
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki Zakład Systemów Teleinformatycznych		
Przedmiot	Skryptowe języki programowania		
Prowadzący	mgr inż. Martyna Tarczewska		
Temat	Python - wprowadzenie		
Student	Paweł Jońca		
Nr lab.	3	Data wykonania	28.10
Ocena		Data oddania spr.	28.10

Zad 1.

Nie udało się wykonać wszystkich operacji, ponieważ lista zawiera różne typy danych liczby i tekst, a python nie może porównać tych typów podczas sortowania co powoduje błąd, żeby tego uniknąć trzeba sprawdzić czy lista zawiera elementy tego samego typu

Listy w pythonie są numerowane od 0

```

zadanie.py x
1  def li(): 1 usage
2      list = [] #definiowanie listy
3      print(list)
4      list.extend([6,7,8,9,10])
5      first_two = list[:2]
6      print(first_two)
7      print(list[-2:])
8      print(list)
9      print(len(list))
10     print(list[:2])
11     list.append(50)
12     # list.append("Hej")
13     list.sort()
14     list.pop()
15     list.reverse()
16     list.insert(2, 100)
17     num = list.count(13)
18     print(num)
19     print(list)
20  def main(): 1 usage
21      li()
22
23  if __name__ == '__main__':
24      main()

```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skrzytowe Języki programowania\pythonProject1\zadanie.py"
Traceback (most recent call last):
  File "C:\Users\pawel\Dokumenty\03 - studia\Skrzytowe Języki programowania\pythonProject1\zadanie.py", line 27, in <module>
    main()
  File "C:\Users\pawel\Dokumenty\03 - studia\Skrzytowe Języki programowania\pythonProject1\zadanie.py", line 24, in main
    li()
  File "C:\Users\pawel\Dokumenty\03 - studia\Skrzytowe Języki programowania\pythonProject1\zadanie.py", line 16, in li
    list.sort()
TypeError: '<' not supported between instances of 'str' and 'int'
[]
[6, 7]
[9, 10]
[6, 7, 8, 9, 10]
5
[6, 8, 10]

Process finished with exit code 1
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skrzytowe Języki programowania\pythonProject1\zadanie.py"
[]
[6, 7]
[9, 10]
[6, 7, 8, 9, 10]
5
[6, 8, 10]
0
[10, 9, 100, 8, 7, 6]

Process finished with exit code 0
```

Zad 2

```
zadanie2.py x zadanie.py
1 def new_list(): 1 usage Pawelgabrieljonca *
2     numbers = []
3     while len(numbers) < 10:
4         try:
5             num = float(input("Write number(left {} to write: ).format(10 - len(numbers))))
6             numbers.append(num)
7         except ValueError:
8             print("Failed, give number")
9
10    numbers.sort()
11    min_number = numbers[0]
12    max_number = numbers[-1]
13    #calculating the average
14    no_negative_numbers = [num for num in numbers if num >= 0]
15    average_negative = sum(no_negative_numbers) / len(no_negative_numbers) if no_negative_numbers else 0
16
17    print("Numbers:", numbers)
18    print("Min:", min_number)
19    print("Max:", max_number)
20    print("Average:", average_negative)
21
22 def main(): 1 usage Pawelgabrieljonca *
23     new_list()
24
25 if __name__ == '__main__':
26     main()
27
```

```

"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\zadanie2.py"
Write number(left 10 to write: )12
Write number(left 9 to write: )4
Write number(left 8 to write: )-6
Write number(left 7 to write: )12
Write number(left 6 to write: )4
Write number(left 5 to write: )3
Write number(left 4 to write: )2
Write number(left 3 to write: )6
Write number(left 2 to write: )8
Write number(left 1 to write: )9
Numbers: [-6.0, 2.0, 3.0, 4.0, 4.0, 6.0, 8.0, 9.0, 12.0, 12.0]
Min: -6.0
Max: 12.0
Average: 6.666666666666667

Process finished with exit code 0

```

Zad 3

The screenshot shows an IDE with a project named 'pythonProject1'. The file explorer on the left shows a folder named '.venv' containing 'zadanie.py', 'zadanie2.py', and 'zadanie3.py'. The main editor displays the code for 'zadanie3.py'.

