

Wstęp do programowania

INP003203L

rok akademicki 2020/21

semestr zimowy

Laboratorium 11

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

L-1 p. 220



Plan prezentacji

- Przykłady wykorzystania pętli:
 - Wczytywanie danych
 - Weryfikacja danych wejściowych
- Pętle zagnieżdżone

Wczytywanie danych z wartownikiem

- Do wczytywania danych o różnych długościach wykorzystywaliśmy dwie techniki:
 - na koniec każdej iteracji pytamy użytkownika, czy chce kontynuować,
 - na początku programu pytamy użytkownika, o długość danych.
- Dla długich list obie techniki są niewygodne:
 - w pierwszej trzeba wielokrotnie potwierdzać kontynuację,
 - w drugiej użytkownik musi sam przeliczyć elementy.

Wczytywanie danych z wartownikiem

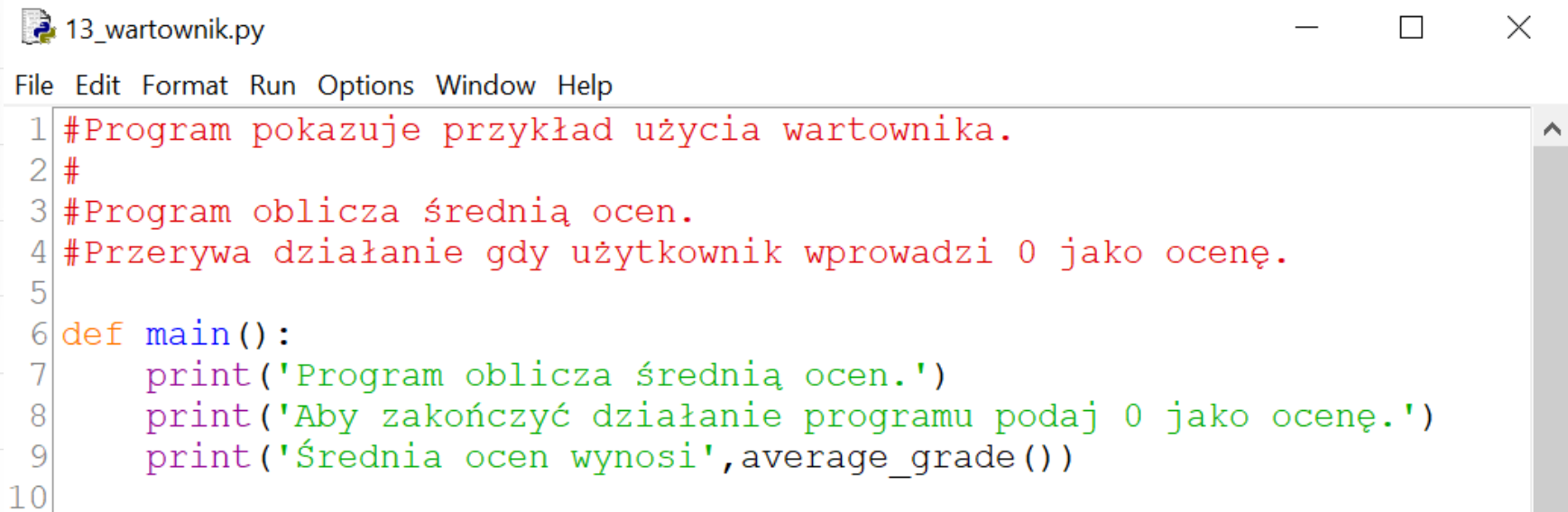
- Rozwiązaniem może być wykorzystanie wartownika.
- Wartownik to specjalna wartość, która wskazuje koniec listy elementów.
- Przykładowo: w programie obliczającym średnią ocen - wprowadzenie wartości 0 (która nie jest poprawną oceną) może oznaczać koniec danych.



Wczytywanie danych z wartownikiem

- Wybór rozwiązania zależy od specyfiki problemu (znana długość danych? interakcja z użytkownikiem? możliwość wskazania wartości wartownika?)
- Wybór rozwiązania może także zależeć od preferencji programisty

Wczytywanie danych z wartownikiem



```
13_wartownik.py
File Edit Format Run Options Window Help
1 #Program pokazuje przykład użycia wartownika.
2 #
3 #Program oblicza średnią ocen.
4 #Przerywa działanie gdy użytkownik wprowadzi 0 jako ocenę.
5
6 def main():
7     print('Program oblicza średnią ocen.')
8     print('Aby zakończyć działanie programu podaj 0 jako ocenę.')
9     print('Średnia ocen wynosi', average_grade())
10
```

Wczytywanie danych z wartownikiem

13_wartownik.py

File Edit Format Run Options Window Help

```
11 # I: brak
12 # P: funkcja wczytuje dane od użytkownika obliczając jednocześnie
13 #    sumę wszystkich ocen oraz licząc ich liczbę;
14 #    pętla while jest kontynuowana dopóki oceną nie będzie 0;
15 #    zwracana wartość to iloraz sumy ocen i ich liczby
16 #    instrukcja if pozwala uniknąć dzielenia przez 0 jeśli
17 #    nie wczytano żadnych danych
18 # O: średnia ocen
19 def average_grade():
```

Wczytywanie danych z wartownikiem

13_wartownik.py

File Edit Format Run Options Window Help

```
19 def average_grade():
20     #inicjalizacja zmiennych
21     total = 0.0 #suma wczytanych ocen
22     counter = 0 #liczba wczytanych ocen
23
24     #odczytanie pierwszej oceny
25     grade = get_grade()
26
27     #pętla jest kontynuowana, jeśli ocena jest różna od zera
28     while grade != 0:
29         total += grade #wczytana ocena jest dodawana do sumy
30         counter += 1 #oraz zwiększany jest licznik ocen
31         print(counter, grade, total)
32         grade = get_grade() #wczytywana jest kolejna ocena
33
34     #zakończenie obliczeń
35     #średnia obliczana jako suma ocen dzielona przez ich liczbę
36     #if zapobiega dzieleniu przez 0
37     if counter != 0:
38         # w tym przypadku obliczamy średnią i ją zwracamy
39         return total/counter
40     else:
41         # w tym przypadku zwracamy wartość None
42         return
```


Wczytywanie danych z wartownikiem

13_wartownik.py - D:\repozytoria\python-examples\pętle\13_wartownik.py (3.9.7)

File Edit Format Run Options Window Help

```
44 # I: brak
45 # P: wczytanie oceny (jako float)
46 # O: wczytana ocena
47 def get_grade():
48     return float(input('Podaj ocenę: '))
49
50 main()
51
52
```

Wczytywanie danych z wartownikiem - porównanie wersji

13_wartownik.py

File Edit Format Run Options Window Help

```
19 def average_grade():
20     #inicjalizacja zmiennych
21     total = 0.0 #suma wczytanych ocen
22     counter = 0 #liczba wczytanych ocen
23
24     #odczytanie pierwszej oceny
25     grade = get_grade()
26
27     #pętla jest kontynuowana, jeśli ocena jest różna od zera
28     while grade != 0:
29         total += grade #wczytana ocena jest dodawana do sumy
30         counter += 1 #oraz zwiększany jest licznik ocen
31         print(counter, grade, total)
32         grade = get_grade() #wczytywana jest kolejna ocena
33
34     #zakończenie obliczeń
35     #średnia obliczana jako suma ocen dzielona przez ich liczbę
36     #if zapobiega dzieleniu przez 0
37     if counter != 0:
38         # w tym przypadku obliczamy średnią i ją zwracamy
39         return total/counter
40     else:
41         # w tym przypadku zwracamy wartość None
42         return
```

Wczytywanie danych z wartownikiem - porównanie wersji

13_wartownik_B.py

File Edit Format Run Options Window Help

```
19 def average_grade():
20     #inicjalizacja zmiennych
21     total = 0.0 #suma wczytanych ocen
22     counter = 0 #liczba wczytanych ocen
23
24     grade = True #zmienna grad steruje przebiegiem pętli
25
26     #pętla jest kontynuowana, jeśli ocena jest różna od zera
27     while grade:
28         grade = float(input('Podaj ocenę: '))
29         if grade:
30             total += grade #wczytana ocena jest dodawana do sumy
31             counter += 1 #oraz zwiększany jest licznik ocen
32         else:
33             print("OK. Kończę liczenie średniej")
34             print(counter, grade, total)
35
36     #zakończenie obliczeń
37     #średnia obliczana jako suma ocen dzielona przez ich liczbę
38     #if zapobiega dzieleniu przez 0
39     if counter:
40         # w tym przypadku obliczamy średnią
41         total /= counter
42     else:
43         # w tym przypadku wynikiem jest NaN
44         total = float('NaN')
45
46     #zwracamy obliczoną średnią
47     return total
```

Wczytywanie danych z wartownikiem

Najważniejsze różnice:

1. Dwukrotne vs. jednokrotne wywołanie wczytywania danych
2. Konieczność dwukrotnego sprawdzania warunku w drugiej wersji

Pozostałe różnice:

1. Różny zapis warunków w pętli `while` oraz instrukcjach `if`
2. Wykorzystanie wartości `None/NaN` do sygnalizacji wyjątkowej sytuacji
3. Różne umiejscowienie polecenia `return`
4. Wykorzystanie funkcji do obsługi wczytywania danych

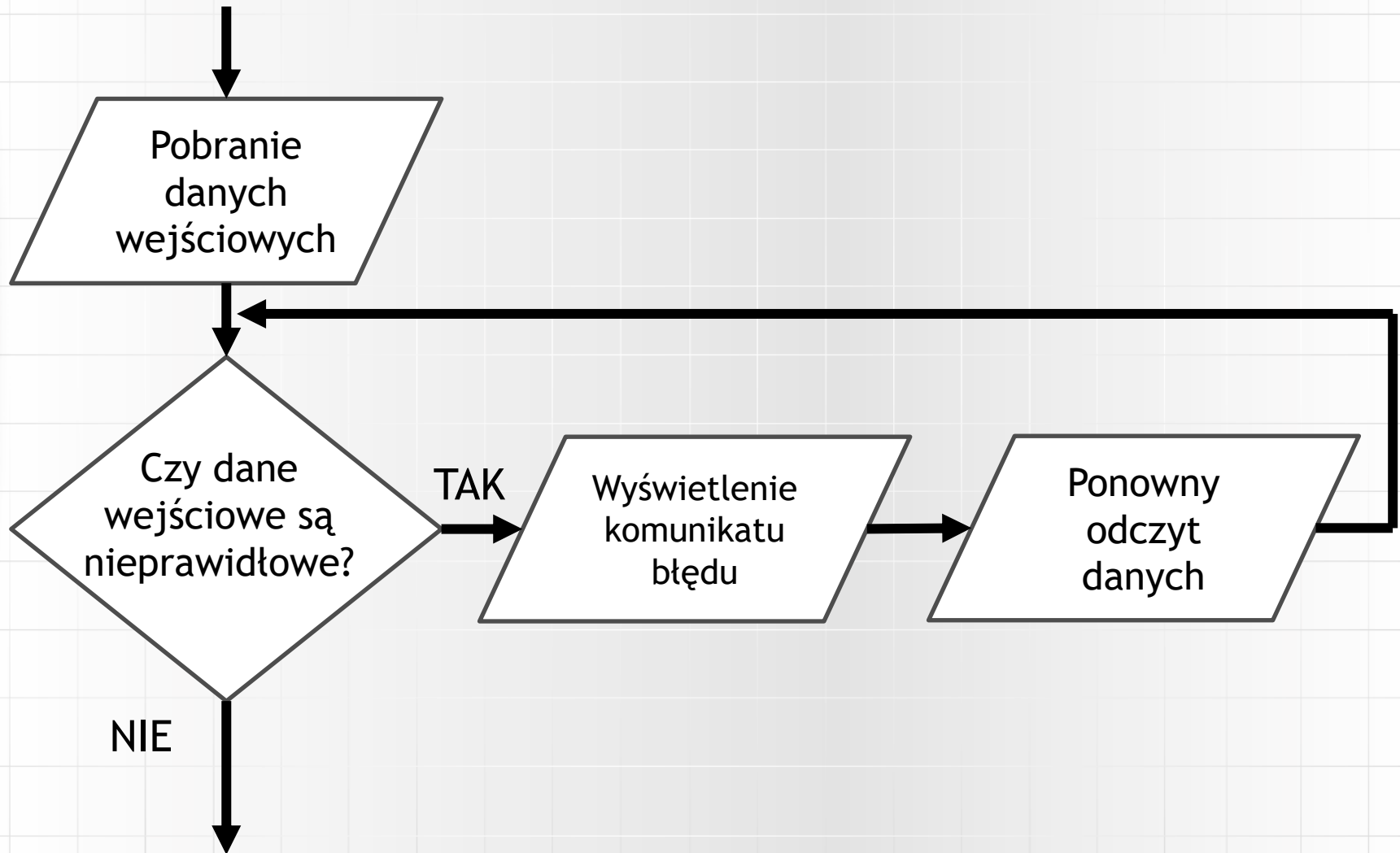
Wczytywanie danych z wartownikiem

- Ten sam problem można rozwiązać na wiele różnych sposobów
- Wybór konkretnego rozwiązania zależy od wielu czynników

Weryfikacja danych wejściowych

- Proces sprawdzania poprawności danych, to weryfikacja danych wejściowych
- Dane najczęściej odczytuje się w pętli, wykonywanej dopóki dane są nieprawidłowe

Weryfikacja danych wejściowych



Weryfikacja danych wejściowych

14_weryfikacja_danych.py

File Edit Format Run Options Window Help

```
44 # I: brak
45 # P: wczytanie oceny (jako float)
46 #   jako prawidłowe dane przyjmowane są poprawne oceny lub wartość 0
47 # O: wczytana ocena
48 def get_grade():
49     grade = float(input('Podaj ocenę: '))
50     # sprawdzaj, czy wczytana liczba jest poprawną oceną lub zerem
51     # jeśli nie poproś o liczbę jeszcze raz
52     while not (is_grade(grade) or grade == 0.0):
53         print('Podana liczba nie jest oceną!')
54         grade = float(input('Podaj ocenę: '))
55     return grade
56
57 # I: liczba
58 # P: prawidłowa ocena jest jedną z wartości: 2, 3, 3.5, 4, 4.5, 5, 5.5;
59 # O: wartość logiczna - czy liczba jest oceną
60 def is_grade(number):
61     return number == 2.0 or \
62            number == 3.0 or number == 3.5 or \
63            number == 4.0 or number == 4.5 or \
64            number == 5.0 or number == 5.5
```




Pętle zagnieżdżone

- Wewnątrz pętli można także umieszczać pętle
- Taka pętla nazywana jest pętlą zagnieżdżoną

- Przykład pętli zagnieżdżonej - zegar
Zegar odlicza w pętli godziny

Pętle zagnieżdżone

- Wewnątrz pętli można także umieszczać pętle
- Taka pętla nazywana jest pętlą zagnieżdżoną
- Przykład pętli zagnieżdżonej - zegar
Zegar odlicza w pętli godziny, ale na każdą godzinę przypada pętla odliczająca minuty



Pętle zagnieżdżone

- Wewnątrz pętli można także umieszczać pętle
- Taka pętla nazywana jest pętlą zagnieżdżoną
- Przykład pętli zagnieżdżonej - zegar
Zegar odlicza w pętli godziny, ale na każdą godzinę przypada pętla odliczająca minuty, a w każdej minucie jest pętla odliczająca sekundy.

Pętle zagnieżdżone



15_petle_zagniezdzone.py



File Edit Format Run Options Window Help

```
1 #Program ilustrujący działanie pętli for.
2 #Program wypisuje liczby od 0 do 59.
3 #Imituje odliczanie stopera.
4
5 def main():
6     for seconds in range(60):
7         print(seconds)
8
9 main()
10
```

Pętle zagnieżdżone

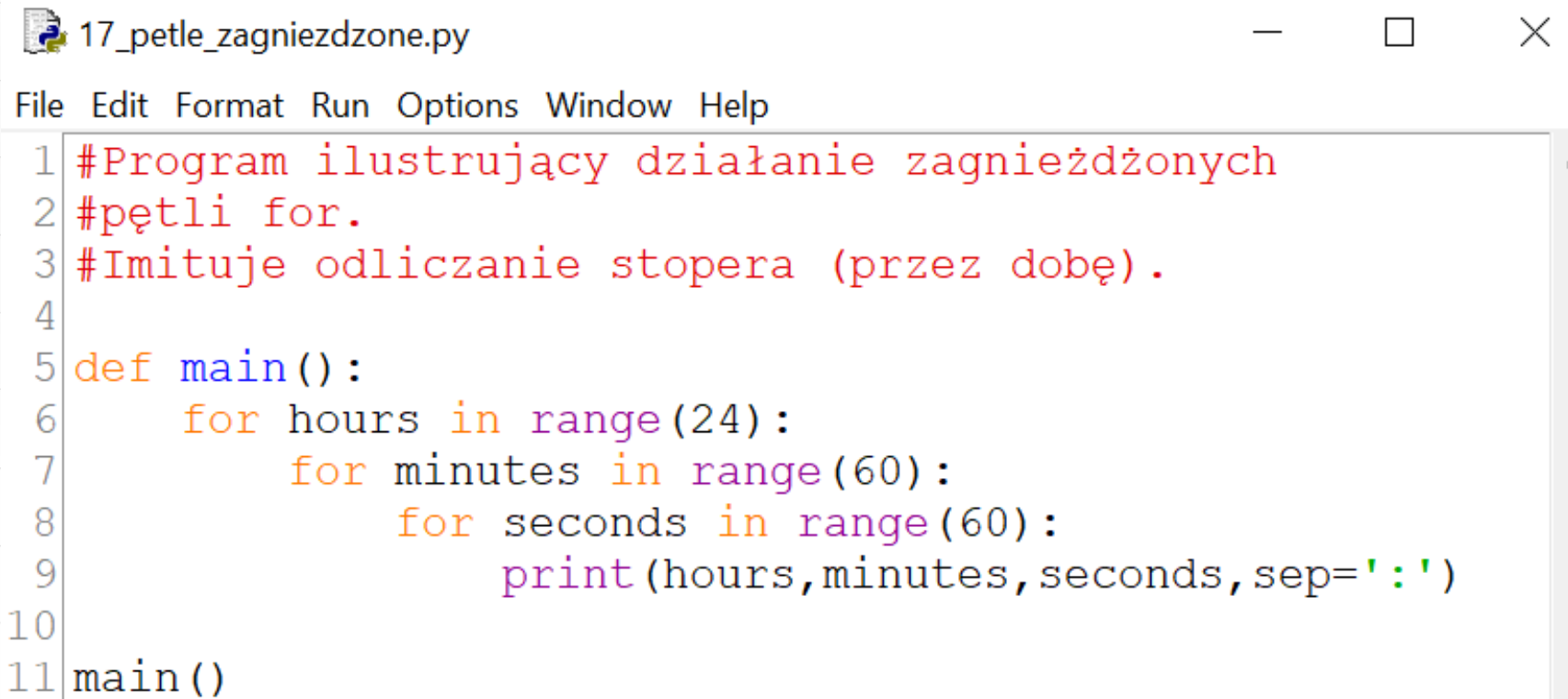
16_petle_zagniezdzone.py



File Edit Format Run Options Window Help

```
1 #Program ilustrujący działanie zagnieżdżonych
2 #pętli for.
3 #Imituje odliczanie stopera (przez godzinę).
4
5 def main():
6     for minutes in range(60):
7         for seconds in range(60):
8             print(minutes, seconds, sep=':')
9
10 main()
```

Pętle zagnieżdżone



```
17_petle_zagniezdzone.py
File Edit Format Run Options Window Help
1 #Program ilustrujący działanie zagnieżdżonych
2 #pętli for.
3 #Imituje odliczanie stopera (przez dobę).
4
5 def main():
6     for hours in range(24):
7         for minutes in range(60):
8             for seconds in range(60):
9                 print(hours, minutes, seconds, sep=':')
10
11 main()
```

Pętle zagnieżdżone

- Zmienna sterująca pętlą wewnętrzną może zależeć od wartości zmiennej kontrolującej pętlę zewnętrzną



Pętle zagnieżdżone

```
18_petle_zagniezdzone.py
File Edit Format Run Options Window Help
1 #Program "rysuje" na ekranie schodki o n stopniach.
2
3 def main():
4     print('Program wyświetla schodki o n stopniach')
5     n = int(input('Podaj n (dodatnią liczbę całkowitą): '))
6     stairs(n)
7
8 # I: funkcja pobiera liczbę całkowitą
9 # P: funkcja wypisuje n wierszy
10 #     w zerowym wierszu wypisuje zero spacji i znak '#'
11 #     w pierwszym wierszu jedną spację i znak '#'
12 #     w kolejnych wierszach zawsze o jedną spację więcej
13 #     w ten sposób powstają schodki
14 # O: funkcja nie zwraca wartości
15 def stairs(n):
16     #pętla po wierszach
17     for row in range(n):
18         #wypisanie spacji
19         #liczba spacji odpowiada - numerowi wiersza
20         for col in range(row):
21             print(' ', end='')
22         #wypisanie końcowego znaku '#'
23         print('#')
24
25 main()
```




Pętle zagnieżdżone

```
19_petle_zagniezdzone.py
File Edit Format Run Options Window Help
1 #Program "rysuje" na ekranie schodki o n stopniach.
2
3 def main():
4     print('Program wyświetla schodki o n stopniach')
5     n = int(input('Podaj n (dodatnią liczbę całkowitą): '))
6     stairs(n)
7
8 # I: funkcja pobiera liczbę całkowitą
9 # P: funkcja wypisuje n wierszy (schodków)
10 #   schodki wypisywane są w funkcji single_stair()
11 # O: funkcja nie zwraca wartości
12 def stairs(n):
13     for row in range(n):
14         single_stair(row)
15
16 # I: funkcja pobiera liczbę całkowitą (numer schodka)
17 # P: funkcja wypisuje n spacji oraz znak '#'
18 # O: funkcja nie zwraca wartości
19 def single_stair(n):
20     for x in range(n):
21         print(' ', end='')
22         print('#')
23
24
25 main()
```



Absolutne minimum

- Trzy sposoby wczytywania danych
- Weryfikacja danych wejściowych w pętli
- Pętle zagnieżdżone