Wrocław University of Science and Technology

Wstęp do programowania

INP003203L

Semestr zimowy 2020/2021

Poniedziałek, 7:30 - 9:00

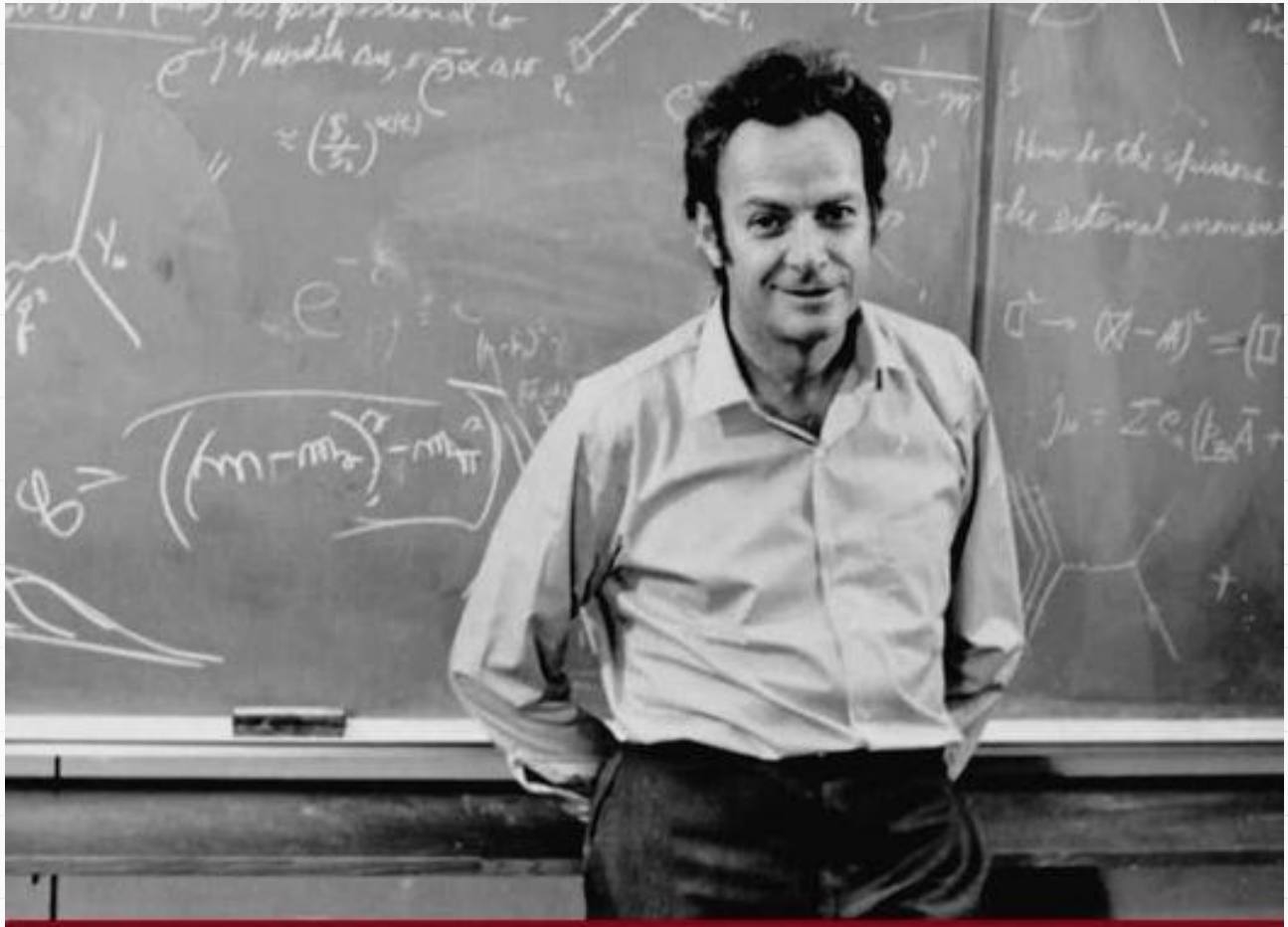
sala wirtualna
– zajęcia online

Sylwia Majchrowska

sylwia.majchrowska@pwr.edu.pl

https://majsylw.netlify.app/teaching/

pokój 213, budynek L-1

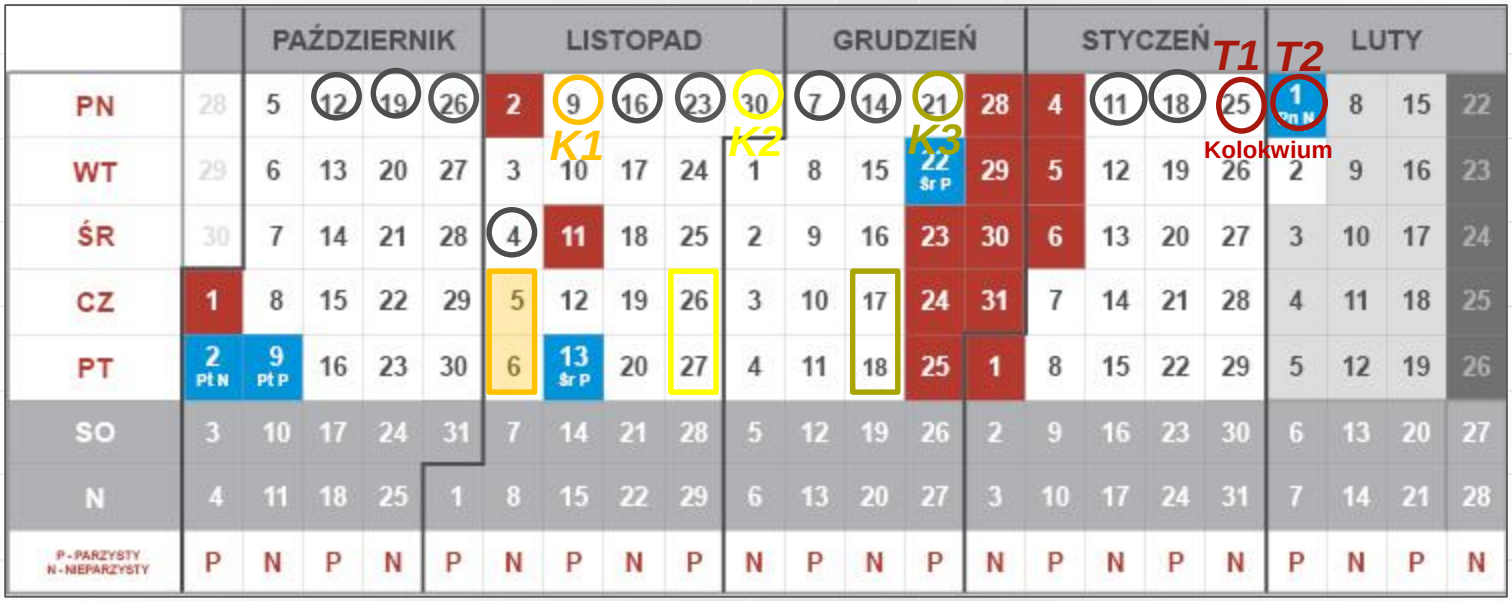


SCIENCE:

- If you don't make mistakes, you're doing it wrong.
- If you don't correct those mistakes, you're doing it really wrong.
- If you can't accept that you're mistaken, you're not doing it at all.

Wrocław University of Science and Technology

Spojrzenie na kalendarz



Poniedziałek, 7:30 - 9:00
sala wirtualna

Konsultacje:
Poniedziałki, 19:30 - 20:30
Środy, 19:30 – 20:30

Wrocław University of Science and Technology

Zagadnienia – kartkówka nr 1

1. Zmienne i typy danych:
   - Łańcuchy znaków: str
   - Liczby: całkowite (int), zmiennoprzecinkowe (float)
2. Operacje wyjścia – funkcja **print**():
   - wypisywanie komunikatów oraz ich formatowanie,
   - znaki sterujące,
   - argumenty pozycyjne i nazwane.
3. Operacje wejścia – funkcja **input()**.
4. Komentarze:
   - Jednolinijkowe, wielolinijkowe,
   - Kiedy je stosujemy.

Wrocław University of Science and Technology

Zagadnienia – kartkówka nr 1 cd.

5. Operacje matematyczne:
   - operatory arytmetyczne,
   - operatory binarne i unarne,
   - operowanie na napisach,
   - kolejność wykonywania działań.
6. Funkcje:
   - definiowanie funkcji (*i procedur*),
   - przekazywanie argumentów,
   - zwracanie wyniku.
7. Projektowanie programu:
   - IPO – input, processing, output,
   - sposoby prezentacji programu – funkcja main().

