,"**)  
print(**" - "** + str(suma\_paczek) + **" pączki po "** + str(paczek) + **" zł za sztukę."**)  
zamówienie = round(suma\_chleb \* chleb + suma\_sok \* sok + suma\_paczek \* paczek, 2)  
  
print(**"Twoje zamówienie razem to: "** + str(zamówienie) + **" zł."**)

zadanie 30:

nazwa = **"ALX"**print(nazwa[2])  
print(len(nazwa)) *# zlicza wszystkie znaki*print(nazwa.capitalize()) *# pierwsza litera zawsze WIELKA reszta małe*print(nazwa.count(**"X"**)) *# zlicza podane znaki*print(nazwa.replace(**"X"**, **"V"**)) *# zamienia znaki*print(nazwa.strip()) *# usuwa początkowe i końcowe białe znaki*

zadanie 31:

lista = [0, 2, 4, 6, 8]  
print(lista)  
print(lista[0]) *# wyświetla element o wybranym indeksie, tutaj indeks nr 0*print(lista[4]) *# wyświetla element o wybranym indeksie, tutaj indeks nr 4*lista.append(10) *# dodaje zawsze jeden element zawsze na końcu*print(lista)  
**del** lista[2] *# usuwa*print(lista)  
lista.insert(2, 4) *# dodaje na konkretnej pozycji*print(lista)  
print(2 **in** lista)  
lista.pop(1)  
print(lista)  
  
kursy = [ [**"Python"**, 15, **"Katowice"**], [**"Java"**, 10, **"Kraków"**] ] *# 2 elementy, może być więcej elementów, ale zwykle są 2  
# jak dostać się do informacji "Java"*print(kursy[1][0])

zadanie 31\_1:

*"""  
Utworz liste z 5 skladająca się z elementow artykulow spozywczych  
wyswietla na ekranie jako tekst zawartosc pierszego i ostatniego elementu z listy (wykorzystaj 2 spowoby)  
Uwzględnij przypadek, ze dlugosc listy moze ulec zmianie  
"""*lista = [**"chleb"**, **"jabłko"**, **"sok"**, **"woda"**, **"piwo"**]  
  
*# wyświetlanie pierwszego, sposoby:*print(lista[0])  
print(lista.pop(0))  
  
*# wyświetlanie ostatniego, sposoby:*print(lista[-1]) *# nie podajemy nr kolejnego, tylko odnosimy sie do ostatniego, bo ilosc moze sie zmienic*print(lista[len(lista)-1])  
lista.reverse()  
print(lista.pop(-1))  
print(lista) *# ostatni sposób usunął 1 i ostatni element po jego wyświetleniu, takze to tylko ciekawostka*

Zadanie 32:

lista1 = [2,8,11,66,99]  
lista2 = lista1 *# przypisanie zmiennej do zmiennej*print(lista1)  
print(lista2)  
  
lista1[0] = 123  
print(lista1)  
print(lista2)

zadanie 33:

*"""  
TYPY SEKWENCYJNE  
  
napisz program, ktory 5 razy poprasi o podanie imienia  
podane imiona beda zapisywane do listy  
wypisz dla wszystkich imion ponizszy komunikat  
Cześć <tutaj imię z listy>  
"""*imiona = [] *# pusta lista*im1 = input(**"Wpisz imię nr 1: "**)  
imiona.append(im1)  
  
im2 = input(**"Wpisz imię nr 2: "**)  
imiona.append(im2)  
  
  
im3 = input(**"Wpisz imię nr 3: "**)  
imiona.append(im3)  
  
im4 = input(**"Wpisz imię nr 4: "**)  
imiona.append(im4)  
  
im5 = input(**"Wpisz imię nr 5: "**)  
imiona.append(im5)  
  
print(**"Cześć "** + imiona[0] + **"! Miło mi Cię poznać :)"**) *# odwołujemy się do listy, dlatego używamy imiona[0] a nie im1*print(**"Cześć "** + imiona[1] + **"! Miło mi Cię poznać :)"**)  
print(**"Cześć "** + imiona[2] + **"! Miło mi Cię poznać :)"**)  
print(**"Cześć "** + imiona[3] + **"! Miło mi Cię poznać :)"**)  
print(**"Cześć "** + imiona[4] + **"! Miło mi Cię poznać :)"**)  
  
print(imiona)

Zadanie 34:

*"""  
Napisz program, w ktorym zadeklarujesz dwie listy, ktore beda przechowywaly po 3 dowolne liczby  
Program powinien wyswietlic sume wszystkich liczb z obu tablic  
"""*lista1 = [1, 2, 3]  
lista2 = [1, 2, 3]  
  
suma = lista1[0] + lista1[1] + lista1[2] + lista2[0] + lista2[1] + lista2[2] *# odwoływanie do indeksów*print(suma) *# wynik to 12*lista3 = [2,5,6]  
iloczyn = lista3[0] \* lista3[1] \* lista3[2]  
print(iloczyn) *# wynik to 60*lista4 = [100,5,2]  
iloraz = lista4[0] / lista4[1] / lista4[2]  
print(iloraz) *# wynik to 10*

Zadanie 35:

*"""  
podaj dwie liczby slownie, wynik ma byc liczbą  
zakres [1-5]  
i odwrotnie  
"""*lista = {**"jeden"**:1, **"dwa"**:2, **"trzy"**:3, **"cztery"**:4, **"pięć"**:5}  
liczba1 = input(**"Podaj słownie liczbę pierwszą z zakresu [1-5]: "**)  
liczba2 = input(**"Podaj słownie liczbę drugą z zakresu [1-5]: "**)  
print(lista[liczba1] + lista[liczba2])  
*# print(lista[liczba1 + liczba2]) taki zapis tutaj nie działa*lista2 = {1:**"jeden"**, 2:**"dwa"**, 3:**"trzy"**, 4:**"cztery"**, 5:**"pięć"**} *# to będzie działać, ale do sumy max "pięć", bo innych liczb nie zdefiniowaliśmy*liczba3 = int(input(**"Podaj NIEsłownie liczbę pierwszą z zakresu [1-5]: "**)) *# trzeba zamienić na intiger*liczba4 = int(input(**"Podaj NIEsłownie liczbę drugą z zakresu [1-5]: "**))  
print(lista2[liczba3 + liczba4])

Zadanie 36\_If:

**if** (2 > 5):  
 print(**"OK"**)  
print(**"OK2"**) *# to nie jest częścią if*print(**"OK3"**) *# to nie jest częścią if*print(**"OK4"**) *# to nie jest częścią if***"""  
if (2 > 5):  
 print("OK")   
 print("OK2")  
 print("OK3")  
 print("OK4")  
"""***# if ((2>5) and ()): stosujemy nawiasy***if** (2 > 5):  
 print(**"OK"**)  
**else**:  
 print(**"OK2"**)  
 print(**"OK3"**)  
 print(**"OK4"**)  
  
zmienna = input(**"Podaj cyfrę: "**)  
zmienna = int(zmienna)  
  
**if** (zmienna > 10):  
 print(**"Wpisałeś cyfrę większą niż dziesięć"**)  
**else**:  
 print(**"Wpisałeś cyfrę mniejszą niż dziesięć"**)  
print(**"Koniec programu"**)  
  
  
**if** (zmienna > 0):  
 print(**"Wpisałeś cyfrę dodatnią"**)  
**elif** (zmienna == 0):  
 print(**"Wpisałeś zero"**)  
**else**:  
 print(**"Wpisałeś cyfrę ujemną"**)  
print(**"Koniec programu"**)  
  
  
  
**if** (zmienna > 0):  
 print(**"Wpisałeś cyfrę dodatnią"**)  
 **if** (zmienna % 2 == 0):  
 print(**"Liczba parzysta"**)  
 **else**:  
 print(**"Liczba nieparzysta"**)  
**elif** (zmienna == 0):  
 print(**"Wpisałeś zero"**)  
**else**:  
 print(**"Wpisałeś cyfrę ujemną"**)  
print(**"Koniec programu"**)

zadanie hello:

*#to wypisuje dane na ekranie  
  
"""to  
jest komentarz  
blokowy"""*print(**"hello"**)  
imie = **"Robert"**wiek = 12.5  
  
print(wiek\*2)  
print(type(wiek))  
kodPocztowy = 40-820 *#zmienna tworzona przez Wielką literę, tzw. camel case*kod\_pocztowy = 40-820

Zadanie If:

*"""if (True):  
 print("tekst1")  
  
if (True):  
 print("tekst2")  
  
if (True):  
 print("tekst3")  
"""***if** (**True**):  
 print(**"tekst1"**)  
**elif** (**True**):  
 print(**"tekst2"**)  
**elif** (**True**):  
 print(**"tekst3"**)

Zadanie If\_2:

*"""  
Napisz program ktory sprawdz czy wpisana liczba jest parzysta i nieparzysta  
"""*liczba = input(**"Wpisz liczbę: "**)  
liczba = int(liczba)  
  
**if** (liczba % 2 == 0):  
 print(**"Liczba parzysta"**)  
**else**:  
 print(**"Liczba nieparzysta"**)  
  
**"""  
liczba = input("Wpisz liczbę: ")  
liczba = int(liczba)  
  
if (liczba % 2 != 0):  
 print("Liczba nieparzysta")  
else:  
 print("Liczba parzysta")  
"""**

Zadanie If\_3:

*"""  
Napisz program ktory sprawdza  
"""***"""  
a = 2  
b = 100  
  
if (a > b):  
 print("Zmienna a większa niż b")  
else:  
 print("Zmienna a nie jest większa niż b")  
"""**a = 1  
b = 2  
c = 300  
*# założenia, żeby poniższe działało to: a,b,c nie mogą być sobie równe i muszą być różne od zera***if** ((a > b) **and** (a>c)):  
 print(**"Zmienna a jest największa"**)  
**elif**((b>a) **and** (b>c)):  
 print(**"Zmienna b jest największa"**)  
**else**:  
 print(**"Zmienna c jest największa"**)

Zadanie If\_4:

*"""  
Wczytaj z klawiatury 3 wartosci i wypisz je po kolei od najwiekszego do najmnijeszej  
"""***"""  
a = float(input("Wpisz liczbę nr 1: "))  
b = float(input("Wpisz liczbę nr 2: "))  
c = float(input("Wpisz liczbę nr 3: "))  
  
if ((a > b) and (a > c)):  
 if (b > c):  
 print(a, b, c)  
 else:  
 print(a, c, b)  
elif((b > a) and (b > c)):  
 if (a > c):  
 print(b, a, c)  
 else:  
 print(b, c, a)  
elif ((c > a) and (c > b)):  
 if (a > b):  
 print(c, a, b)  
 else:  
 print(c, b, a)  
"""**a = input(**"Wpisz liczbę nr 1: "**)  
b = input(**"Wpisz liczbę nr 2: "**)  
c = input(**"Wpisz liczbę nr 3: "**)  
  
**if** ((a > b) **and** (a > c)):  
 **if** (b > c):  
 print(a, b, c)  
 **else**:  
 print(a, c, b)  
**elif**((b > a) **and** (b > c)):  
 **if** (a > c):  
 print(b, a, c)  
 **else**:  
 print(b, c, a)  
**elif** ((c > a) **and** (c > b)):  
 **if** (a > b):  
 print(c, a, b)  
 **else**:  
 print(c, b, a)

Zadanie If\_5:

*"""  
Napisz program wczytujący 3 zmienne i sprawdzające czy z odcinków  
o takich długościach da się zbudować trójkat prostokatny  
a2 + b2 = c2  
Na koncu ma wypisac informacje  
"""  
  
# trojkat prostokatny zrobi sie z danych 3, 4 i 5  
# print(pow(2, 3)) - to jest 2 do potęgi 3*a = int(input(**"Wpisz bok a: "**))  
b = int(input(**"Wpisz bok b: "**))  
c = int(input(**"Wpisz bok c: "**))  
  
**""" METODA z symbolem "\*\*" potęgi  
if(a\*\*2 + b\*\*2 == c \*\* 2):  
 print("Z podanych liczb MOŻNA zbudować trójkąt prostokątny")  
else:  
 print("Z podanych liczb NIE można zbudować trójkąta prostokątnego")  
"""***# metoda z potęgą zapisaną pow(a, 2)***if**(pow(a,2) + pow(b, 2) == pow(c, 2)):  
 print(**"Z podanych liczb MOŻNA zbudować trójkąt prostokątny"**)  
**else**:  
 print(**"Z podanych liczb NIE można zbudować trójkąta prostokątnego"**)

