



Sprawozdanie z laboratorium

Języki skryptowe

Py3 103: Instrukcje warunkowe, pętle i typy złożone

26.03.2025

Prowadzący: dr Beata Zieba

Grupa: 24-INF-SP/A

Hubert Jarosz

Oliwer Pawelski

1 Wstęp

Celem laboratorium było zapoznanie się z podstawowymi strukturami kontrolnymi i mechanizmami manipulacji danymi w Pythonie, takimi jak instrukcje warunkowe, pętle, oraz obsługa wyjątków. W ramach ćwiczeń wykorzystywano listy, słowniki oraz macierze do wykonywania różnorodnych operacji matematycznych i logicznych. Dodatkowo, implementować gry logiczne, takie jak Mastermind, oraz pracować z bardziej zaawansowanymi strukturami danych, jak tensory. Zajęcia pozwoliły na pogłębienie wiedzy o funkcjach wbudowanych w Pythonie, a także na praktyczne zastosowanie tych narzędzi w różnych kontekstach programistycznych.

2 Zadania

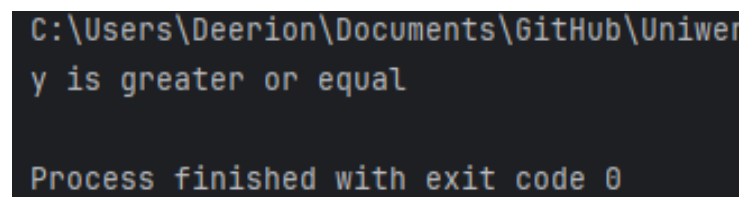
Zadanie 1 W jaki sposób zapisać instrukcję if/else w postaci wyrażenia?

```
1 # W jaki sposób zapisac instrukcje if/else w postaci wyrażenia?
2
3 # Przykład:
4 x = 10
5 y = 20
6
7 # Wyrażenie if/else
8 result = "x is greater" if x > y else "y is greater or equal"
9
10 print(result)
```

Listing 1: Kod do zadania 1

Składnia polecenia:

"wartość-jeśli-prawda" if "warunek" else "wartość-jeśli-fałsz"



```
C:\Users\Deerion\Documents\GitHub\Uniwer
y is greater or equal

Process finished with exit code 0
```

Zadanie 2 Jaka jest różnica pomiędzy break a continue?

```
1 # Jaka jest roznica pomiedzy break a continue?
2
3 # Przyklad uzycia break
4 print("Break:")
5 for i in range(10):
6     if i == 5:
7         break # Przerywa petle
8     print(i)
9 # Wynik: 0 1 2 3 4
10
11 print()
12
13 # Przyklad uzycia continue
14 print("Continue:")
15 for i in range(10):
16     if i == 5:
17         continue # Pomija reszte petli i przechodzi do nastepnej
18         iteracji
19     print(i)
20 # Wynik: 0 1 2 3 4 6 7 8 9
```

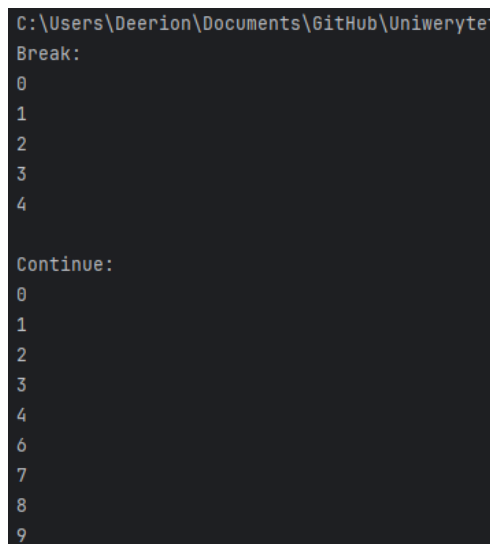
Listing 2: Kod do zadania 2

Break:

- Używane do zatrzymania wykonywania pętli.
- Po napotkaniu break, pętla jest natychmiast przerywana, a program przechodzi do pierwszej instrukcji po zakończeniu pętli.

Continue:

- Używane do pominięcia bieżącej iteracji pętli i przejścia do następnej.
- Gdy continue zostanie napotkane, wykonanie pętli przeskakuje do następnej iteracji, a reszta kodu w danej iteracji (po continue) jest ignorowana.



```
C:\Users\Deerion\Documents\GitHub\Uniwersyte
Break:
0
1
2
3
4

Continue:
0
1
2
3
4
6
7
8
9
```

Zadanie 3 Jaka jest różnica pomiędzy while a for?

```
1 # Jaka jest roznica pomiedzy while a for?
2
3 # Przykład uzycia while (Nieokreslona liczba iteracji)
4 print("While loop:")
5 i = 0
6 while i < 5: # Petla bedzie wykonywana dopoki i < 5
7     print(i)
8     i += 1
9 # Wynik: 0 1 2 3 4
10
11 print()
12
13 # Przykład uzycia for (Okreslona liczba iteracji)
14 print("For loop:")
15 for i in range(5): # Petla bedzie wykonywana dla i = 0, 1, 2, 3, 4
16     print(i)
17 # Wynik: 0 1 2 3 4
```

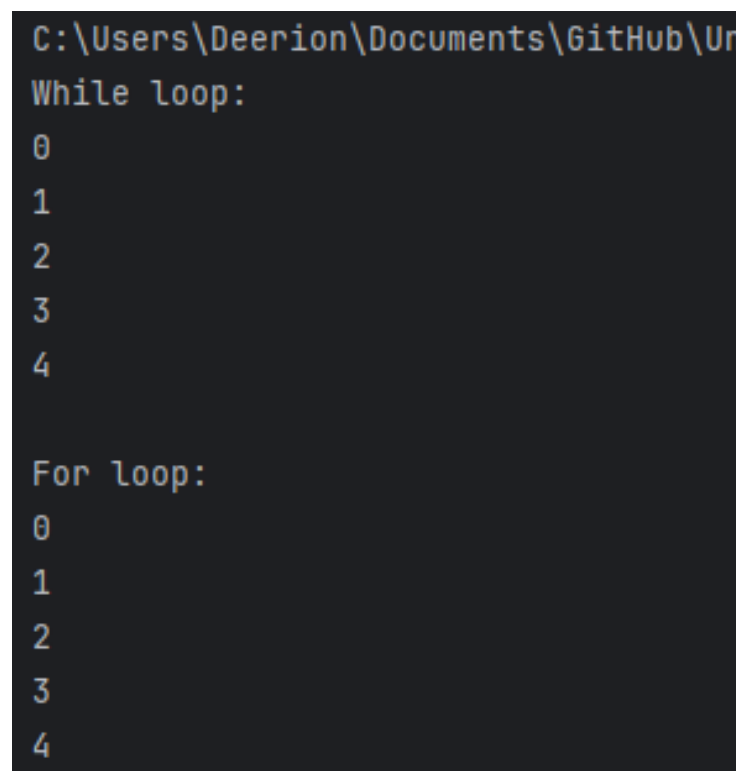
Listing 3: Kod do zadania 3

While:

Pętla while wykonuje blok kodu, dopóki warunek jest prawdziwy. Działa na zasadzie sprawdzania warunku przed każdą iteracją.

For:

Pętla for jest używana do iteracji po sekwencji (np. listach, krotkach, ciągach znaków, słownikach, zakresach itp.). Wykonuje kod dla każdego elementu w tej sekwencji.



```
C:\Users\Deerion\Documents\GitHub\Un
While loop:
0
1
2
3
4

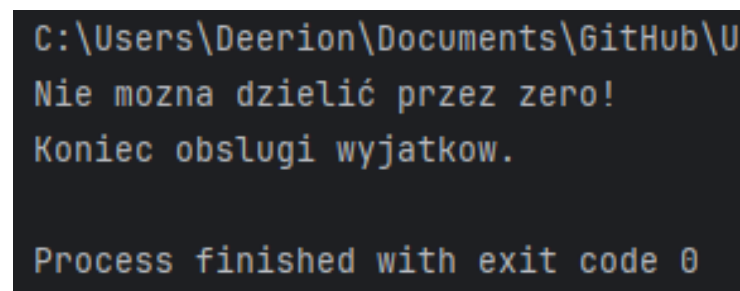
For loop:
0
1
2
3
4
```

Zadanie 4 Jak obsługuje się wyjątki w Pythonie?

```
1 # Jak obsługuje się wyjątki w Pythonie?
2
3 try:
4     # Kod, który może wywołać wyjątek
5     result = 10 / 0
6 except ZeroDivisionError:
7     # Kod obsługujący wyjątek
8     print("Nie można dzielić przez zero!")
9 else:
10    # Kod, który wykona się, jeśli nie wystąpi wyjątek
11    print("Wynik:", result)
12 finally:
13    # Kod, który wykona się niezależnie od tego, czy wystąpił wyjątek
14    # , czy nie
15    print("Koniec obsługi wyjątków.")
```

Listing 4: Kod do zadania 4

W Pythonie obsługa wyjątków odbywa się za pomocą bloków try, except, oraz (opcjonalnie) else i finally. Dzięki temu możemy uchwycić błędy w kodzie i odpowiednio zareagować, nie przerywając działania programu



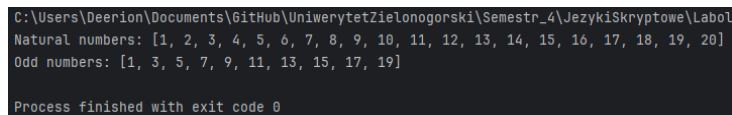
```
C:\Users\Deerion\Documents\GitHub\U
Nie można dzielić przez zero!
Koniec obsługi wyjątków.

Process finished with exit code 0
```

Zadanie 5 Utworzyć listę liczb naturalnych od 1 do 20. Następnie za pomocą mechanizmu list składanych utworzyć listę liczb nieparzystych.

```
1 # 1) Utworzyć listę liczb naturalnych od 1 do 20.
2 # 2) Następnie za pomocą mechanizmu list składanych utworzyć listę
   liczb nieparzystych.
3
4 # 1)
5 natural_numbers = list(range(1, 21))
6
7 # 2)
8 odd_numbers = [num for num in natural_numbers if num % 2 != 0]
9
10 print("Natural numbers:", natural_numbers)
11 print("Odd numbers:", odd_numbers)
```

Listing 5: Kod do zadania 5

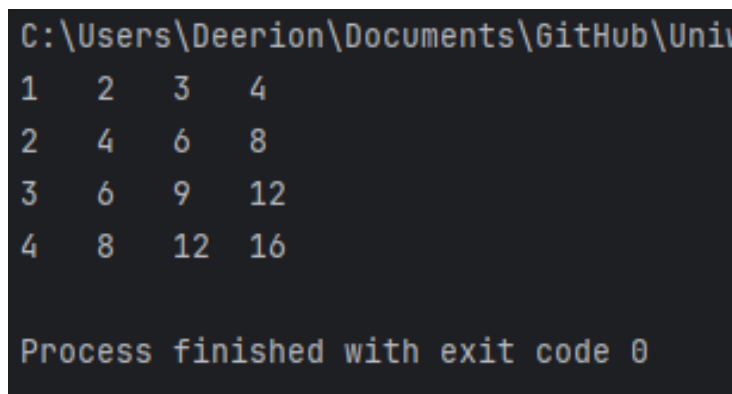


```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielonogorski\Semestr_4\JęzykiSkryptowe\LaboL
Natural numbers: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
Odd numbers: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
Process finished with exit code 0
```

Zadanie 6 Utworzyć macierz o wymiarach 4×4 iloczynów indeksów macierzy.

```
1 # Utworzyć macierz o wymiarach 4 4 iloczynów indeksów macierzy.
2 #
3 # a1,1 a1,2 a1,3 a1,4
4 # | a2,1 a2,2 a2,3 a2,4 |
5 # | a3,1 a3,2 a3,3 a3,4 |, ai ,j = ij
6 # a4,1 a4,2 a4,3 a4,4
7
8 matrix = [(i + 1) * (j + 1) for j in range(4)] for i in range(4)]
9
10 for row in matrix:
11     print('\t'.join(map(str, row))) # Wynik z tabami
12     # print(row) # Surowy wynik (bez tab w)
```

Listing 6: Kod do zadania 6



```
C:\Users\Deerion\Documents\GitHub\Univ
1 2 3 4
2 4 6 8
3 6 9 12
4 8 12 16
Process finished with exit code 0
```

Zadanie 7 Utworzyć tensor o wymiarach $3 \times 3 \times 3$ ilorazów indeksów tensora, analogicznie jak w Zadaniu 10.6.

```
1 # Utworzyc tensor o wymiarach 3 3 3 ilorazow indeksow tensora ,  
2   analogicznie jak w Zadaniu 6.  
3 tensor = [[[(i + 1) / (j + 1) / (k + 1) for k in range(3)] for j in  
4           range(3)] for i in range(3)]  
5 for layer in tensor:  
6     for row in layer:  
7       print('\t'.join(map(str, row)))  
8     print()
```

Listing 7: Kod do zadania 7

```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielonogorski\S  
1.0 0.5 0.3333333333333333  
0.5 0.25 0.16666666666666666  
0.3333333333333333 0.16666666666666666 0.11111111111111111  
  
2.0 1.0 0.6666666666666666  
1.0 0.5 0.3333333333333333  
0.6666666666666666 0.3333333333333333 0.2222222222222222  
  
3.0 1.5 1.0  
1.5 0.75 0.5  
1.0 0.5 0.3333333333333333  
  
Process finished with exit code 0
```

Zadanie 8 **Utworzyć słownik w którym kluczami są adresy email, a wartościami imiona i nazwiska a następnie zwrócić posortowany słownik.**

```
1 # Utworzyc slownik w ktorym kluczami sa adresy email, a wartosciami
2 # imiona i nazwiska, a nast pnie zwrocic posortowany slownik.
3
4 data = {
5     "piotr.kowalski@example.com": "Piotr Kowalski",
6     "barbara.kowalska@example.com": "Barbara Kowalska",
7     "bartosz.zielony@example.com": "Bartosz Zielony",
8     "ice.tink@example.com": "Ice Tink",
9 }
10
11 # Sort the dictionary by keys (email addresses)
12 sorted_data = dict(sorted(data.items()))
13
14 print("Posortowany slownik:\n" + "\n".join(f"\t{key}: {value}" for
      key, value in sorted_data.items()))
```

Listing 8: Kod do zadania 8

```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielono
Posortowany słownik:
    barbara.kowalska@example.com: Barbara Kowalska
    bartosz.zielony@example.com: Bartosz Zielony
    ice.tink@example.com: Ice Tink
    piotr.kowalski@example.com: Piotr Kowalski

Process finished with exit code 0
```


Zadanie 9 Utworzyć listę słowników zawierającą: imiona nazwiska, wiek, oraz 10 ocen studentów. Wykorzystując mechanizm list składanych obliczyć średnią ocen każdego studenta, średnią ocen wszystkich studentów oraz medianę wieku oraz medianę długości nazwiska.

```
1 import statistics
2
3 # Utworz liste slownikow zawierajaca dane studentow
4 students = [
5     {"name": "John", "surname": "Doe", "age": 20, "grades": [2, 3, 4,
6     5, 3, 4, 5, 2, 3, 4]},
7     {"name": "Jane", "surname": "Smith", "age": 22, "grades": [5, 3,
8     2, 4, 5, 3, 4, 5, 2, 3]},
9     {"name": "Alice", "surname": "Johnson", "age": 21, "grades": [4,
10    2, 5, 5, 3, 4, 5, 2, 3, 4]},
11    {"name": "Bob", "surname": "Brown", "age": 23, "grades": [3, 3,
12    2, 4, 5, 3, 4, 5, 2, 3]},
13    {"name": "Eve", "surname": "Williams", "age": 24, "grades": [5,
14    3, 4, 5, 3, 4, 5, 2, 3, 4]}
15 ]
16
17 # Oblicz srednia ocen kazdego studenta
18 avg_grades = [sum(student["grades"]) / len(student["grades"]) for
19 student in students]
20
21 # Oblicz srednia ocen wszystkich studentow
22 total_grades = sum(sum(student["grades"]) for student in students)
23 total_count = sum(len(student["grades"]) for student in students)
24 avg_all_students = total_grades / total_count
25
26 # Oblicz mediane wieku studentow
27 ages = [student["age"] for student in students]
28 median_age = statistics.median(ages)
29
30 # Oblicz mediane dlugosci nazwisk studentow
31 surname_lengths = [len(student["surname"]) for student in students]
32 median_surname_length = statistics.median(surname_lengths)
33
34 print("Srednia ocen kazdego studenta:", avg_grades)
35 print("Srednia ocen wszystkich studentow:", avg_all_students)
36 print("Mediana wieku studentow:", median_age)
37 print("Mediana dlugosci nazwisk:", median_surname_length)
```

Listing 9: Kod do zadania 9

```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielonogorski>
Średnia ocen każdego studenta: [3.5, 3.6, 3.7, 3.4, 3.8]
Średnia ocen wszystkich studentów: 3.6
Mediana wieku studentów: 22
Mediana długości nazwisk: 5
```

Zadanie 10 Napisać program w którym użytkownik musi odgadnąć losową sekwencję 4 z 5 cyfr. W każdym kroku otrzymuje on informację o tym ile cyfr jest na swoim miejscu oraz ile występuje w sekwencji jednak na innej pozycji (gra Mastermind). Wykorzystać fragment kodu z Listingu 15.

```
1 import random
2 random .choices([x for x in range(1,4)], k=4)
3 #ponizej własny kod

1 import random
2
3 # Generowanie losowej sekwencji
4 secret = random.choices([x for x in range(1, 4)], k=4)
5 print(secret) # Sekwencja do odgadnięcia
6 # Gra
7 while True:
8     guess = list(map(int, input("Podaj swoją propozycja (4 cyfry od
9     1-3): ").split()))
10    correct_pos = sum([1 for i in range(4) if guess[i] == secret[i]])
11    correct_digits = sum([min(guess.count(x), secret.count(x)) for x
12    in set(guess)]) - correct_pos
13    print(f"Cyfry na właściwych miejscach: {correct_pos}")
14    print(f"Cyfry w sekwencji, ale na innych miejscach: {
15    correct_digits}")
16    if correct_pos == 4:
17        print("Gratulacje! Zgadłeś sekwencję!")
18        break
```

Listing 10: Kod do zadania 10

```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielonogorski\Seme
[3, 2, 1, 3]
Podaj swoją propozycja (4 cyfry od jeden do pięciu): 2 3 1 2
Cyfry na właściwych miejscach: 1
Cyfry w sekwencji, ale na innych miejscach: 2
Podaj swoją propozycja (4 cyfry od jeden do pięciu): 2 2 4 2
Cyfry na właściwych miejscach: 1
Cyfry w sekwencji, ale na innych miejscach: 0
Podaj swoją propozycja (4 cyfry od jeden do pięciu): 3 2 1 3
Cyfry na właściwych miejscach: 4
Cyfry w sekwencji, ale na innych miejscach: 0
Gratulacje! Zgadłeś sekwencję!

Process finished with exit code 0
```

Zadanie 11 Podać cztery operacje modyfikujące obiekt słownika.

```
1 import random
2
3 # Pobranie dlugosci sekwencji oraz zakresu z ktorego losowane sa
  liczby
4 sequence_length = int(input("Podaj dlugosc sekwencji: "))
5 range_start = int(input("Podaj poczatkowa liczbe zakresu: "))
6 range_end = int(input("Podaj koncowa liczbe zakresu: "))
7
8 # Generowanie losowej sekwencji
9 secret = random.choices([x for x in range(range_start, range_end + 1)
10 ], k=sequence_length)
11 print("Sekwencja do odgadnienia zostala wylosowana!")
12 print(secret) # Sekwencja do odgadnienia
13 # Gra
14 while True:
15     guess = list(
16         map(int, input(f"Podaj swoja propozycje ({sequence_length}
17         cyfr od {range_start} do {range_end}): ").split()))
18
19     if len(guess) != sequence_length:
20         print(f"Prosze podac dokladnie {sequence_length} cyfr.")
21         continue
22
23     correct_pos = sum([1 for i in range(sequence_length) if guess[i]
24     == secret[i]])
25     correct_digits = sum([min(guess.count(x), secret.count(x)) for x
26     in set(guess)]) - correct_pos
27
28     print(f"Cyfry na wlasciwych miejscach: {correct_pos}")
29     print(f"Cyfry w sekwencji, ale na innych miejscach: {
30     correct_digits}")
31
32     if correct_pos == sequence_length:
33         print("Gratulacje! Zgadles sekwencje!")
34         break
```

Listing 11: Kod do zadania 11

```
C:\Users\Deerion\Documents\GitHub\UniwersytetZielonogorski\Semes
Podaj dlugosc sekwencji: 8
Podaj poczatkowa liczbe zakresu: 1
Podaj koncowa liczbe zakresu: 10
Sekwencja do odgadnienia zostala wylosowana!
[5, 6, 10, 6, 3, 5, 7, 8]
Podaj swoja propozycje (8 cyfr od 1 do 10): 3 2
Proszę podać dokładnie 8 cyfr.
Podaj swoja propozycje (8 cyfr od 1 do 10): 2 2 3 4 6 7 8 10
Cyfry na wlasciwych miejscach: 0
Cyfry w sekwencji, ale na innych miejscach: 5
Podaj swoja propozycje (8 cyfr od 1 do 10): 5 6 10 6 3 5 7 8
Cyfry na wlasciwych miejscach: 8
Cyfry w sekwencji, ale na innych miejscach: 0
Gratulacje! Zgadles sekwencje!
```