Wrocław University of Science and Technology

Zmienne i typy danych

Zmienna to miejsce w pamięci komputera reprezentowane przez określoną nazwę. Zmienna posiada **nazwę** i **wartość**.

wiek = 19

wiek ⟶ 19

przypisanie

= ⟶ operator przypisania

Jeśli chcesz nadać nazwę zmiennej, musisz przestrzegać kilku ścisłych zasad:

- nazwa zmiennej może składać się z wielkich lub małych liter, cyfr i znaku _ (podkreślenia)
- nazwa zmiennej musi zaczynać się od litery;
- znak podkreślenia jest uznawany za literę;
- duże i małe litery są traktowane jako różne znaki (Alicja i ALICE to te różne napisy);
- nazwa zmiennej nie może być żadnym ze słów zastrzeżonych w Pythonie (to tak zwane słowa kluczowe).

Typy danych:

- str: łańcuch znaków (ang. string) *'ala'* oraz *"ala„*
- int: liczba całkowita (ang. integer) *1* oraz *-1*
- float: liczba zmiennoprzecinkowa (ang. float) *.9* oraz *-1.0*
- bool: zmienna logiczna (ang. boolean) *True* oraz *False*

Słowa kluczowe

```
['False',      'None',
'True',  'and',  'as',
'assert',      'break',
'class',    'continue',
'def',  'del',  'elif',
'else',       'except',
'finally',       'for',
'from',       'global',
'if',  'import',  'in',
'is',         'lambda',
'nonlocal',      'not',
'or',  'pass',  'raise',
'return',        'try',
'while',         'with',
'yield']
```

Wrocław University of Science and Technology

Operacje wyjścia

- funkcja print(): wypisywanie komunikatów na ekran

```
print("Witaj świecie!")
```

→ wywołanie funkcji print()

Funkcja print() - argumenty nazwane:
sep, end, file, flush

- Do funkcji print() istnieje możliwość przekazania kilku napisów rozdzielając je przecinkiem.

```
1  print("The itsy bitsy spider" , "climbed up" , "the waterspout.")
2
3
```

- W ten sposób można też przekazywać literały różnego typu.

```
>>> print("Ala ma",2,"koty.")
Ala ma 2 koty.
```

- Wprowadzone po przecinku dane w wypisywanym komunikacie oddzielone są za pomocą spacji.

https://docs.python.org/3.8/library/functions.html#print

Wrocław University of Science and Technology

Funkcja print()

- formatowanie napisów

Znaki sterujące:

- Znak specjalny końca linii *'\n'*,
- Znak specjalny tabulacji *'\t'*,
- Znak specjalny backspace *'\b'*,
- Znak apostrofu \',
- Znak cudzysłowu \",
- Ukośnik \\.

```
format(value[, format_spec])
```

Funkcja format():

- pozwala na wyświetlenie znaku przy liczbie
- pozwala na ustalenie liczby miejsc po przecinku dla liczb zmiennoprzecinkowych
- pozwala na zachowanie odpowiedniego odstępu dla liczb oraz napisów

Typ	Znaczenie
'E', 'e'	Notacja naukowa
'F', 'f'	Liczby zmiennoprzecinkowe
'G', 'g', None	Automatycznie ustala czy wyświetlić coś jak 'f' czy ‚e' (tryb domyślny)
'D', 'd'	Liczby całkowite
'%'	Wartość procentowa
'S', 's'	Napis

https://docs.python.org/3/library/string.html#formatspec

Wrocław University of Science and Technology

Odczyt danych wejściowych - funkcja input()

→ `input()`

- Do pobierania danych wejściowych dostarczanych za pomocą klawiatury służy funkcja ***input()***
- Funkcja ***input()*** zwraca napis (literał znakowy), która aby wykorzystać go później należy przypisać do zmiennej
- Funkcja ***input()*** może pobierać dane w postaci napisu

```
>> anything = input("Tell me anything...")
>> print("Hmm...", anything, "... Really?")
```

- Aby zmienić typ danych (skonwertować dane) można wykorzystać funkcje ***int()*** oraz ***float()***

```
>> anything = float(input("Enter a number: "))
>> something = anything ** 2.0
>> print(anything, "to the power of 2 is", something)
```

Wrocław University of Science and Technology

Komentarze

- Komentarz – informacja umieszczona w kodzie źródłowym programu, która objaśnia jego działanie
- Komentarze są ignorowane przez interpreter
- Komentarze ułatwiają zrozumienie kodu
- W Pythonie komentarz to fragment tekstu, który zaczyna się od znaku # (krzyżyka) i rozciąga się do końca linii.

- Dobrzy, odpowiedzialni programiści opisują każdy ważny fragment kodu, np. wyjaśniając rolę zmiennych; chociaż trzeba stwierdzić, że najlepszym sposobem komentowania zmiennych jest nadawanie im jednoznacznych nazw – **samo komentujące się zmienne.**

```
1  # Ten program wylicza dlugosc przeciwprostokatnej trojkata prostokatnego
2  # a i b są jego przyprostokatnymi
3  a = 3.0
4  b = 4.0
5  c = (a ** 2 + b ** 2) ** 0.5  # korzystamy z ** aby podniesc zmienne do kwadratu
6  print("c =", c)
```

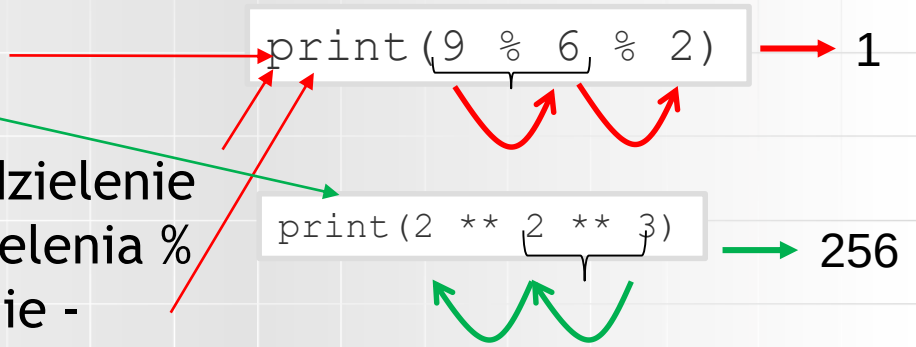
- Komentarze mogą być przydatne pod innym względem - możesz ich użyć do oznaczenia fragmentu kodu, który obecnie nie jest potrzebny z jakiegokolwiek powodu, np. podczas testowania.

```
1  # To jest test programu
2  x = 1
3  y = 2
4  # y = y + x
5  print(x + y)
```

Wrocław University of Science and Technology

Operatory arytmetyczne

1. Operatory unarne - i +
2. Wyrażenia w nawiasach
3. Potęgowanie **
4. Mnożenie *, dzielenie /, dzielenie całkowite //, reszta z dzielenia %
5. Dodawanie +, odejmowanie -



Działanie	Operator
Dodawanie	+
Odejmowanie	-
Mnożenie	*
Dzielenie	/
Dzielenie całkowitoliczbowe	//
Modulo	%
Potęgowanie	**

operacje na napisach: konkatenacja i powtarzanie

```
print("2" * 5) # "22222"
```

Uwaga!
Zwróć uwagę na typy wykorzystanych danych (zmiennych) aby móc określić typ wyniku.

Wrocław University of Science and Technology

Funkcje

- samodzielne definiowanie funkcji

definicja funkcji suma()

```
def suma(a, b):
    s = a + b
    return s
```

dwukropek

nagłówek funkcji suma()

ciało funkcji suma()
(wydzielone przez 4 spacje lub 1 tabulator)

nazwa funkcji

argumenty

słowa kluczowe

zwracane wartości

wywołanie funkcji suma()

```
suma(2,3)
s1 = 4
s2 = 5
suma(s1, s2)
```

```
>>> type(suma)
<class 'function'>
```

```
>>> type(print)
<class 'builtin_function_or_method'>
```

Funkcja to oddzielna część kodu komputerowego, która może:

- **wywołać jakiś efekt** (np. wysłać tekst do terminala, stworzyć plik, narysować obrazek, odtworzyć dźwięk itp.); jest to coś zupełnie niespotykanego w świecie matematyki;
- **obliczyć jakaś wartość** lub wartości (np. pierwiastek kwadratowy z wartości lub długość danego tekstu); to właśnie sprawia, że funkcje programistyczne są krewnymi pojęć matematycznych.

Wrocław University of Science and Technology

Funkcje

- przekazywanie argumentów

argumenty domyślne

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

```
print("a", "b", sep="***", end=" ")
```

- Przekazywanie argumentów względem ich pozycji (**argumenty pozycyjne**):
  - Kolejność przekazywanych argumentów musi opowiadać ich kolejności w definicji funkcji; wszelkie argumenty nazwane należy umieścić po ostatnim argumencie pozycyjnym.
- Przekazywanie argumentów względem ich nazwy (**argumenty nazwane**):
  - Przekazywany argument nazwany składa się z 3 części – nazwy argumentu, operatora przypisania (=) oraz wartości przypisywanej do tego argumentu

UWAGA!!!

- **argumenty** są znane jedynie wewnątrz funkcji, w której zostały wywołane,
- zmienne zdefiniowane wewnątrz funkcji (**zmienne lokalne**) ‚żyją' tylko w jej obrębie,
- **stałe nazwane** mogą być wykorzystane w każdym punkcie programu, ale mogą zostać **przykryte** wewnątrz funkcji (ang. shadowing),.

Wrocław University of Science and Technology

Funkcje

- zwracanie wartości i ich przypisywanie do zmiennych

Funkcja to oddzielna część kodu komputerowego, która może:

- **zwracać konkretne wartości** (instrukcja return <sth>)
- **wykonywać konkretne operacje** (pusta instrukcja return lub bez tego słowa kluczowego) – zwracać ***None*** (brak wartości)

UWAGA!!!

- zawsze możesz **zignorować** rezultat funkcji,
- jeśli funkcja ma **zwrócić jakąś wartość** powinna zawierać słowo kluczowe **return**.
- jeśli funkcja **ma zwrócić kilka wartości** należy rozdzielić je przecinkiem (i również odpowiednio przypisać ich wartości do zmiennych)

Wrocław University of Science and Technology

Projektowanie programu

- IPO, funkcja main()

Co musimy wiedzieć?

1. Jakie mamy dane wejściowe.
2. Co z nimi chcemy zrobić.
3. Jak przedstawiamy wynik.

```
def main():
  print("Zaczynamy.")
  wiadomosc()
  print("Kończymy.")
def wiadomosc():
  print("Jesteś w funkcji")
main()
```

Projektowanie programu można sprowadzić do dwóch kroków:

- określenie, jakie zadania ma wykonywać program,
- określenie kroków, za pomocą których program wykona to zadanie.

- Tworzenie funkcji głównej nie jest wymogiem Pythona
- Tworzenie funkcji głównej jest konwencją, ale wykorzystywanie konwencji zwiększa czytelność kodu

Wrocław University of Science and Technology

Rzeczy do zapamiętania!

1. Funkcja ***print()*** jest funkcją wbudowaną, która drukuje określony komunikat w oknie konsoli.
2. Funkcje wbudowane, w przeciwieństwie do funkcji zdefiniowanych przez użytkownika, są zawsze dostępne i nie trzeba ich importować - pełną listę można znaleźć w porządku alfabetycznym w bibliotece standardowej (https://docs.python.org/3.8/library/functions.html).
3. Aby wywołać funkcję, musisz użyć nazwy funkcji, po której następuje nawias, wewnątrz którego wypisujemy przekazywane argumenty.
4. Ciągi znaków są ujmowane w cudzysłowu lub apostrofach, np. ”Jestem łańcuchem” lub ’Ja też jestem łańcuchem’.
5. Programy komputerowe to zbiory instrukcji. Instrukcja jest poleceniem wykonania określonego zadania, np. wydrukowania określonej wiadomości na ekranie.
6. W napisach ukośnik (\) jest znakiem specjalnym, który informuje, że następny znak ma inne znaczenie, np. \n (znak nowej linii) rozpoczyna nowy wiersz.

Wrocław University of Science and Technology

Rzeczy do zapamiętania!

7. Argumenty pozycyjne to takie, których znaczenie jest podyktowane ich pozycją, np. drugi argument jest wyprowadzany po pierwszym, trzeci po drugim itd.
8. Argumenty nazwane to takie, których znaczenie nie jest podyktowane ich lokalizacją, ale specjalnym słowem (słowem kluczowym) używanym do ich identyfikacji.
9. Argumenty *end* i *sep* mogą być używane do formatowania danych wyjściowych funkcji *print()*. Argument *sep* określa separator między argumentami wyjściowymi (np. print("H", "E", "L", "L", "O", sep = "-"), podczas gdy argument *end* określa, co wydrukować na końcu instrukcji drukowania.
10. Literały to notacje reprezentujące pewne ustalone wartości w kodzie. W pythonie mamy różne typy literałów - na przykład literał może być liczbą (literały numeryczne, np. 123) lub ciągiem znaków (literały łańcuchowe/znakowe, np. ”Jestem literałem”).
11. System binarny to system liczbowy, w którym podstawą jest 2. Dlatego liczba binarna składa się tylko z zer i jedynek, np. 1010 to 10 w systemie dziesiętnym.

Wrocław University of Science and Technology

Rzeczy do zapamiętania!

12. Podobnie, systemy numeracji ósemkowej i szesnastkowej stosują odpowiednio 8 i 16 jako podstawy systemów. System szesnastkowy używa liczb dziesiętnych i sześciu kolejnych liter z alfabetu.
13. Liczby całkowite (**int**) są jednym z typów liczbowych obsługiwanych przez Pythona. Są to liczby zapisane bez części ułamkowej, np. 256 lub -1 (liczby całkowite ujemne).
14. Liczby zmiennoprzecinkowe (**float**) to kolejny typ liczbowy obsługiwany przez Pythona. Są to liczby, które zawierają (lub mogą zawierać) składnik ułamkowy, np.1,27.
15. Aby zakodować apostrof lub cudzysłów w łańcuchu znaków, możesz użyć ukośnika, np. *'I\'m happy'* lub wykorzystać innego typu znakowego, np. *"I'm happy".*
16. Wartości logiczne to dwa stałe obiekty ***True*** i ***False*** (w kontekście liczbowym 1 to **True**, a 0 to **False**).
17. Jest jeszcze jeden specjalny literał używany w Pythonie: literał **None**. Ten literał jest tak zwanym obiektem ***NoneType*** i jest używany do reprezentowania braku wartości.

Wrocław University of Science and Technology

Rzeczy do zapamiętania!

18. Operatory to specjalne symbole, które pozwalają na wykonanie szeregu operacji, np. operator mnożenia *, przemnaża dwie wartości.
19. Wyrażenia to kombinacja zmiennych, wartości i operatorów, np. 1 + 2.
20. Kolejność wykonywania działań w pythonie jest taka sama jak znana nam z matematyki: unarny + i -, potem potęgowanie – mnożenie, dzielenie, modulo – dodawanie i odejmowanie.
21. Operatory arytmetyczne działają od lewej do prawej z wyjątkiem operatora potęgowania - działa od prawej do lewej.
22. Operacje w nawiasach zawsze są wykonywane jako pierwsze.
23. Operatory dodawania + i mnożenia * działają także na napisach, odpowiednio łącząc i replikując je.
24. Zdefiniowane przez użytkownika funkcje mogą przyjmować dowolnie wiele argumentów (nawet żadnego).
25. Argumenty pozycyjne są wprowadzane w wywołaniu funkcji w zdefiniowanej kolejności, argumenty nazwane po ich nazwie.
26. Argument domyślny posiada pewną predefiniowaną nazwę zawartą w definicji funkcji.

Wrocław University of Science and Technology

Rzeczy do zapamiętania!

27. Argumenty pozycyjne muszą poprzedzać argumenty nazwane.
28. Zmienne zdefiniowane poza funkcją mogą na czas działania funkcji zostać przykryte przez zmienne o tej samej nazwie (chyba, że poprzedzi je słowo kluczowe global), zmienne zdefiniowane w funkcji są zmiennymi lokalnymi i istnieją tylko w jej obrębie.
29. Słowo kluczowe return pozwala funkcji na zwracanie wartości (może być ich kilka, oddzielonych przecinkiem), jeśli słowo return nie występuje w ciele funkcji bądź po tym słowie nic się nie pojawia to funkcja zwraca brak wartości (None).
30. Zwracany wynik można łatwo przypisać do zmiennej, można tę operacje też pominąć.
31. Korzystając z funkcji print() wykorzystuj funkcję format() do formatowania liczb, np. do wyświetlenia odpowiedniego zaokrąglenia.