Zadanie If\_6:

*"""  
Napisz prosty kalkulator dla liczb zmiennoprzecinkowych, ktory pozwala uzytkownikowi kolejno:  
- wprowadzenie pierwszej liczby  
- wprowadzenie jednego z podstawowych dzialam matematycznych (plus, minus, podzeil, pomnoz) w postaci znaków = - \* /  
- wprowadzenie drugie liczby  
"""*a = float(input(**"Wpisz pierwszą dowolną liczbę rzeczywistą: "**))  
b = input(**"Wpisz jeden wybrany znak '+', '-', '\*', '/': "**)  
c = float(input(**"Wpisz pierwszą dowolną liczbę rzeczywistą: "**))  
  
**if** (b == **"+"**):  
 print(a + c)  
**elif**(b == **"-"**):  
 print(a + c)  
**elif** (b == **"\*"**):  
 print(a \* c)  
**elif** (b == **"/"**):  
 **if** (c == 0):  
 print(**"Dzielenie przez zero"**)  
 **else**:  
 print(a / c)

Zadanie If\_7:

*"""  
Wczytaj od uzytkownika 5 liczb calkowitych i wypisz na ekran najwiekszą oraz najmniejsza z nich  
Zakładają ze wrowadzone liczby rożnią się od siebie  
  
Nie mozna uzywac sortowania, min i max, bo jeszcze tego nie znamy  
"""*liczba = int(input(**"Podaj liczbę: "**))  
max = liczba  
min = liczba  
  
liczba = int(input(**"Podaj liczbę: "**))  
  
**if** (liczba > max):  
 max = liczba  
**elif** (liczba < min):  
 min = liczba  
  
liczba = int(input(**"Podaj liczbę: "**))  
  
**if** (liczba > max):  
 max = liczba  
**elif** (liczba < min):  
 min = liczba  
  
liczba = int(input(**"Podaj liczbę: "**))  
  
**if** (liczba > max):  
 max = liczba  
**elif** (liczba < min):  
 min = liczba  
  
liczba = int(input(**"Podaj liczbę: "**))  
  
**if** (liczba > max):  
 max = liczba  
**elif** (liczba < min):  
 min = liczba  
  
print(max, min)  
  
  
  
**"""  
a = int(input("Wpisz 1-szą liczbę całkowitą: "))  
b = int(input("Wpisz 2-gą liczbę całkowitą: "))  
c = int(input("Wpisz 3-cią liczbę całkowitą: "))  
d = int(input("Wpisz 4-tą liczbę całkowitą: "))  
e = int(input("Wpisz 5-tą liczbę całkowitą: "))  
  
if (a > b > c > d > e):  
 print(a)  
 print(e)  
"""**

Zadanie imie:

*"""  
print("test1")  
imie = input("Wprowadź swoje imię: ")  
print("test2")  
wiek = int(input("Wprowadź swój wiek: "))  
print("test3")  
wiek = float(input("Wprowadź swój wiek: "))  
print("test4")  
"""*imie = input(**"Wprowadź swoje imię: "**)  
print(imie + **", witaj w kursie Python."**)

Zadanie krotka:

*"""  
dane = (1,5,3,8)  
print(dane)  
dane[1] = # nie da się modyfikowac krotki  
print(dane[1])  
"""*dane = [1,5,3,6]  
dane = tuple(dane)  
print(type(dane)) *# sprawdzenie rodzaju*dane = (1,5,3,6)  
dane = list(dane)  
print(type(dane)) *# sprawdzenie rodzaju*dane = [1,5,3,6]  
dane1 = tuple(dane)  
dane[1] = **"X"**print(dane)  
print(dane1)  
  
dane = [1,5,3,6]  
dane[1] = **"X"**dane1 = tuple(dane)  
print(dane)  
print(dane1)  
  
  
*# len(krotka)***"""  
Napisz program, ktory tekst np. "Witaj na kursie Python" przechowuje w postaci krotki slowo po slowu  
Nastepnie wypisz na ekran calą zawartosc krotki  
"""**tekst = (**"Witaj"**, **"na"**, **"kursie"**, **"Python"**)  
print(tekst)

Zadanie słowniki:

*"""  
SLOWNIKI, stosujemy nawiasy klamrowe {}  
"""  
  
# utworzenie słownika*magazyn = {**"chleb"**:10, **"mleko"**:3, **"masło"**:5, **"szynka"**:8}  
print(magazyn)  
  
*# dodanie elementu*magazyn[**"pomidor"**] = 11  
print(magazyn)  
magazyn[**"banany"**] = 500  
print(magazyn)  
magazyn[**"banany"**] = 1500  
print(magazyn)  
  
*# usuwanie elementu***del** magazyn[**"banany"**]  
print(magazyn)  
  
*# pusty słownik, do którego dodajemy*z = {}  
z[**"vvv"**] = 200  
print(z)  
  
  
**"""  
Napisz progrma ktory zmieni podana przez uzytkowanika liczbe zapisana slownie na wartosc (z zakresu od 1 do 5)   
na odpowiadajaca jej liczbe dziesietną  
"""**lista1 = {**"jeden"**:1, **"dwa"**:2, **"trzy"**:3, **"cztery"**:4, **"pięć"**:5}  
liczba1 = input(**"Podaj słownie liczbę [1-5]: "**)  
print(lista1[liczba1])  
  
**"""  
coś nie dziala  
lista2 = {1:"jeden", 2:"dwa", 3:"trzy", 4:"cztery", 5:"pięć"}  
liczba2 = input("Podaj NIEsłownie liczbę [1-5]: ")  
print(lista2[liczba2])  
"""**

Zadanie zbiory:

*"""  
ZBIORY  
Napisz program zliczajacy liczbe wartosci unikatowych wprowadzonych przez uzytkownika  
"""*zbior = set() *# deklaracja pustego zbioru*liczba1 = input(**"Wpisz liczbę: "**)  
zbior.add(liczba1)  
liczba2 = input(**"Wpisz liczbę: "**)  
zbior.add(liczba2)  
liczba3 = input(**"Wpisz liczbę: "**)  
zbior.add(liczba3)  
  
print(**"W zbiorze jest unikatowych cyfr "** + str(len(zbior)))  
  
print(zbior) *# zwraca wpisane dane w losowej kolejnosci  
  
  
# kolejnosc, + , ilosc liczb*

Zadanie If\_8:

*"""  
Pole = 3,14 \* r2  
Obwód = 2 \* 3,14 \* r  
  
Napisz program do obliczania pola lub obwodu kola o wskazanym promieniu  
Program powienien poprowsic o ponizsze dane  
- podaj co bedziemy liczyc - pole czy obwod  
- podaj promien kola  
"""*obliczenia = input(**"Wpisz 'pole lub 'obwód': "**)  
promien = float(input(**"Wpisz promien koła: "**))  
  
**if**(obliczenia == **"pole"**):  
 print(3.14 \* promien \* promien) *# zamiast "promien \* promien" można "promien \*\* 2"***elif**(obliczenia == **"obwód"**):  
 print(2 \* 3.14 \* promien)  
**else**:  
 print(**"Wpisano źle"**)

Zadanie Petla\_1:

*"""  
PĘTLE:  
  
for x in range(1, 15): # można też zapisać tak: for 1 in range(15):  
 imie = input("Podaj imię " + str(x) + " uczestnika: ")  
"""*lista = [1, 2, 3, 4, 5]  
**for** element **in** lista:  
 print(element)

Zadanie Petla\_2:

*# Pętle, zwracamy wartości i indeksy*lista = [1, 2, 3, 4, 5]  
**for** indeks, wartosc **in** enumerate(lista):  
 print(indeks, wartosc)

Zadanie Petla\_3:

*"""  
Pelta rowniez pozwala wykonac operacje na slownikach, zwracajac jego klucz  
"""*slownik = {**"jeden"**: 1, **"dwa"**: 2, **"trzy"**: 3}  
**for** key **in** slownik:  
 print(key, slownik[key])

Zadanie Petla\_4:

lista = [1,2,3,4,5,6,7,8,9,10]  
  
*# wypisuje liczby parzyste, wynik: 2,4,6,8,10***for** element **in** lista:  
 **if** (element % 2 == 0):  
 print(element)  
  
*# wypisuje liczby parzyste, wynik: 1,3,5,7,9***for** element **in** lista:  
 **if** (element % 2 != 0):  
 print(element)

Zadanie Petla\_5:

*# liczby 0-100 parzyste***for** x **in** range(0, 101):  
 **if** (x % 2 == 0):  
 print(x)  
  
  
*# liczby podzielne przez 5, 1 sposób***for** x **in** range(0, 101):  
 **if** (x % 5 == 0):  
 print(x)  
  
*# liczby podzielne przez 5, 2 sposób***for** x **in** range(5, 101, 5):  
 print(x)

Zadanie Petla\_6:

*# napisz program ktory wypelni liste liczbami od 1 do 10, a nastepnie wyswietli co trzeci element tablicy  
  
"""  
lista = [1,2,3,4,5,6,7,8,9,10]  
  
for x in lista:  
 if (x % 3 == 0):  
 print(x)  
"""***"""  
for x in range(1, 10):  
 if (x % 3 == 0):  
 print(x)  
"""**lista = []  
**for** x **in** range(1, 11):  
 lista.append(x)  
  
print(lista)  
  
**for** x **in** range(0, 10, 3):  
 print(x) *#* print(lista[x]) *# odwolujac sie do listy przypisuje do indeksow wartosci; petla jest niezalezna od listy  
 # dla for to jest liczba, dla print(lista[x]) to jest indeks  
 # list to taki magazyn danych  
  
  
  
#for x in range(100)  
 # print(x)*

Zadanie Petla\_7:

*# liczby 0-100, podaj ich srednią  
  
  
# tutaj print jest w środku pętli, więc podaje osobne sumy (wiele wyników)  
"""  
suma = 0  
for x in range(1, 101):  
 suma = suma + x  
 print(suma)  
 #print("Krok", x, "Suma", suma, "Średnia", suma/x)  
"""  
  
# tutaj print jest poza pętlą, więc podaje sumę całkowitą (jeden wynik)*suma = 0  
**for** x **in** range(1, 101):  
 suma = suma + x  
print(suma)

Zadanie Petla\_8:

*"""  
Pętle "while"  
"""***"""  
while(True):  
 print("Robert")  
# gdybyśmy to wywyłali, to kręciłoby się bez końca, do zagrzania procesora  
"""**x = 5  
**while** x > 0:  
 print(x)  
 x = x - 1

Zadanie Petla\_9:

*"""# zwraca liczby  
liczby = [-2, -1, 0, 1, 2]  
for x in liczby:  
 if x < 0:  
 continue  
 print(x)  
"""  
  
# Wypisz liczby od 0 do 10 bez 5***for** x **in** range(11):  
 **if** x == 5:  
 **continue** print(x)  
  
  
*# Wypisz liczby od 0 do 10 parzyste - rozwiązanie z continue***for** x **in** range(11):  
 **if** x % 2 != 0:  
 **continue** print(x)  
  
*# Wypisz liczby od 0 do 10 parzyste - rozwiązanie BEZ continue***for** x **in** range(11):  
 **if** x % 2 == 0:  
 print(x)

Zadanie Petla\_10:

*"""  
for x in range(110000):  
 if x == 10:  
 break  
  
 print(x)  
"""*

Zadanie Petla\_11:

*"""  
Napisz program wczytujacy z klawiatury wartości, a następnie wyswietlajacy srednia tych wartości,  
niech program konczy wprowadzanie, kiedy natrafi na cyfre 0, ktorej nie bierzemy do liczenia  
Komentarz: najlepiej wykorzystać while, zaczynamy z pustym kontem, czyli suma jest 0  
i ilosc wpisanych liczb (jako mianownik do średniej) też jest równy zero  
"""*suma = 0  
ile = 0  
**while**(**True**):  
 x = int(input(**"Podaj liczbę (0 aby zakończyć): "**))  
  
 **if** x == 0:  
 **break  
 else**:  
 suma = suma + x  
 ile = ile +1  
  
print(suma/ile)  
  
  
*# inny sposob na rozwiązanie:*sum = 0  
x = 0  
srednia = 0  
a = int(input(**"Wprowadź liczbę: "**))  
  