```

1
2 def new_list(): 1 usage
3     list1 = [1,2,3,4,5]
4     list2 = [3,4,5,6,7]
5
6     unique_item = [item for item in list1 if item not in list2]
7     print(list1)
8     print(list2)
9     print(unique_item)
10
11
12 def main(): 1 usage
13     new_list()
14
15 if __name__ == '__main__':
16     main()

```

The Run window at the bottom shows the execution output for 'zadanie3':

```

"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\zadanie3.py"
[1, 2]
[1, 2, 3, 4, 5]
[3, 4, 5, 6, 7]

Process finished with exit code 0

```

Zad 4

```
zadanie2.py  zadanie3.py  zadanie4.py x  zadanie.py

1  def new_list(): 1 usage new *
2      list = [1,2,3,4,5,6,7,8,9,0]
3
4      check = [num for num in list if num % 2 != 0]
5      if check:
6          smallest = min(list)
7      else:
8          smallest = None
9
10
11     print("list of numbers",list)
12     print("odd numbers",check)
13     print("Smallest",smallest)
14
15 def main(): 1 usage new *
16     new_list()
17
18 if __name__ == '__main__':
19     main()
```

Run zadanie4 x

↑
↓
⇌
⇌
🖨
🗑

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\SK
list of numbers [1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
odd numbers [1, 3, 5, 7, 9]
Smallest 0

Process finished with exit code 0
```

Zad 5

```
Project ▾
└─ pythonProject1 C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania
    └─ .venv
        ├── zadanie.py
        ├── zadanie2.py
        ├── zadanie3.py
        ├── zadanie4.py
        └─ zadanie5.py
    └─ External Libraries
    └─ Scratches and Consoles

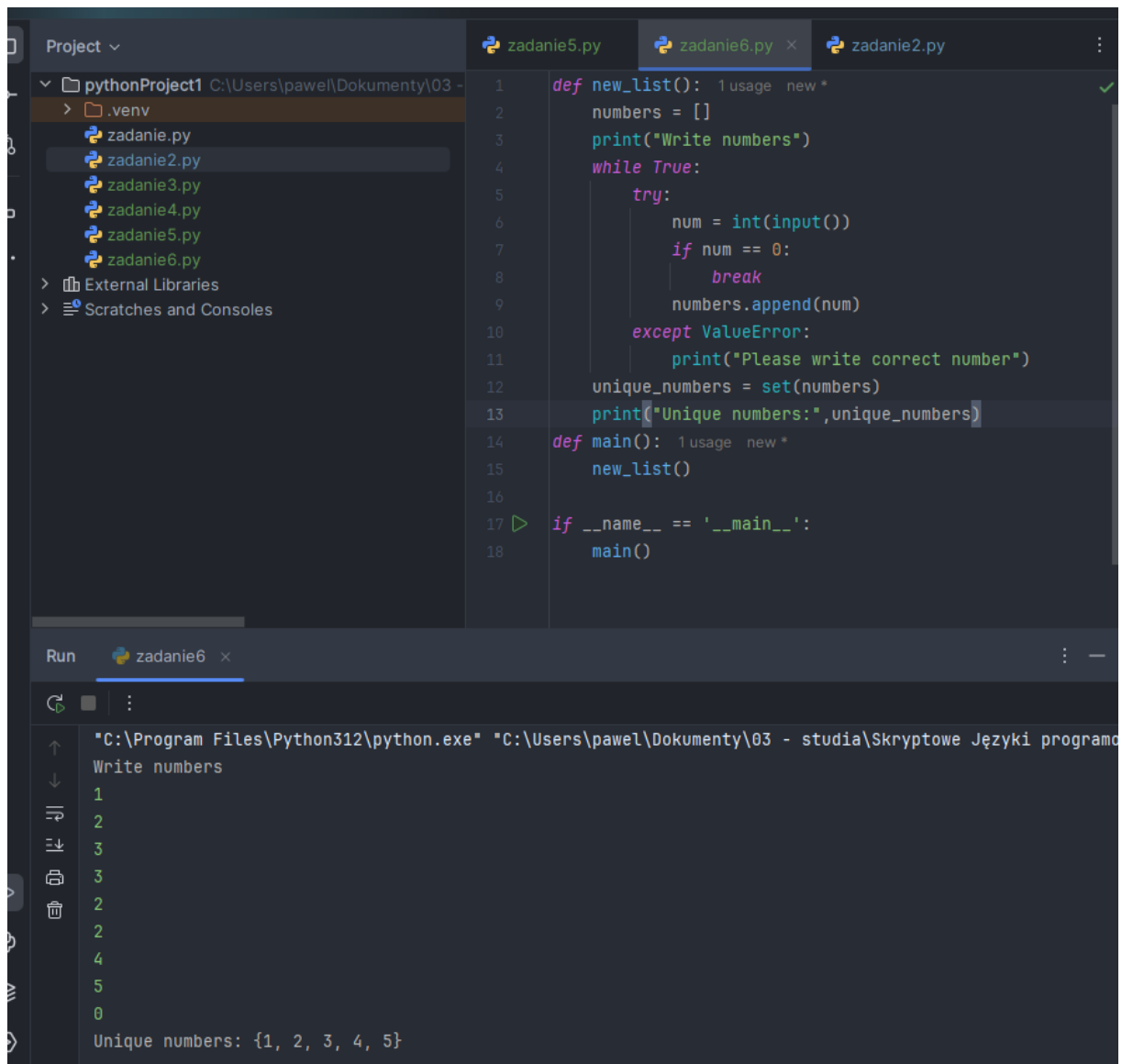
zadanie5.py
1 def new_list(): 1 usage
2     list_A = [1, 2, 3]
3     list_B = ['a', 'b', 'c']
4
5     combined_A_first = list_A + list_B
6     print(combined_A_first)
7     combined_B_first = list_B + list_A
8     print(combined_B_first)
9
10 def main(): 1 usage
11     new_list()
12
13 if __name__ == '__main__':
14     main()
```

Run zadanie5

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie5.py"
[1, 2, 3, 'a', 'b', 'c']
['a', 'b', 'c', 1, 2, 3]
Process finished with exit code 0
```

Można to po prostu zrobić dodając dwie listy do siebie, bo nie wiem czy dobrze zrozumiałem instrukcje z wskazówką.

Zad 6



Zad 7

```
2
3 tuple = ("apple", "banana", "cherry")
4 tuple_b = ("orange",)
5 tuple += tuple_b #dodawanie krotek
6 multi_tuple = tuple * 2 #mnożenie krotek
7 print(len(tuple)) #długość krotki - liczba elementów
8 for x in tuple: #wypisanie wszystkich elementów krotki
9     print(x)
10
```

Run zadanie7 x

"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - st

4
apple
banana
cherry
orange

Process finished with exit code 0

7a) Dodawanie krotek działa na zasadzie łącznie dwóch krotek w jedną

7b) Dodanie tej samej krotki dwukrotnie spowoduje, że elementy krotki zostaną powtórzone, tworząc dłuższą krotkę

7c) Mnożenie krotki przez liczbę całkowitą powoduje powielenie jej elementów określoną liczbę razy

7d) Przecinek w zapisie jest konieczny aby Python rozpoznał to jako krotkę z jednym elementem a nie jako zwykły string

7e) Dla krotek nie można użyć metody .sort(), ponieważ krotki są niemodyfikowalne.

Zad 8

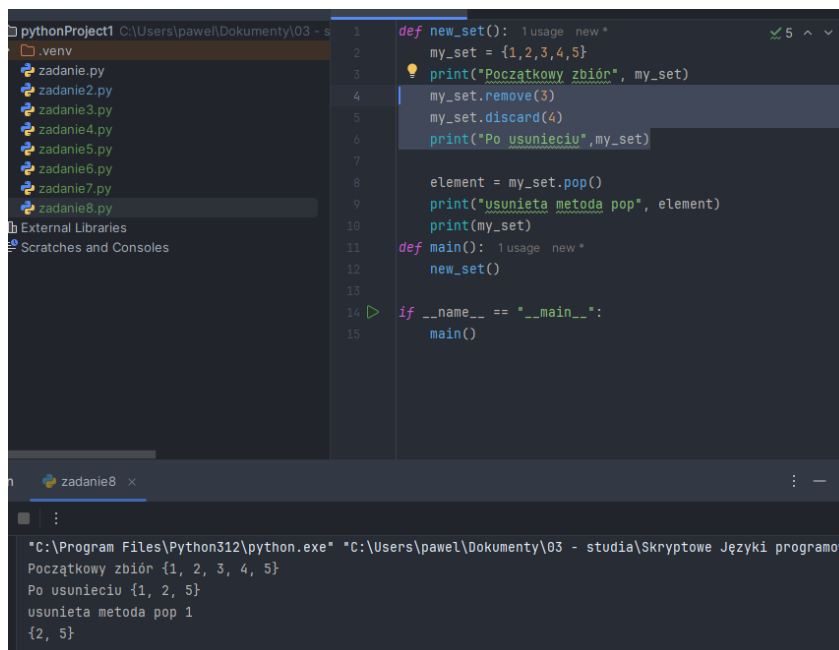
Próba usunięcia nieistniejącego elementu metoda **remove** generuje błąd

discard usuwa wskazany element, ale jeśli element nie istnieje, to po prostu ignoruje operację bez wywoływania błędu.

pop() usuwa i zwraca losowy element ze zbioru, ponieważ zbiór w Pythonie nie jest uporządkowany. Dlatego **pop()** nie zawsze usuwa ten sam element

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie8.py"
Początkowy zbiór {1, 2, 3, 4, 5}
Po usunięciu {1, 2, 4, 5}
Traceback (most recent call last):
  File "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie8.py", line 11, in <module>
    main()
  File "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie8.py", line 10, in main
    new_set()
  File "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie8.py", line 5, in new_set
    my_set.remove(3)
KeyError: 3

Process finished with exit code 1
```



The screenshot shows an IDE with a file explorer on the left containing files named 'zadanie1.py' through 'zadanie8.py'. The main editor displays the following Python code:

```
def new_set():
    my_set = {1,2,3,4,5}
    print("Początkowy zbiór", my_set)
    my_set.remove(3)
    my_set.discard(4)
    print("Po usunięciu", my_set)

    element = my_set.pop()
    print("usunięta metoda pop", element)
    print(my_set)

def main():
    new_set()

if __name__ == "__main__":
    main()
```

Below the editor, a console window titled 'zadanie8' shows the program's output:

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\zadanie8.py"
Początkowy zbiór {1, 2, 3, 4, 5}
Po usunięciu {1, 2, 5}
usunięta metoda pop 1
{2, 5}
```

Zad 9


```

def new_set(): 1 usage
    # Definiowanie dwóch zbiorów
    the_set = {1, 2, 3, 4, 5}
    this_set = {4, 5, 6, 7, 8}

    # a. Sprawdzenie, czy zbioru są rozłączne
    is_disjoint = the_set.isdisjoint(this_set)
    print("a. the_set.isdisjoint(this_set):", is_disjoint, "- typ:", type(is_disjoint))

    # b. Sprawdzenie, czy the_set jest podzbiorem this_set
    is_subset = the_set.issubset(this_set)
    print("b. the_set.issubset(this_set):", is_subset, "- typ:", type(is_subset))

    # c. Sprawdzenie, czy the_set jest nadzbiorem this_set
    is_superset = the_set.issuperset(this_set)
    print("c. the_set.issuperset(this_set):", is_superset, "- typ:", type(is_superset))

    # d. Połączenie obu zbiorów
    union_set = the_set.union(this_set)
    print("d. the_set.union(this_set):", union_set, "- typ:", type(union_set))

    # e. Różnica zbiorów
    difference_set = the_set.difference(this_set)
    print("e. the_set.difference(this_set):", difference_set, "- typ:", type(difference_set))

    # f. Przecięcie zbiorów
    intersection_set = the_set.intersection(this_set)
    print("f. the_set.intersection(this_set):", intersection_set, "- typ:", type(intersection_set))

def main(): 1 usage
    new_set()

if __name__ == "__main__":
    main()

```

```

n zadanie9 x
:
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki
a. the_set.isdisjoint(this_set): False - typ: <class 'bool'>
b. the_set.issubset(this_set): False - typ: <class 'bool'>
c. the_set.issuperset(this_set): False - typ: <class 'bool'>
d. the_set.union(this_set): {1, 2, 3, 4, 5, 6, 7, 8} - typ: <class 'set'>
e. the_set.difference(this_set): {1, 2, 3} - typ: <class 'set'>
f. the_set.intersection(this_set): {4, 5} - typ: <class 'set'>

Process finished with exit code 0

```

Zad 10

zadanie8.py

zadanie10.py ×

```
1  def new_set(): 1 usage new *
2      # Definicja słownika
3      car_dict = {
4          "brand": "Ford",
5          "model": "Mustang",
6          "year": 1964,
7          "new": False
8      }
9      # Pobieranie wartości z klucza 'model'
10     x = car_dict["model"]
11     print("Wartość klucza 'model' za pomocą []:", x)
12     x = car_dict.get("model")
13     print("Wartość klucza 'model' za pomocą get():", x)
14     # Lista kluczy, wartości i par klucz-wartość
15     print("Lista kluczy:", car_dict.keys())
16     print("Lista wartości:", car_dict.values())
17     print("Lista par klucz-wartość:", car_dict.items())
18     # Dodanie nowej pary klucz-wartość
19     car_dict["color"] = "white"
20     print("Po dodaniu nowej pary ('color': 'white'):", car_dict)
21     # Aktualizacja wartości klucza 'year'
22     car_dict["year"] = 1963
23     print("Po aktualizacji wartości klucza 'year' na 1963:", car_dict)
24     # Sprawdzenie, czy klucz 'model' znajduje się w słowniku
25     if "model" in car_dict:
26         print("element found!")
27     # Usunięcie elementu o kluczu 'model'
28     car_dict.pop("model")
29     print("Po usunięciu klucza 'model':", car_dict)
30
31     # Usunięcie ostatnio dodanej pary klucz-wartość
32     car_dict.popitem()
33     print("Po usunięciu ostatniej pary klucz-wartość:", car_dict)
34
35     def main(): 1 usage new *
36         new_set()
37
38     if __name__ == "__main__":
39         main()
```

```
"C:\Program Files\Python312\python.exe" "C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\zadanie10.py"
Wartość klucza 'model' za pomocą []: Mustang
Wartość klucza 'model' za pomocą get(): Mustang
Lista kluczy: dict_keys(['brand', 'model', 'year', 'new'])
Lista wartości: dict_values(['Ford', 'Mustang', 1964, False])
Lista par klucz-wartość: dict_items([('brand', 'Ford'), ('model', 'Mustang'), ('year', 1964), ('new', False)])
Po dodaniu nowej pary ('color': 'white'): {'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'new': False, 'color': 'white'}
Po aktualizacji wartości klucza 'year' na 1963: {'brand': 'Ford', 'model': 'Mustang', 'year': 1963, 'new': False, 'color': 'white'}
element found!
Po usunięciu klucza 'model': {'brand': 'Ford', 'year': 1963, 'new': False, 'color': 'white'}
Po usunięciu ostatniej pary klucz-wartość: {'brand': 'Ford', 'year': 1963, 'new': False}

Process finished with exit code 0
```

Słownik może zawierać klucze o różnych typach

Wartości również mogą mieć różne typy

Zad 11

```
def student_grades(): 5 usages
    # Zagnieżdżony słownik z danymi studentów
    students = {
        12345: {
            "imię": "Paweł",
            "nazwisko": "Kowalski",
            "oceny": [4, 5, 3, 4, 5]
        },
        23456: {
            "imię": "Kacper",
            "nazwisko": "Nowak",
            "oceny": [3, 3, 4, 5, 4]
        },
        34567: {
            "imię": "Artur",
            "nazwisko": "Wiśniewski",
            "oceny": [5, 4, 4, 5, 4]
        }
    }

    # Wypisywanie informacji o każdym studencie
    for index, details in students.items():
        imie = details["imię"]
        nazwisko = details["nazwisko"]
        oceny = details["oceny"]
        srednia_ocen = sum(oceny) / len(oceny)
        print(f"Numer indeksu: {index}, Imię: {imie}, Nazwisko: {nazwisko}, Średnia ocen: {srednia_ocen:.2f}")

def main(): 1 usage
    student_grades()

if __name__ == "__main__":
    main()
```

```
C:\Program Files\Python312\python.exe C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\zadanie11.py
Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Kacper, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40
```

Zadanie 12

```
1 import pytest
2 from zadanie11 import student_grades
3
4
5 # Testy dla funkcji student_grades
6 def test_student_grades_basic():  # Pawelgabrieljonca
7     results = student_grades()
8     assert results[0]["średnia_ocen"] == 4.2
9     assert results[1]["średnia_ocen"] == 3.8
10    assert results[2]["średnia_ocen"] == 4.4
11
12 def test_student_grades_correct_data():  # Pawelgabrieljonca *
13     results = student_grades()
14     assert results[0]["imie"] == "Paweł"
15     assert results[1]["nazwisko"] == "Nowak"
16     assert results[2]["numer_indeksu"] == 12345
17
18 def test_student_grades_data_structure():  # Pawelgabrieljonca
19     results = student_grades()
20     assert isinstance(results, list)
21     assert isinstance(results[0], dict)
22     assert "średnia_ocen" in results[0]
```

```

"C:\Program Files\Python312\python.exe" "C:/Users/pawel/AppData/Local/Programs/PyCharm Professional/plugins/python-ce/helpers/pycharm/_jb_pytest_runner.py" --path "C:\Users\pawel\
Testing started at 15:41 ...
Launching pytest with arguments C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\test2.py --no-header --no-summary -q in C:\Users\pawel\Dokumenty\03 - studia\Skryptowe Języki programowania\pythonProject1\test2.py
===== test session starts =====
collecting ... collected 3 items

test2.py::test_student_grades_basic FAILED [ 33%]Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Kacper, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40

test2.py:5 (test_student_grades_basic)
def test_student_grades_basic():
    results = student_grades()
    > assert results[0]["średnia_ocen"] == 4.2
E     TypeError: 'NoneType' object is not subscriptable
|
test2.py:8: TypeError

test2.py::test_student_grades_correct_data FAILED [ 66%]Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Kacper, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40

test2.py:11 (test_student_grades_correct_data)
def test_student_grades_correct_data():
    results = student_grades()
    > assert results[0]["imię"] == "Paweł"
E     TypeError: 'NoneType' object is not subscriptable

test2.py:14: TypeError

test2.py::test_student_grades_data_structure FAILED [100%]Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Kacper, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40

test2.py:17 (test_student_grades_data_structure)

```

```

1  import pytest
2  from zadanie6 import unique_list
3
4
5  # Testy dla funkcji unique_list
6  def test_unique_list_basic():  # Pawelgabrieljonca
7      assert unique_list([1, 2, 3, 4, 3]) == {1, 2, 3, 4}
8
9  def test_unique_list_no_duplicates():  # Pawelgabrieljonca
10     assert unique_list([1, 2, 3, 4]) == {1, 2, 3, 4}
11
12  def test_unique_list_empty():  # Pawelgabrieljonca
13     assert unique_list([]) == set()
14

```

```
Project - Python tests in test2.py - Run
Test Results
test2
test_student_grades_basic
test_student_grades_correct_data
test_student_grades_data_structure

test2.py:5 (test_student_grades_basic)
def test_student_grades_basic():
    results = student_grades()
    > assert results[0]["średnia_ocen"] == 4.2
E   TypeError: 'NoneType' object is not subscriptable
test2.py:8: TypeError

test2.py::test_student_grades_correct_data FAILED [ 60%]Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Maciej, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40

test2.py:11 (test_student_grades_correct_data)
def test_student_grades_correct_data():
    results = student_grades()
    > assert results[0]["imię"] == "Paweł"
E   TypeError: 'NoneType' object is not subscriptable
test2.py:16: TypeError

test2.py::test_student_grades_data_structure FAILED [100%]Numer indeksu: 12345, Imię: Paweł, Nazwisko: Kowalski, Średnia ocen: 4.20
Numer indeksu: 23456, Imię: Maciej, Nazwisko: Nowak, Średnia ocen: 3.80
Numer indeksu: 34567, Imię: Artur, Nazwisko: Wiśniewski, Średnia ocen: 4.40

test2.py:17 (test_student_grades_data_structure)
def test_student_grades_data_structure():
    results = student_grades()
    > assert isinstance(results, list)
E   assert False
E   + where False = isinstance(None, list)
test2.py:20: AssertionError

===== 3 failed in 0.22s =====
Process finished with exit code 1
```

Wnioski:

Testy chodzą co prawda nie przechodzą ale dają informacje o błędach. Zadania pokazują jak należy posługiwać się listami, krotkami, zbiorami i słownikami. Dodatkowo trzeba było wykonać testy, dzięki czemu miałem okazję poznać jak się robi testy w pythonie.