**while** (a != 0)  
 x = x + 1  
 sum = sum + a  
 srednia = sum / x  
 print(x, srednia)  
 a = int(input(**"Wprowadź liczbę: "**))  
**else**:  
 print(**"Koniec"**, **"Średnia = "**, srednia)

Zadanie Format:

*#for x in range(0, 100, 10):  
# print("%4i%6i%8i" % (x, x\*\*2, x\*\*3))  
  
#for x in range(100):  
# print("%4i%16i%8i" % (x, x\*\*2, x\*\*3))  
  
#for x in range(100):  
# print("%4i%10.2f" % (x, x\*\*0.5))  
  
#for x in range(-100, 100):  
# print("%+4i" % (x))  
  
#for x in range(-100, 100):  
# print("%+4i%10i" % (x, x\*\*2))  
  
  
# puste miejsca uzupelnione zerami  
#for x in range(-100, 100):  
# print("%+4i%010i" % (x, x\*\*2))  
  
# ze znakiem -+, wyrównanie***for** x **in** range(-100, 100):  
 print(**"%+-4i%010i"** % (x, x\*\*2))

Zadanie Gra:

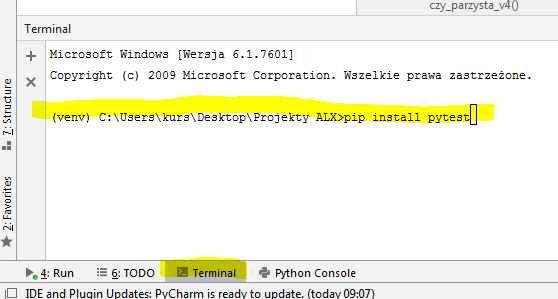
*"""  
Gra w losowanie liczby, program mówi że liczba jest za mała lub za duża, aż do trafienia  
Komentarz: można liczyć ilość prób  
"""***import** random  
  
liczba = random.randint(0,100) *# kod na losowanie liczby losowej***while**(**True**):  
 x = int(input(**"Podaj liczbę: "**))  
 **if** (x == liczba):  
 print(**"Brawo"**)  
 **break  
 elif** (x > liczba):  
 print(**"Podana liczba jest za duża"**)  
 **elif** (x < liczba):  
 print(**"Podana liczba jest za mała"**)

====== MODUŁ nr 2 =====

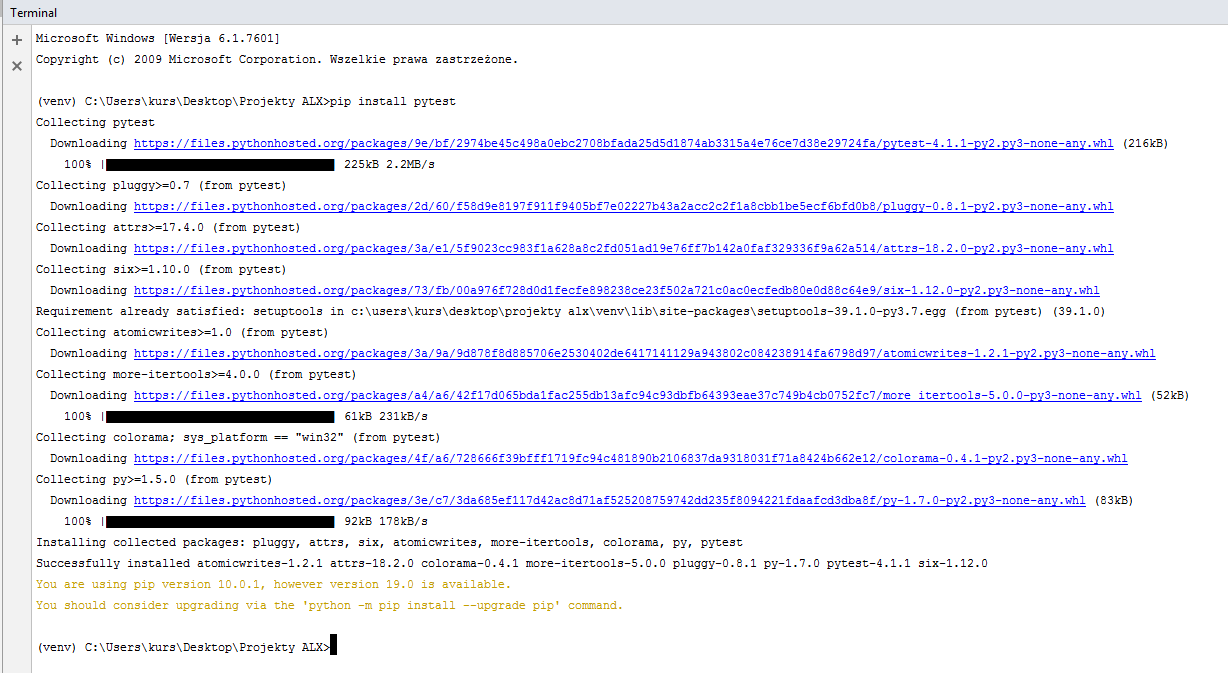
Dodawanie biblioteki pytest:

Wchodzimy do terminala na dole strony i wpisujemy na końcu ścieżki: „pip install pytest”.

Dopisujemy to do „(venv) C:\Users\kurs\Desktop\Projekty ALX> pip install pytest”, reszta będzie już wpisana. I wciskamy Enter



Pojawi się takie coś:



”””

Microsoft Windows [Wersja 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. Wszelkie prawa zastrzeżone.

(venv) C:\Users\kurs\Desktop\Projekty ALX>pip install pytest

Collecting pytest

Downloading https://files.pythonhosted.org/packages/9e/bf/2974be45c498a0ebc2708bfada25d5d1874ab3315a4e76ce7d38e29724fa/pytest-4.1.1-py2.py3-none-any.whl (216kB)

100% |████████████████████████████████| 225kB 2.2MB/s

Collecting pluggy>=0.7 (from pytest)

Downloading https://files.pythonhosted.org/packages/2d/60/f58d9e8197f911f9405bf7e02227b43a2acc2c2f1a8cbb1be5ecf6bfd0b8/pluggy-0.8.1-py2.py3-none-any.whl

Collecting attrs>=17.4.0 (from pytest)

Downloading https://files.pythonhosted.org/packages/3a/e1/5f9023cc983f1a628a8c2fd051ad19e76ff7b142a0faf329336f9a62a514/attrs-18.2.0-py2.py3-none-any.whl

Collecting six>=1.10.0 (from pytest)

Downloading https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1fecfe898238ce23f502a721c0ac0ecfedb80e0d88c64e9/six-1.12.0-py2.py3-none-any.whl

Requirement already satisfied: setuptools in c:\users\kurs\desktop\projekty alx\venv\lib\site-packages\setuptools-39.1.0-py3.7.egg (from pytest) (39.1.0)

Collecting atomicwrites>=1.0 (from pytest)

Downloading https://files.pythonhosted.org/packages/3a/9a/9d878f8d885706e2530402de6417141129a943802c084238914fa6798d97/atomicwrites-1.2.1-py2.py3-none-any.whl

Collecting more-itertools>=4.0.0 (from pytest)

Downloading https://files.pythonhosted.org/packages/a4/a6/42f17d065bda1fac255db13afc94c93dbfb64393eae37c749b4cb0752fc7/more\_itertools-5.0.0-py3-none-any.whl (52kB)

100% |████████████████████████████████| 61kB 231kB/s

Collecting colorama; sys\_platform == "win32" (from pytest)

Downloading https://files.pythonhosted.org/packages/4f/a6/728666f39bfff1719fc94c481890b2106837da9318031f71a8424b662e12/colorama-0.4.1-py2.py3-none-any.whl

Collecting py>=1.5.0 (from pytest)

Downloading https://files.pythonhosted.org/packages/3e/c7/3da685ef117d42ac8d71af525208759742dd235f8094221fdaafcd3dba8f/py-1.7.0-py2.py3-none-any.whl (83kB)

100% |████████████████████████████████| 92kB 178kB/s

Installing collected packages: pluggy, attrs, six, atomicwrites, more-itertools, colorama, py, pytest

Successfully installed atomicwrites-1.2.1 attrs-18.2.0 colorama-0.4.1 more-itertools-5.0.0 pluggy-0.8.1 py-1.7.0 pytest-4.1.1 six-1.12.0

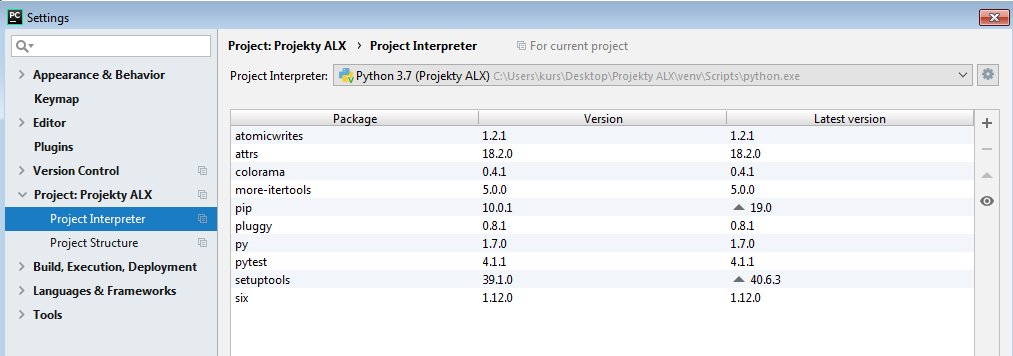
You are using pip version 10.0.1, however version 19.0 is available.

You should consider upgrading via the 'python -m pip install --upgrade pip' command.

(venv) C:\Users\kurs\Desktop\Projekty ALX>

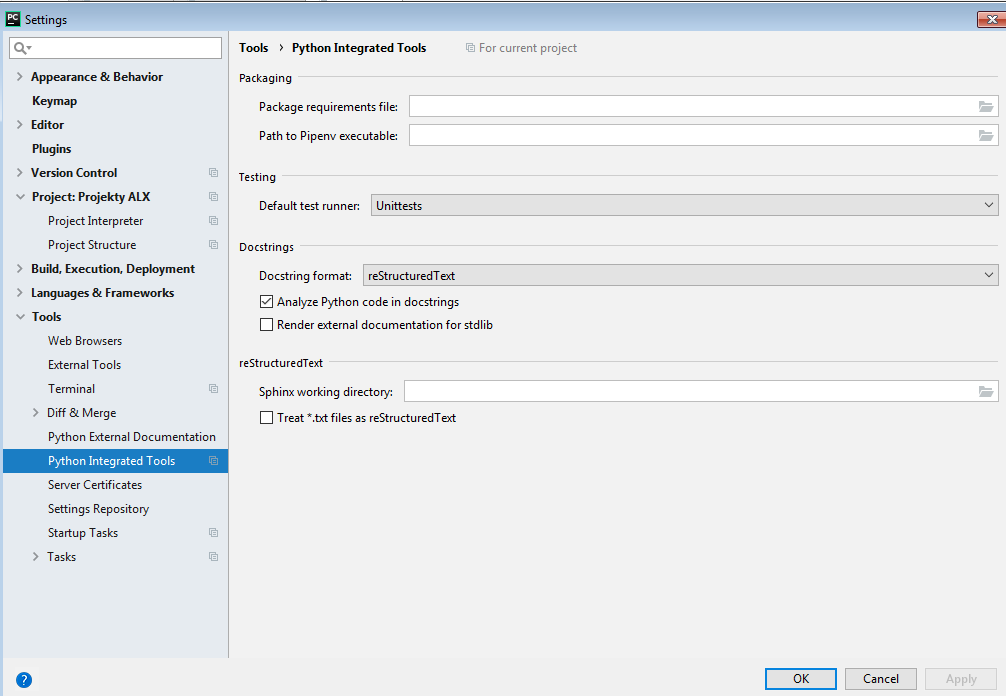
”””

Jeśli nie udało się zainstalować przez „pip…” to idziemy do File -> Settings:

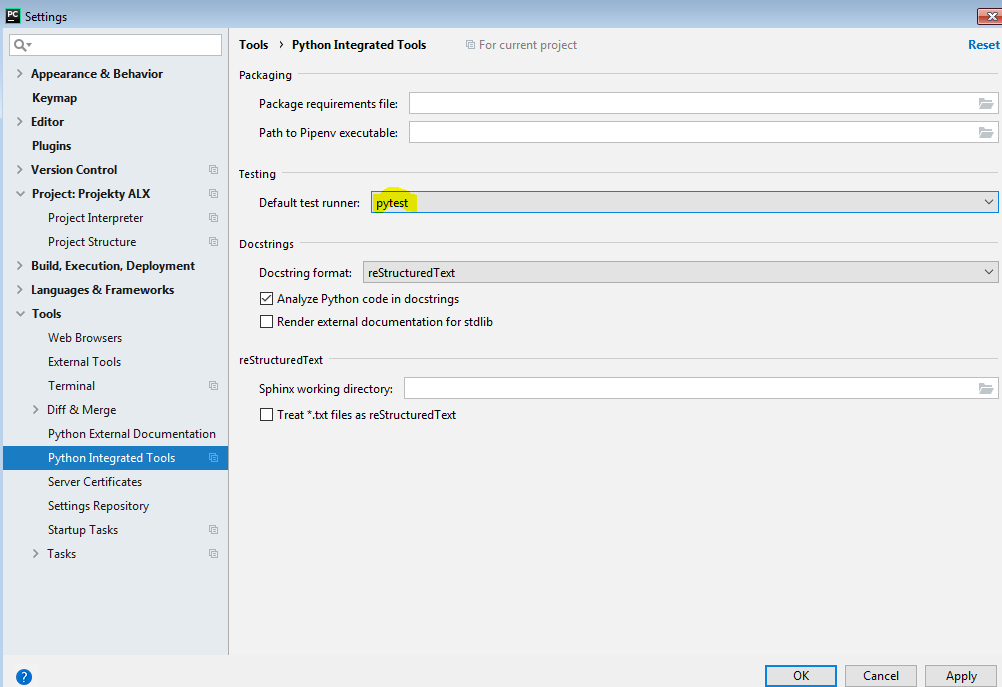


Nie będzie pytest, więc musimy …

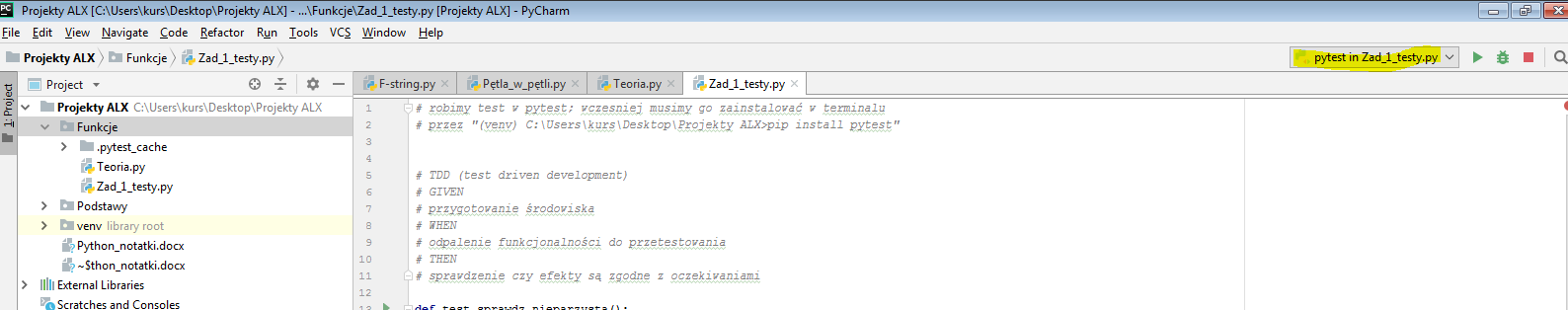
Później robimy:



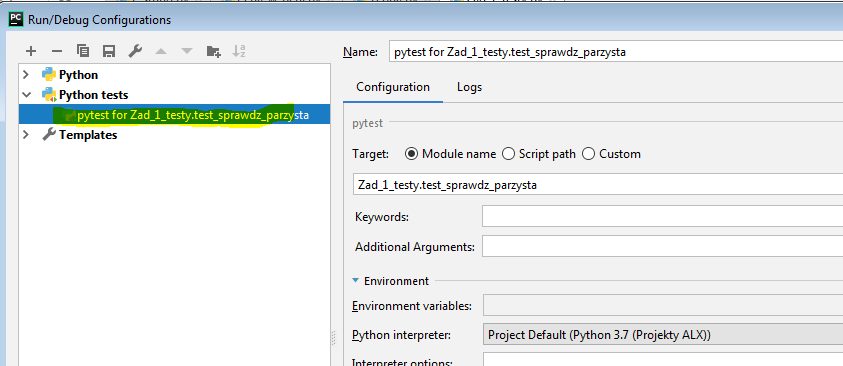
Zmieniamy na



Tu możemy kasować historię pytest:



Dajemy „Edit..”



Kasujemy wszystko co jest pod „Python tests”, kasujemy minusem historię

Stajemy na czerwonym i dajemy lewy Alt + Enter

**def** test\_sprawdz\_nieparzysta():  
 dane\_wejsciowe = 9  
 wynik = czy\_parzysta(dane\_wejsciowe)  
 **assert True** == wynik *# assert sprawdza czy wynik jest zgodny z oczekiwaniami, po lewej stronie jest to co oczekujemy  
 # assert ma podobna funkcjonalność***def** test\_sprawdz\_parzysta():  
 dane\_wejsciowe = 12  
 wynik = czy\_parzysta(dane\_wejsciowe)  
 **assert True** == wynik

i teraz na dole patrzymy czy wyszło, jeśli nie, to co wyszło. Tutaj wyszło tak:

Testing started at 11:48 ...

"C:\Users\kurs\Desktop\Projekty ALX\venv\Scripts\python.exe" "C:\Program Files\JetBrains\PyCharm Community Edition 2018.2.4\helpers\pycharm\\_jb\_pytest\_runner.py" --path "C:/Users/kurs/Desktop/Projekty ALX/Funkcje/Zad\_1\_testy.py"

Launching pytest with arguments C:/Users/kurs/Desktop/Projekty ALX/Funkcje/Zad\_1\_testy.py in C:\Users\kurs\Desktop\Projekty ALX\Funkcje

============================= test session starts =============================

platform win32 -- Python 3.7.0, pytest-4.1.1, py-1.7.0, pluggy-0.8.1

rootdir: C:\Users\kurs\Desktop\Projekty ALX\Funkcje, inifile:collected 2 items

Zad\_1\_testy.py F

Zad\_1\_testy.py:13 (test\_sprawdz\_nieparzysta)

False != True

Expected :True

Actual :False

<Click to see difference>

**def test\_sprawdz\_nieparzysta():**

**dane\_wejsciowe = 9**

**wynik = czy\_parzysta(dane\_wejsciowe)**

**> assert True == wynik # assert sprawdza czy wynik jest zgodny z oczekiwaniami**

**E assert True == False**

Zad\_1\_testy.py:17: AssertionError

. [100%]

================================== FAILURES ===================================

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ test\_sprawdz\_nieparzysta \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

def test\_sprawdz\_nieparzysta():

dane\_wejsciowe = 9

wynik = czy\_parzysta(dane\_wejsciowe)

> assert True == wynik # assert sprawdza czy wynik jest zgodny z oczekiwaniami

E assert True == False

Zad\_1\_testy.py:17: AssertionError

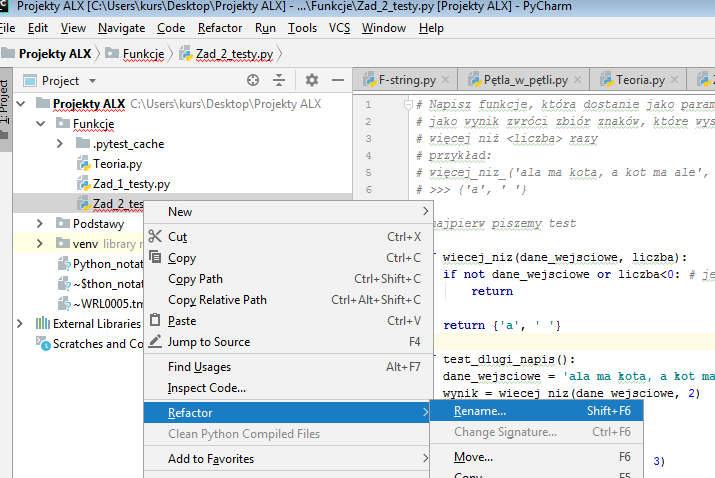
===================== 1 failed, 1 passed in 0.05 seconds ======================

Process finished with exit code 0

Test to jest specyfikacja dla projektu.

W jednym teście najlepiej dać 1 „assert”, bo jak się wykrzaczy, to będzie od razu wiadomo co

Zmiana nazwy pliku:



Zad.2:

Propozycja rozwiązania – pojawia się RunTimeError

*"""  
najpierw piszemy test, później zmieniamy funkcję jesli coś nie wyszło  
Najpierw było tak:  
def wiecej\_niz(dane\_wejsciowe, liczba):  
 if not dane\_wejsciowe or liczba<0: # jeśli nie ma danych wejściowych, to wychodzimy  
 return  
  
 return {'a', ' '}  
"""***def** wiecej\_niz(dane\_wejsciowe, liczba):  
 **if not** dane\_wejsciowe **or** liczba<0: *# jeśli nie ma danych wejściowych, to wychodzimy* **return***# jak sprawdzić ile razy występuje dana literka w tekście* wynik = dict() *# słownik* **for** znak **in** dane\_wejsciowe:  
 wynik[znak] = dane\_wejsciowe.count(znak) *# wstawianie znaku* **for** key **in** wynik:  
 **if** wynik[key]<= liczba:  
 **del** (wynik[key])  
  
 **return** set(wynik.keys()) *# wyciaganie ze słownika***def** test\_dlugi\_napis():  
 dane\_wejsciowe = **'ala ma kota, a kot ma ale'** wynik = wiecej\_niz(dane\_wejsciowe, 2)  
 **assert** {**'a'**, **' '**} == wynik  
  
**def** test\_pusty\_napis():  
 **assert None** == wiecej\_niz(**''**, 3) *# wynikiem będzie None***def** test\_liczba\_duza():  
 **assert** set() == wiecej\_niz(**'ala ma kota'**, 10) *# pusty słownik to {}, pusty zbiór to set()***def** test\_liczba\_ujemna():  
 **assert None** == wiecej\_niz(**'ala ma kota'**, -10) *# jesli liczba będzie ujemna, to wynikiem bedzie pusty zbiór  
 # zamiasta None wcześniej było set()***def** test\_liczba\_zero():  
 **assert** {**'a'**, **'l'**, **' '**, **'m'**, **'k'**, **'o'**, **'t'**} == wiecej\_niz(**'ala ma kota'**, 0)

**Dobre rozwiązanie powyższego rozwiązania:**

**def** wiecej\_niz(dane\_wejsciowe, liczba):  
 **if not** dane\_wejsciowe **or** liczba<0:  
 **return** wynik = set()  
 **for** znak **in** dane\_wejsciowe:  
 **if** dane\_wejsciowe.count(znak)>liczba:  
 wynik.add(znak)  
 **return** wynik  
  
  
**def** test\_dlugi\_napis():  
 dane\_wejsciowe = **'ala ma kota, a kot ma ale'** wynik = wiecej\_niz(dane\_wejsciowe, 2)  
 **assert** {**'a'**, **' '**} == wynik  
  
**def** test\_pusty\_napis():  
 **assert None** == wiecej\_niz(**''**, 3) *# wynikiem będzie None***def** test\_liczba\_duza():  
 **assert** set() == wiecej\_niz(**'ala ma kota'**, 10) *# pusty słownik to {}, pusty zbiór to set()***def** test\_liczba\_ujemna():  
 **assert None** == wiecej\_niz(**'ala ma kota'**, -10) *# jesli liczba będzie ujemna, to wynikiem bedzie pusty zbiór  
 # zamiasta None wcześniej było set()***def** test\_liczba\_zero():  
 **assert** {**'a'**, **'l'**, **' '**, **'m'**, **'k'**, **'o'**, **'t'**} == wiecej\_niz(**'ala ma kota'**, 0)

dobre rozwiązanie skopiowane ze Slacka od wykładowcy:

# napisz funkcję, która dostanie jako parametr napis i liczbę

# jako wynik zwróci zbiór znaków, które wystepują w napisie

# wiecej niz <liczba> razy

# przykład:

# wiecej\_niz('ala ma kota, a kot ma ale', 2)

# >>> {'a',' '}

# rozwiązanie zadania

def wiecej\_niz(dane\_wejsciowe, liczba):

if not dane\_wejsciowe or liczba < 0:

return

wynik = set()

for znak in dane\_wejsciowe:

if dane\_wejsciowe.count(znak) > liczba:

wynik.add(znak)

return wynik

# testy sprawdzajace czy zadanie jest poprawnie wykonane

def test\_dlugi\_napis():

dane\_wejsciowe = 'ala ma kota, a kot ma ale'

wynik = wiecej\_niz(dane\_wejsciowe, 2)

assert {' ','a'} == wynik

def test\_pusty\_napis():

assert None == wiecej\_niz('', 3)

# spodziewam się pustego zbioru

def test\_liczba\_duza():

assert set() == wiecej\_niz('ala ma kota', 10)

def test\_liczba\_ujemna():

assert None == wiecej\_niz('ala ma kota', -10)

def test\_liczba\_zero():

assert {'a', 'l', ' ', 'm', 'k', 'o', 't'} == wiecej\_niz('ala ma kota', 0)

Zad. 3

Początkowe rozwiązanie – z index

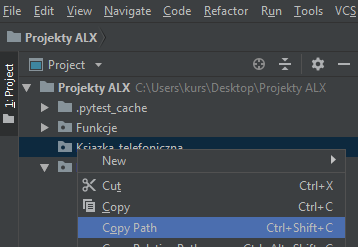
*#Zadanie #3  
# Napisz funkcję obliczającą liczbę znaków w zadanym napisie  
# pomiędzy zadanymi znakami.  
# Znaki, pomiędzy którymi ma odbywać się zliczanie,  
# powinny być argumentami z wartościami domyślnymi odpowiednio < i >.  
# Nawiasy mogą być zagnieżdżone i mogą wystąpić wiele razy.  
# Znaki pomiędzy zagnieżdżonymi nawiasami liczone są  
# zgodnie z poziomem zagnieżdżenia.  
#  
# Przykład użycia:  
# >>> policz\_znaki('ala ma <kota> a kot ma ale')  
# 4  
# >>> policz\_znaki('ala [kota [a kot]] ma [ale]', '[', ']')  
# 18  
# >>> policz\_znaki('a <a<a<a>>>')  
# 6***import** pytest  
  
  
**def** policz\_znaki(napis):  
 gdzie\_nawias\_lewy = napis.index(**"<"**) *# wcześniej było "gdzie\_nawias\_lewy = napis.find("<")", ale find nie wyrzuca informacji o tym, gdzie mamy blad* gdzie\_nawias\_prawy = napis.index(**">"**)  
  
 print(gdzie\_nawias\_lewy, gdzie\_nawias\_prawy)  
 **return** gdzie\_nawias\_prawy - gdzie\_nawias\_lewy -1 *# jeśli nie znaleziono w teście, to trzeba jeszcze dać -1  
  
  
# def test\_pojedynczy\_nawias():  
# assert 4 == policz\_znaki('ala ma <kota> a kot ma ale') # algoryt 'ala ma <kota> a kot ma ale' ma zwrócić 4  
#   
# def test\_pusto\_pomiedzy\_nawiasami():  
# assert 0 == policz\_znaki("tralala <> tralala")  
   
# sposob na sprawdzenie czy kawałek kodu rzuca błąd  
# CHCEMY ŻEBY RZUCIŁ BŁĄD***def** test\_brakuje\_nawiasu():  
 **with** pytest.raises(ValueError):  
 policz\_znaki(**"dkasuhgliuacvrtuie"**) *# nie znalazł.......*

Kolejne etapy, duża część została mieniona w środku kodu:

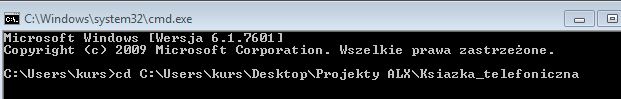
**def** policz\_znaki(napis):  
 gdzie\_nawias\_lewy = napis.find(**'<'**)  
 gdzie\_nawias\_prawy = napis.find(**'>'**)  
 **if** gdzie\_nawias\_lewy == -1 **and** gdzie\_nawias\_prawy == -1:  
 **return** 0  
 **if** gdzie\_nawias\_lewy == -1 **or** gdzie\_nawias\_prawy == -1:  
 **return None  
 if** gdzie\_nawias\_prawy - gdzie\_nawias\_lewy < 0:  
 **return None  
 return** gdzie\_nawias\_prawy - gdzie\_nawias\_lewy -1  
  
**def** test\_pojedynczy\_nawias():  
 **assert** 4 == policz\_znaki(**'ala ma <kota> a kot ma ale'**)  
  
**def** test\_pusto\_pomiedzy\_nawiasami():  
 **assert** 0 == policz\_znaki(**'tralala <> tralala'**)*# # sposób na sprawdzenie czy kawałek kodu rzuca błąd  
  
  
# def test\_pojedynczy\_nawias():  
# assert 4 == policz\_znaki('ala ma <kota> a kot ma ale') # algoryt 'ala ma <kota> a kot ma ale' ma zwrócić 4  
#  
# def test\_pusto\_pomiedzy\_nawiasami():  
# assert 0 == policz\_znaki("tralala <> tralala")  
  
# sposob na sprawdzenie czy kawałek kodu rzuca błąd  
# CHCEMY ŻEBY RZUCIŁ BŁĄD  
# def test\_brakuje\_nawiasu():  
# with pytest.raises(ValueError):  
# policz\_znaki("dkasuhgliuacvrtuie") # nie znalazł.......***def** test\_bez\_nawiasow():  
 **assert** 0 == policz\_znaki(**"dkfadfjlkdfldskf"**)  
  
**def** test\_nawiasy\_w\_zlej\_kolejnosci():  
 **assert None** == policz\_znaki(**"dkfadf>j<lkdfldskf"**)

**DJANGO**

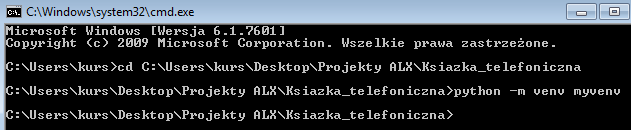
Kopiujemy ścieżkę naszego katalogu w PyCharm

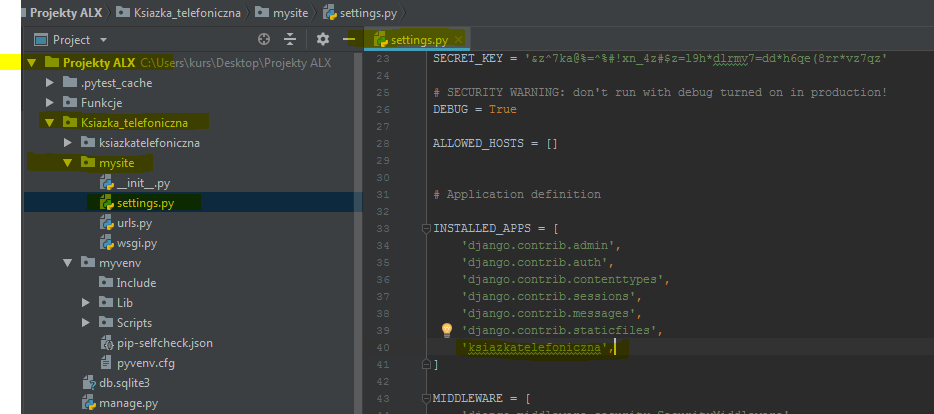


I wklejamy do cdm:

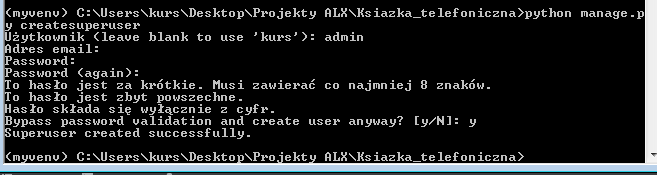


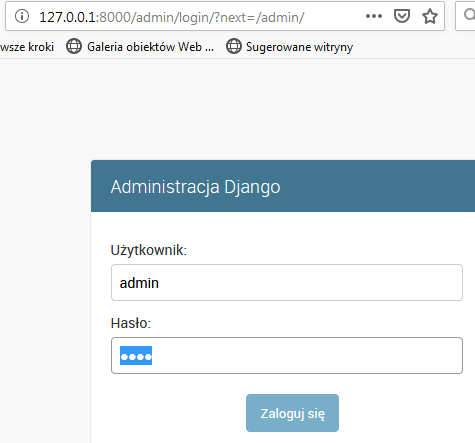
Wpisujemy komendę:

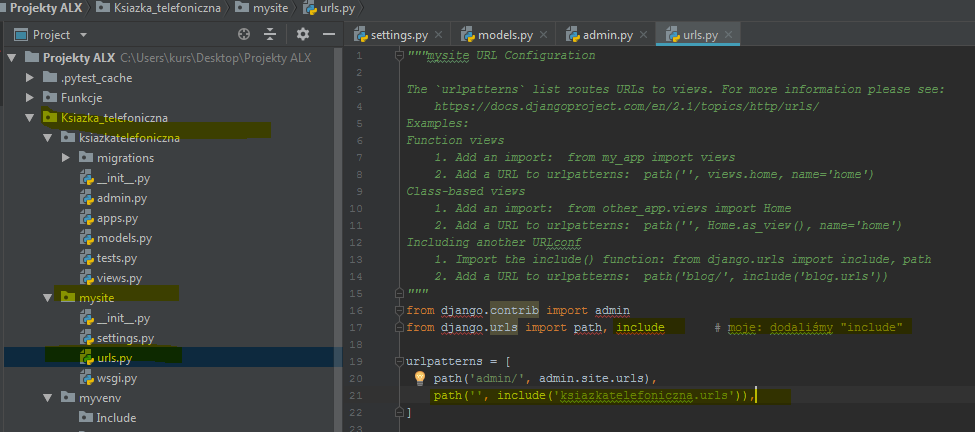




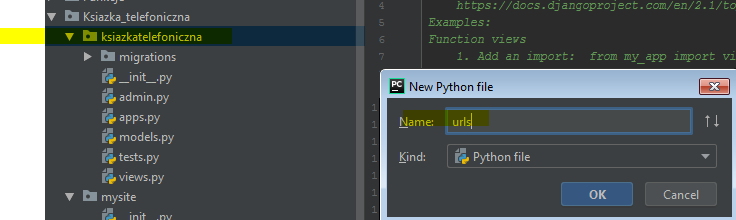
Tworzenie usera:

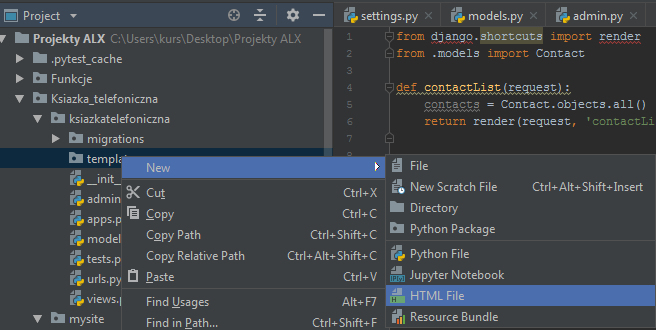


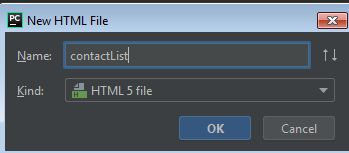




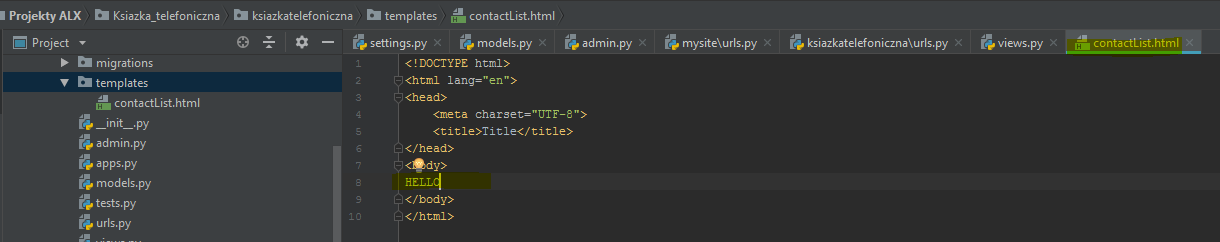
Tworzymy plik pythona o nazwie „urls”





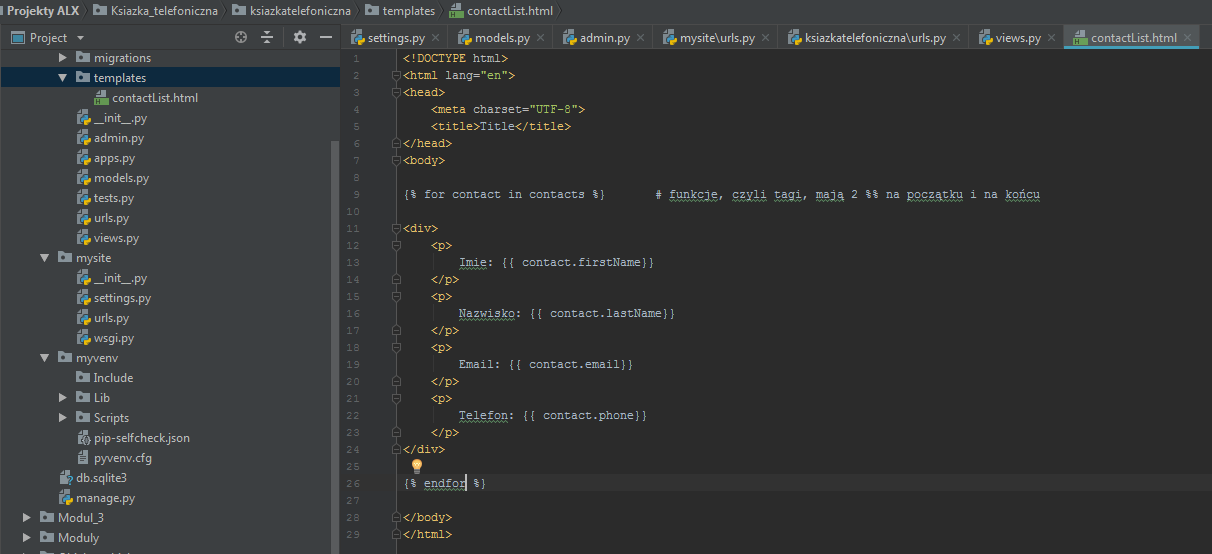


Jak wpiszemy HELLO, to wyświetli się n stronie

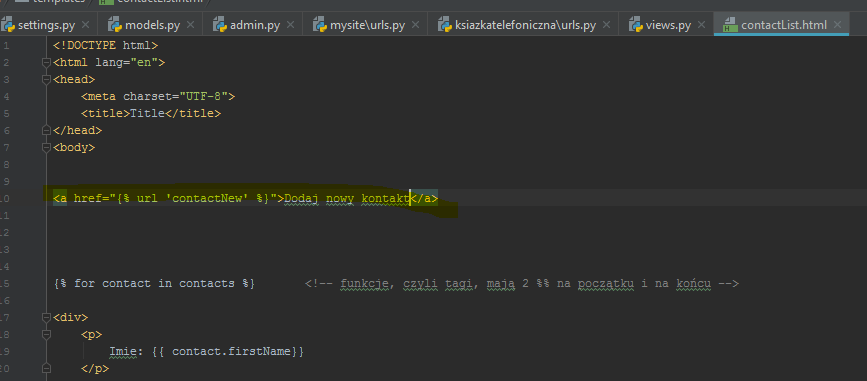


Dodajemy szablon (pętle). Ten HTML jest przetwarzany przez django (bo w HTML to język znaczników, nie pogramowania, czyli nie ma pętli, zmiennych – dlatego renderujemy go

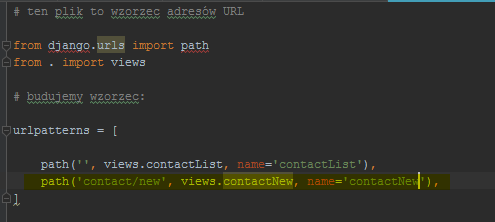
HTML traktowany jest jak szablon



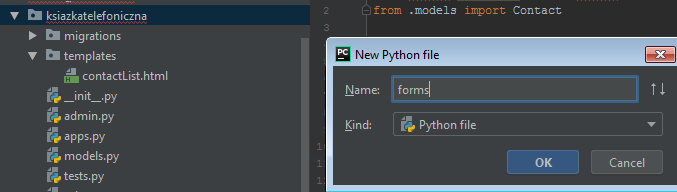
Tworzymy link:



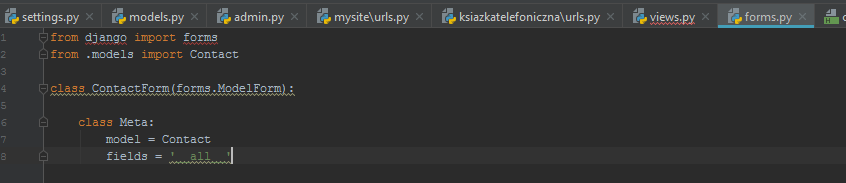
Teraz idziemy do „urls” i definiujemy



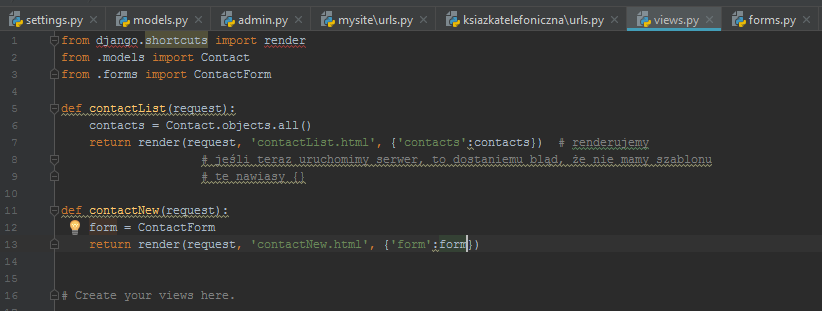
Tworzymy plik pythona „forms”:



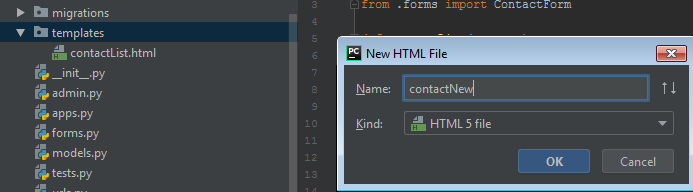
I piszemy:

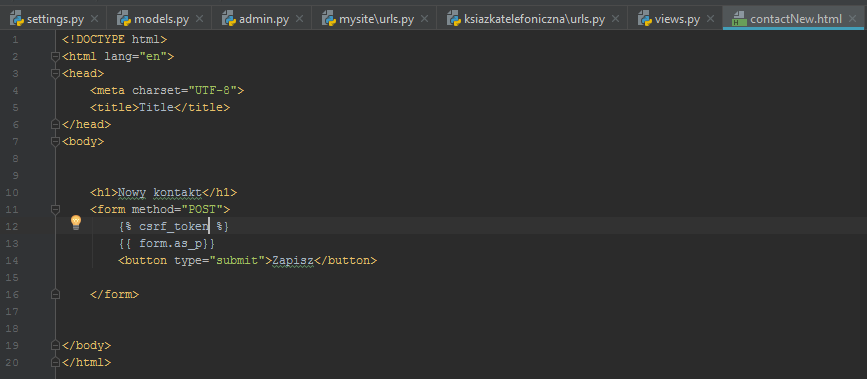


Idziemy do „views” i tworzymy nową funkcję:



Tworzymy html nowy





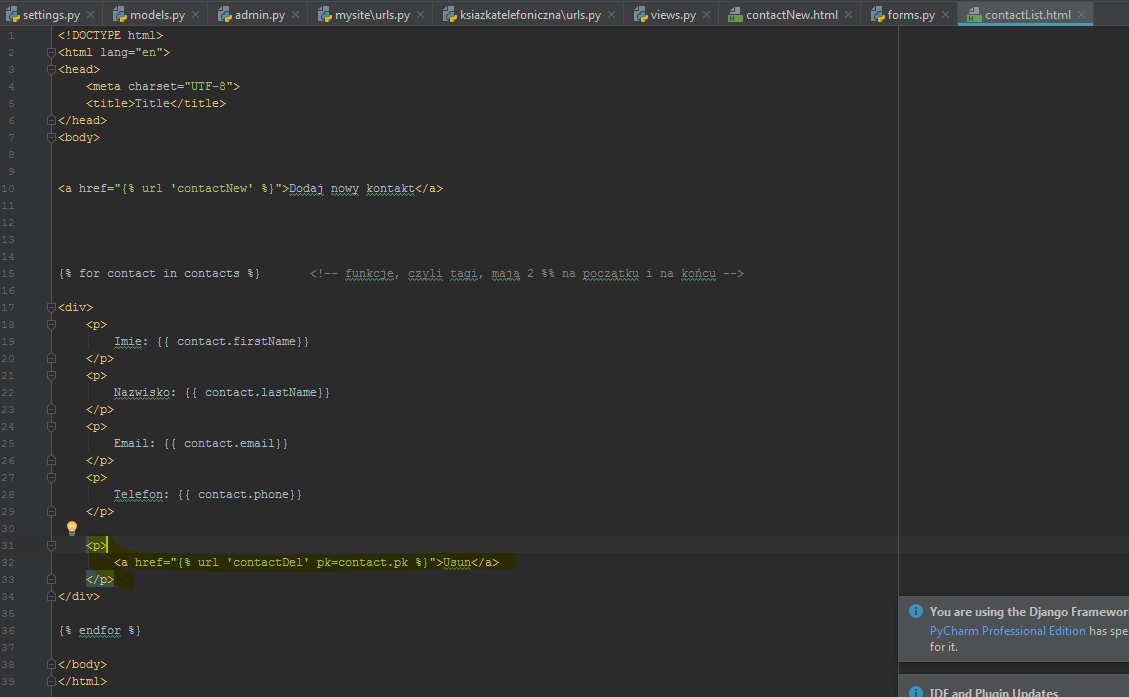
Csrf to ochrona

Teraz robimy if aby zapisywać wprowadzane dane

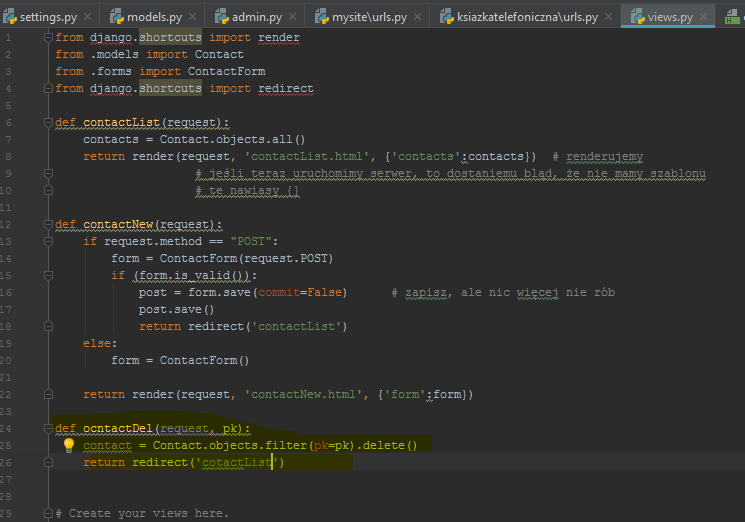
Teraz nasza strona wygląda tak:



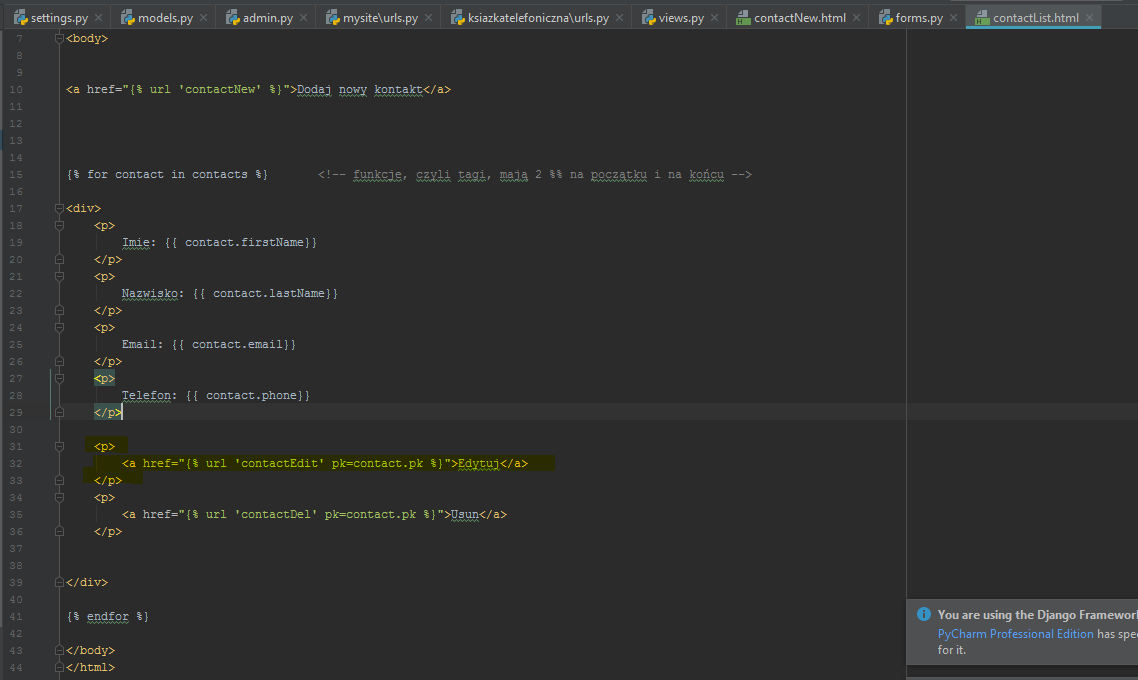
Dopisujemy



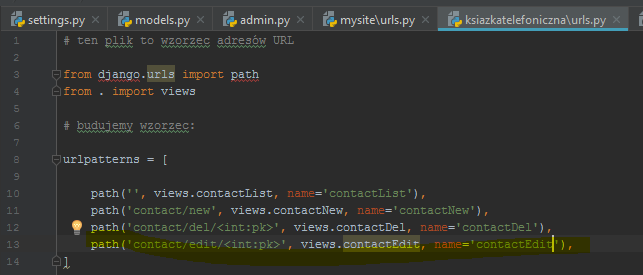
Dopisujemy:



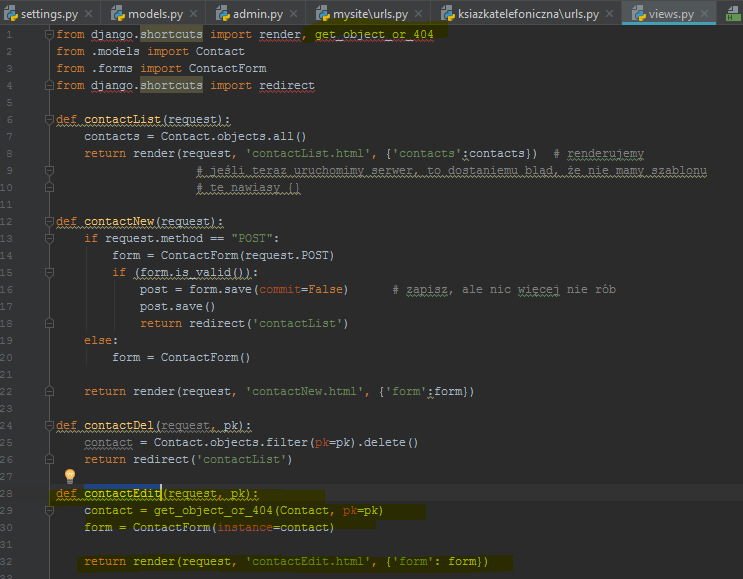
Robimy opcje edycji:



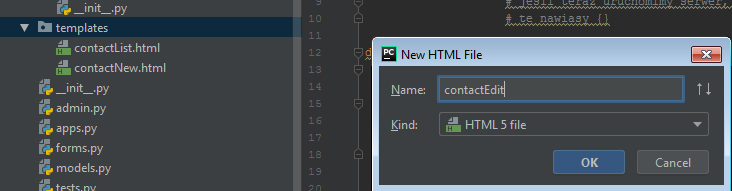
Następnie:



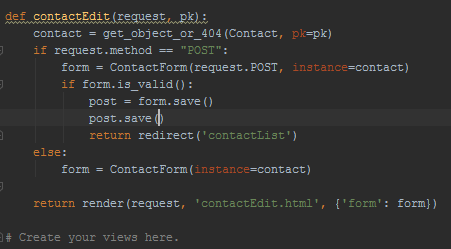
Idziemy do view:

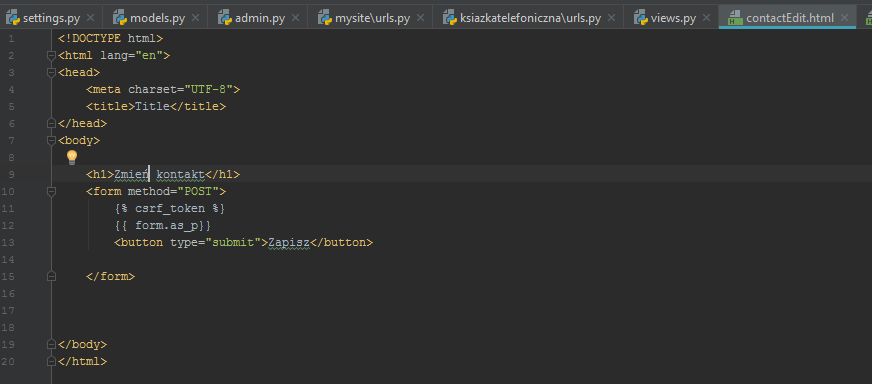


Tworzymy nowy HTML:



I wstawiamy prawie to samo, co w contactNew:





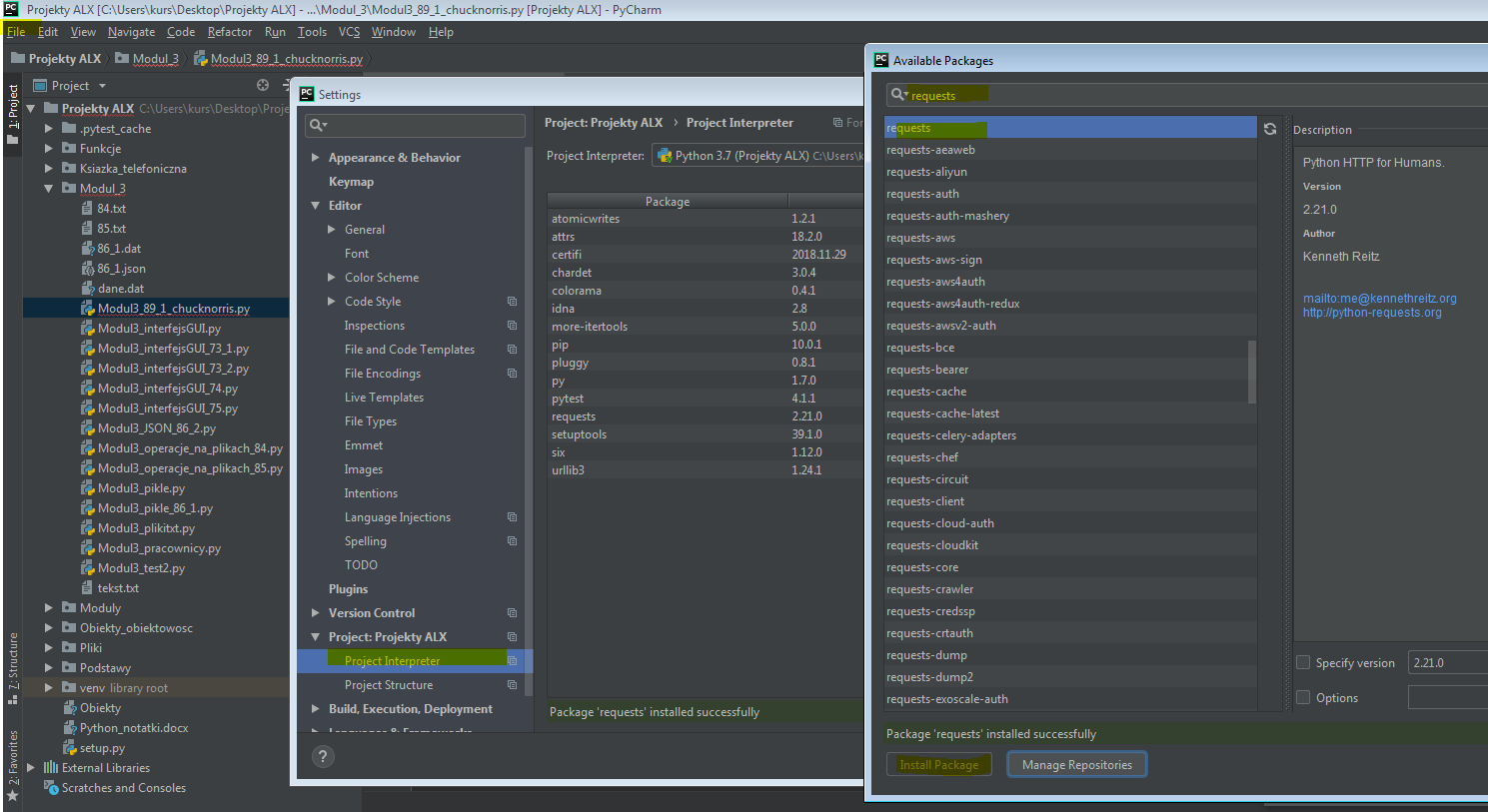
**HTML**

Metoda – sposób obsługi formularza

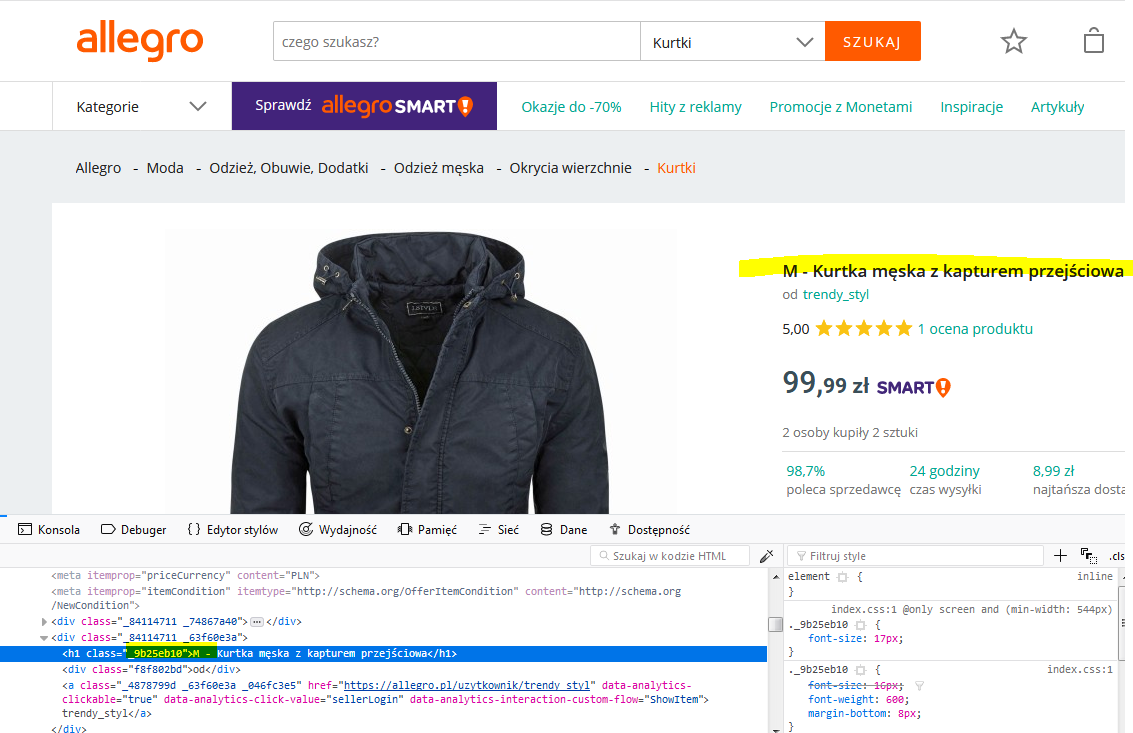
Metoda GET – w adresie url będzie słowo, którego szukamy

Metoda POST – nie wysyła danych do adresu url (np. w przypadku logowania do banku), inny sposób przekazywania danych

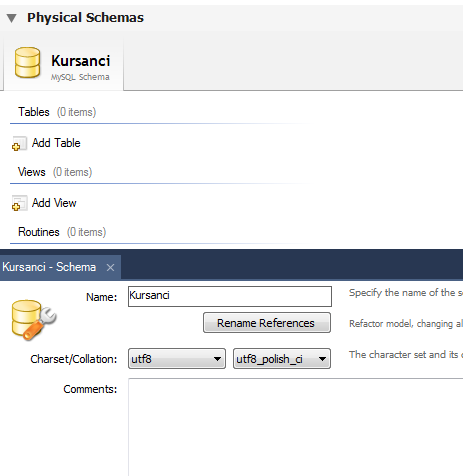
Instalowanie requests:

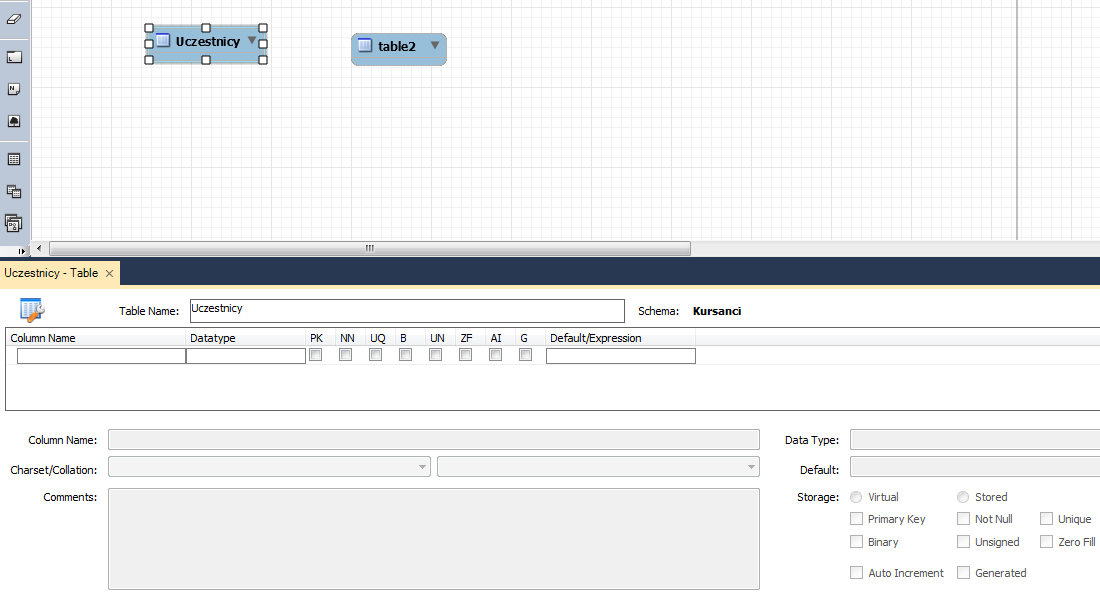


Wyciąganie nazwy z allegro:

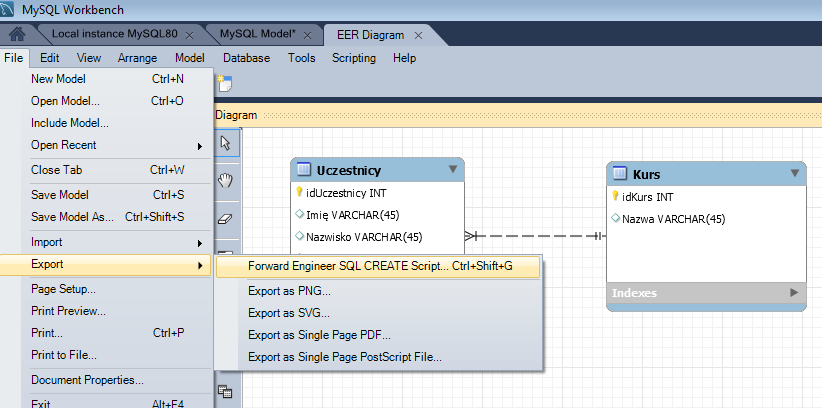


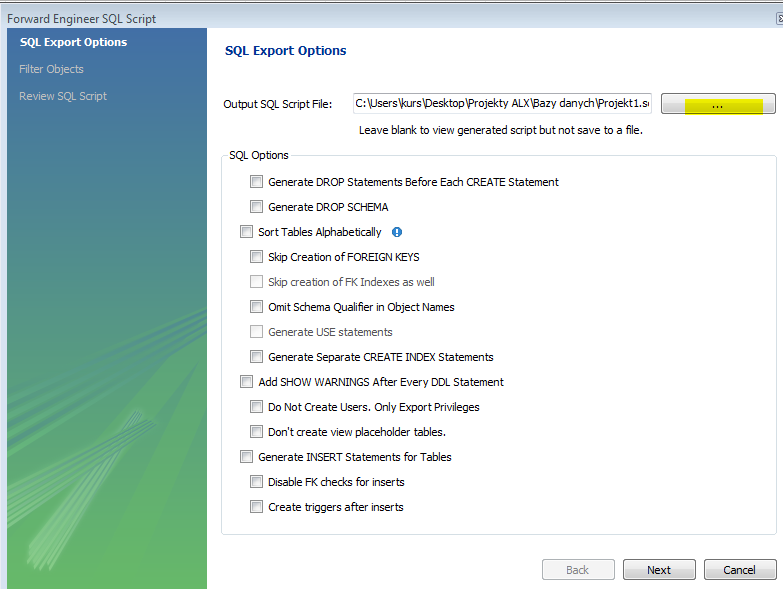
MySQL



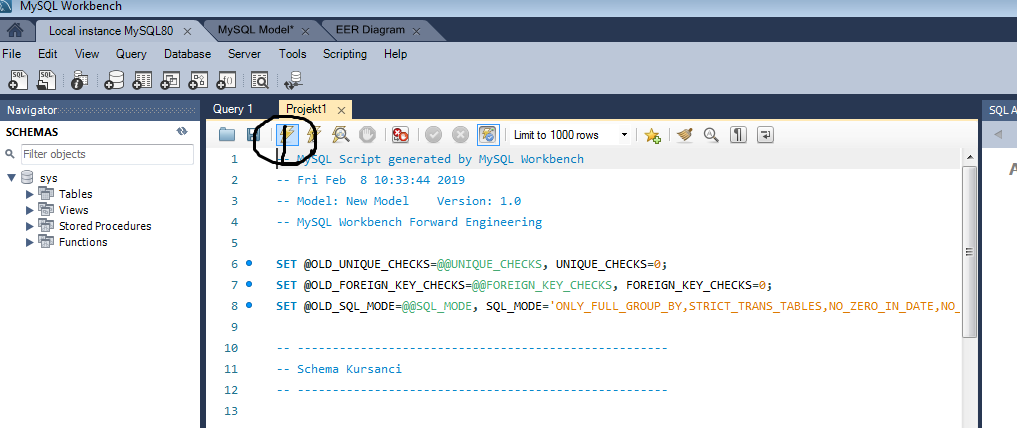


AI – autonumerowanie

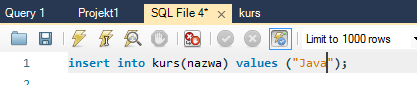




Next next finisz



Wstawianie do bazy:

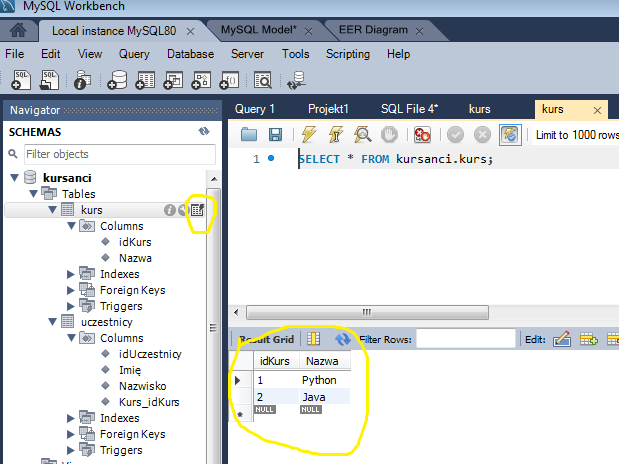


Wpisujemy i klikamy błyskawicę:

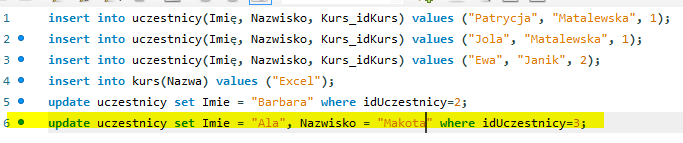


Druga błyskawica, to wstawia tylko podświetlony rekord:

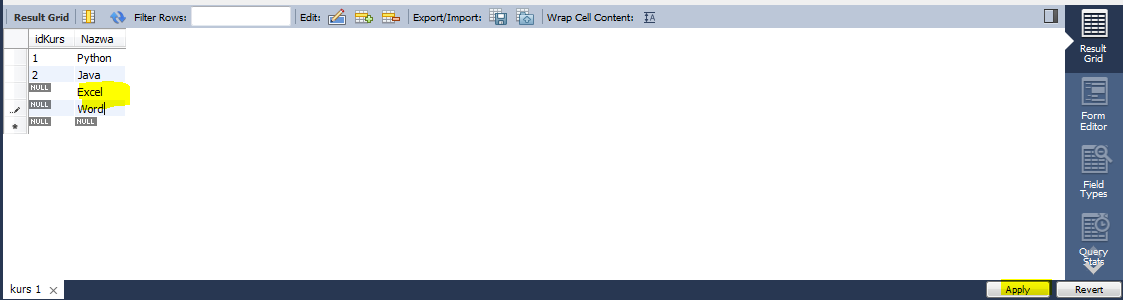
Wyświetlanie wyniku:



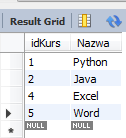
**Aktualizacja rekordów:**



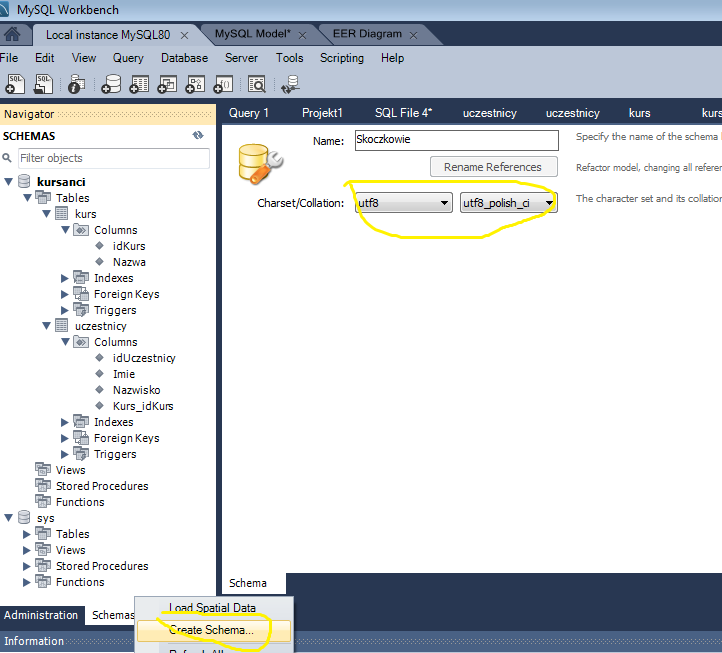
Dodawanie z poziomu graficznego interfejsu:



I pojawi się:



Nowy schemat:



Adres do ściągnięcia danych wykorzystwanych na zajęciach:

<http://www.kmguxim.pl/baza.zip>

select \* from zawodnicy

where id\_skoczka=3;

zadanie: wielkość druzyn narodowych:

select count(\*), kraj from zawodnicy

group by kraj;

zadanie: największy wzrost i segregacja po krajach

select kraj, max(wzrost) as mw from zawodnicy

group by kraj

order by mw desc;

import danych z Excela do bazy:

