PyStart #26 Wchodzimy w funkcje

Garść informacji

- ➔ Pomagają uporządkować kod, bloki kodu zostają nazwane.
- ➔ Unikamy powtórzeń, kod staje się reużywalny.
- ➔ Możemy współdzielić kod pomiędzy projektami.
- ➔ Trudniej wejść sobie w drogę w zespole
- ➔ Kod staje się możliwy do automatycznego testowania.
- ➔ DWIE ZASADY: funkcja powinna robić jedną rzecz, a jej nazwa powinna być nazwą czynności

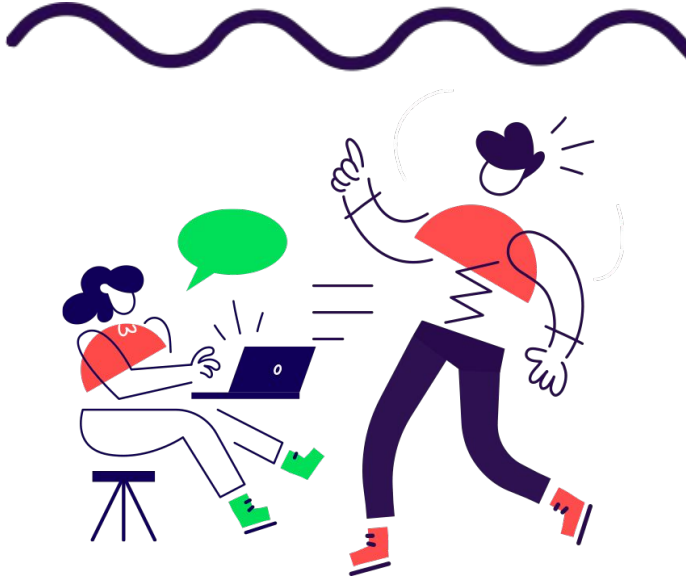
PyStart #26 Wchodzimy w funkcje

Słowo kluczowe def

- Oznacza deklarację funkcji, czyli jej “przedstawienie” w programie, samo z siebie nic nie “robi”
- Po słowie kluczowym def przychodzi nazwa funkcji.
  - musi być nazwą czynności
  - słowa rozdzielone _
  - nie mogą być takie jak nazwy innych identyfikatorów np. zmiennych lub innych funkcji

PyStart #26 Wchodzimy w funkcje

Jak to wygląda?



```
1  def add_numbers(a, b):
2      return a + b
3
```

Po deklaracji trzeba wywołać funkcję:

```
4
5  print(add_numbers(10, 5))
```

PyStart #26 Wchodzimy w funkcje

Co robi ten cały return?

- ➔ Zatrzymuje wykonywanie funkcji.
- ➔ Za jego pomocą funkcja może zwracać tylko jeden raz.
- ➔ Zwraca wartość, która znajduje się za nim.

PyStart #26 Wchodzimy w funkcje

Jak z returnem, a jak bez?

```
def policz_vat(netto, vat):
    brutto = netto * (1 + vat)
    print(brutto)

wartosc = policz_vat(100, 0.23)
print(wartosc) # None
```

```
def policz_vat(netto, vat):
    brutto = netto * (1 + vat)
    return brutto

wartosc = policz_vat(100, 0.23)
print(wartosc) # 123
```

PyStart #26 Wchodzimy w funkcje

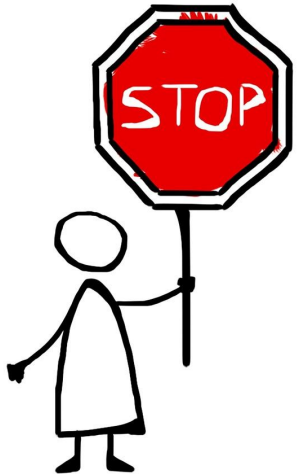
Tak nie wolno robić

```
1  from math import pi
2
3
4  def calculate_field(r):
5      result = pi * r ** 2
6      return result
7
8  print(result)
9  result = calculate_field(10)
10 print(result)
11
```

PyStart #26 Wchodzimy w funkcje

Zadania dla nabrania wprawy

1. Przygotuj funkcję o nazwie `word_counter`, powinna jako argument odbierać dowolny test i zwracać ilość słów z których ten tekst się składa.
2. Przygotuj funkcję, która będzie odbierała od użytkownika słowo, a następnie zwróci zbiór samogłosek znajdujących się w tym słowie.
3. Przygotuj funkcję, która zwróci `True` albo `False`, w zależności od tego czy liczba ta jest pierwsza.
   (Użyj algorytmu z jednej z poprzednich lekcji)



PyStart #27 Więcej o argumentach

Omówienie

```
1  def calculate_circle_area(r):
2      pass
3
4
5  def calculate_triangle_area(a, h):
6      pass
7
```

PyStart #27 Argumenty naszych funkcji

Argumenty domyślne

```
1  def calculate_rectangle_area(a=10, b=20):
2      pass
3
```

```
5  calculate_rectangle_area(20)
6
7  calculate_rectangle_area(20, 100)
8
9  calculate_rectangle_area()
10
```

PyStart #27 Argumenty naszych funkcji

Inny przykład

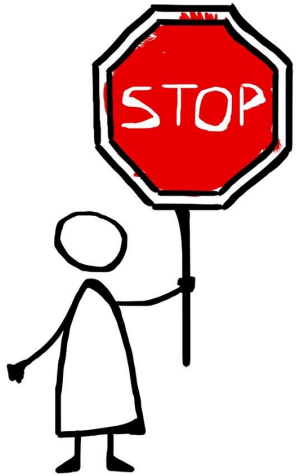
```
def send_email(recipient, content, sender=”kacper@dokodu.it”):
    …

def add_user(login, password, role=”admin”):
    …

def send_notification(type=”alert”, channel=”email”, when=”now”):
    …
```

PyStart #27 Wchodzimy w funkcje

Zadania dla nabrania wprawy

Przygotuj funkcję, która będzie szukała wspólnych dzielników dla dwóch liczb. Funkcja posiada trzeci argument domyślny określający wartość początkową, gdzie największy wspólne dzielniki muszą być zawsze większe niż ta wartość.

```
5   calculate_common_divisor(3, 6)
6   # zwróci [3]
7   calculate_common_divisor(3, 6, 4)
8   # zwróci None
9   calculate_common_divisor(16, 8)
10  # zwróci [2, 4, 8]
```

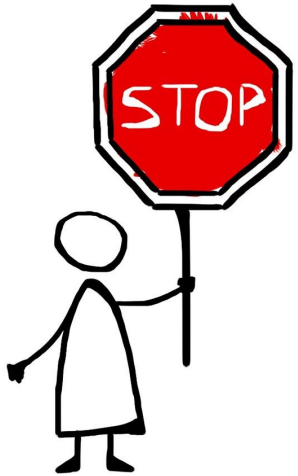


PyStart #27 Wchodzimy w funkcje

Zadania dla nabrania wprawy

Napisz funkcję, która przyjmuje dwa argumenty, gdzie drugi będzie miał domyślną wartością równą 2. Funkcja powinna zwracać podaną liczbę pomnożoną przez mnożnik.

```
multiply(2) # 4
multiply(2, 3) # 6
```



PyStart #28 Argumenty naszych funkcji

Argumenty pozycyjne

- Wszystkie argumenty jakie poznaliśmy do tej pory były argumentami pozycyjnymi
- Argument pozycyjny musi być przekazywany na konkretnej pozycji
- Możemy też przekazywać argumenty jako nazwane posługując się ich nazwami (patrz.. kolejny slajd)

PyStart #28 Argumenty naszych funkcji

Argumenty nazwane

```python
def calculate_brutto(netto, vat):
    return netto * (1 + vat)

calculate_brutto(netto=150, vat=0.23)
```

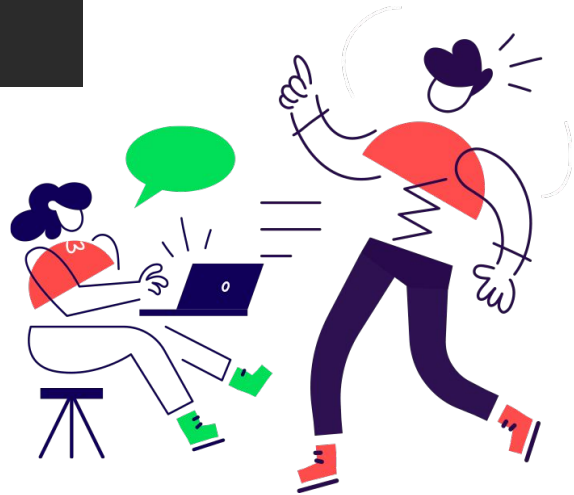
- ➔ Każdy argument domyślny może być podany wraz z nazwą
- ➔ Nie ma znaczenia kolejność argumentów nazwanych
- ➔ W ten sposób łatwiej jest rozkminić czego dotyczy przekazywany argument

PyStart #28 Argumenty naszych funkcji

Argumenty pozycyjne i nazwane

```python
def calculate_brutto(quantity, netto, vat=0.23):
    return quantity * netto * (1 + vat)

calculate_brutto(10, netto=150, vat=0.23)
calculate_brutto(10, vat=150, netto=0.23)
```

- ➔ Argument pozycyjny idzie na pierwszy ogień
- ➔ Kolejność argumentów pozycyjny musi być zachowana
- ➔ Argumenty nazwane mogą być pomijane jeśli mają wartości domyślne

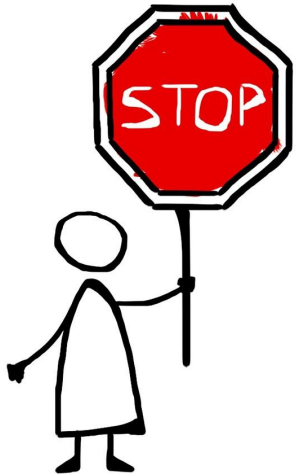


PyStart #28 Wchodzimy w funkcje

Zadania dla nabrania wprawy

Napisz funkcję, która przyjmuje jeden argument pozycyjny (tekst) i dwa argumenty nazwane (delimiter i repeat) z domyślnymi wartościami "," i 1. Funkcja powinna zwracać tekst powtórzony odpowiednią liczbę razy i rozdzielony podanym delimiterem.

```
# Ala ma kota
repeat(‘Ala ma kota’)

# python;python;python
repeat(‘Python’, delimiter=’;’, repeat=3)
```



PyStart #29 Co to jest rekurencja?

Wstęp / wyjaśnienie

PyStart #29 Co to jest rekurencja?

Wstęp / wyjaśnienie

PyStart #29 Co to jest rekurencja?

Wstęp / wyjaśnienie

1. weź lalkę do ręki
2. sprawdź czy możesz ją otworzyć
3. jeśli nie to zakończ
4. jeśli tak to weź lalkę do ręki
5. sprawdź czy możesz ją otworzyć
6. jeśli nie to zakończ
7. jeśli tak to weź lalkę do ręki
8. ... 😊

PyStart #29 Co to jest rekurencja?

Inne przykłady

1. Szukanie pliku po nazwie
2. Indeksowanie plików
3. Kopiowanie folderów
4. Wyświetlanie menu z podkategoriami
5. Przeglądanie dowolnego grafu
6. Pętle, są języki które ich nie mają zamiast tego używa się rekurencji.

PyStart #29 Co to jest rekurencja?

Jak obliczyć silnię?

5! = 5 * 4 * 3 * 2 * 1

4! = 4 * 3 * 2 * 1

3! = 3 * 2 * 1

2! = 2 * 1

1! = 1

0! = 1

PyStart #29 Co to jest rekurencja?

Jak obliczyć silnię?

5! = 5 * 4 * 3 * 2 * 1	5! = 5 * 4!
4! = 4 * 3 * 2 * 1	4! = 4 * 3!
3! = 3 * 2 * 1	3! = 3 * 2!
2! = 2 * 1	2! = 2 * 1!
1! = 1	1! = 1 * 0!
0! = 1	0! = 1

PyStart #29 Co to jest rekurencja?

Jak obliczyć silnię?

5! = 5 * 4 * 3 * 2 * 1

4! = 4 * 3 * 2 * 1

3! = 3 * 2 * 1

2! = 2 * 1

1! = 1

0! = 1

5! = 5 * 4!

4! = 4 * 3!

3! = 3 * 2!

2! = 2 * 1!

1! = 1 * 0!

0! = 1

L! = L * (L-1)!

silnia(L) = L * silnia(L-1)

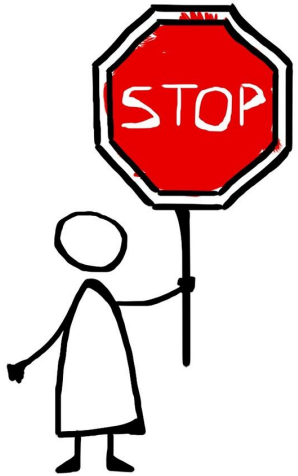
PyStart #29 Co to jest rekurencja?

Zadania dla nabrania wprawy

Korzystając ze zdobytej wiedzy napisz funkcję liczącą silnie dla podanej przez użytkownika w argumencie liczby.

L! = L * (L-1)!

silnia(L) = L * silnia(L-1)



PyStart #30 Pydoc - dokumentacja funkcji

Dlaczego to istotne?

- ➔ Dokumentacja jest dostępna w formie czytelnego tekstu.
- ➔ Dokumentowanie kodu pozwala innym programistom łatwiej zrozumieć jakie dane przekazać do funkcji.
- ➔ Poza nazwą funkcji można również dodawać do funkcji opis dzięki czemu wiadomo co ta funkcja robi.
- ➔ Najlepiej jest pisać dokumentację funkcji po angielsku.
- ➔ Można też automatycznie sprawdzać pisany kod za pomocą `mypy`

PyStart #30 Pydoc - dokumentacja funkcji

Dodawanie typów do funkcji

```
def add(a:int, b:int) -> int:


def sum_lists(a:list, b:list) -> list:


def print_dict(data:dict):


def get_student(student_id:int, with_notes:bool) -> Student:
```

PyStart #30 Pydoc - dokumentacja funkcji

Piszmy Pydoc

```
def add(a:int, b:int) -> int:
    """
    This function adds two numbers together
    :param a: The first number
    :param b: The second number
    :return: The sum of the two numbers
    """
    return a + b
```

""""" ⟸ trzy razy cudzysłów

```
Opis funkcji
:parametr jego_nazwa: Opis parametru
:parametr inna_nazwa: Opis tego parametru
:return: Informacja co funkcja zwraca
"""""
```

PyStart #30 Argumenty naszych funkcji

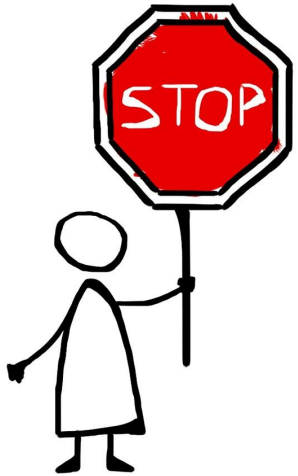
Zadania dla nabrania wprawy

1. Przygotuj funkcję, która na podstawie czasu pracy, ilości zużywanych kilowatogodzin odpowie ile zapłacę za prąd, który zużywa urządzenie. Domyślny koszt 1 kwh to 0,617.

2. Przygotuj funkcję, której deklaracja będzie wyglądała następująco:

   ```
   def count_numbers(numbers: list, count_odd:bool = True, count_even:bool = True):
   ```

   Zadaniem funkcji będzie odpowiedź na pytanie ile jest liczb spełniających podane w argumentach wymagania w liście przekazanej w pierwszym przedziale.



PyStart #31 Argumenty naszych funkcji

A co jeśli chcemy zwrócić więcej danych?

```
2   def get_car_details():
3       return {
4           'brand': 'Fiat',
5           'model': '126p',
6           'nickname': 'Maluch'
7       }
8
9   def get_car_size():
10      width = 1377
11      height = 1335
12      length = 3054
13
14      return width, height, length
15
16  car_width, car_height, car_length = get_car_size()
17
```

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

➔ Przygotuj funkcję, która odbierze dwie listy, wynikiem powinna być nowa lista, której elementami będą sumy. dla przykładu:

```
5   a = [2, 4, 6]
6   b = [8, 6, 4]
7
8   sum_lists(a, b)
9   # zwróci [10, 10, 10]
```

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

- ➔ Przygotuj funkcję, która będzie potrafiła przefiltrować liczby parzyste z listy przekazanej w argumencie.
- ➔ Przygotuj funkcję, która przefiltruje tylko słowa, których długość jest większa od 4 i mniejsza od 8.
- ➔ Przygotuj funkcję, która zliczy ilość znaków w tekście zawierających się wewnątrz nawiasów okrągłych. Nawiasy mogą występować w tekście wielokrotnie, nigdy nie będą się w sobie zawierać.

```python
def count_letters(text, start='(', end=')'):
    pass


count_letters('(ala) ma (kota)')
# zwróci 3 + 4

count_letters('<> kod <103>', '<', '>')
# zwróci 3

count_letters('abrakadabra')
#zwróci 0
```

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

➔ **Przygotuj funkcję, która odbierze od użytkownika procent zdobytych punktów, a w odpowiedzi zwróci ocenę jaką otrzymał użytkownik.**

Minimum procentowe	Ocena
<45%	niedostateczny
45%	dopuszczający
55%	dostateczny
80%	dobry
90%	bardzo dobry
95%	celujący

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

- Przygotuj funkcję, która odbierze dwa punkty w postaci tupli (x, y). Wynikiem powinna być długość odcinka utworzonego w wyniku połączenia tych dwóch punktów.

$$|AB| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

- Przygotuj funkcję, której zadaniem będzie konwersja jednostek: mil na kilometry i drugą wykonującą konwersję kilometrów na mile

- Przygotuj funkcję, która dla dowolnej listy zwróci słownik zawierający:
  total - ilość elementów
  mean - średnią elementów
  max - największą wartość
  min - najmniejszą wartość

- Napisz funkcję, która odbierze tekst, a następnie każde jego słowo zapisze w odwrotnej formie np. programuję w Pythonie zamieni na ejumargorp w einohtyP

- Napisz funkcję, która odbierze dowolną liczbę i zwróci jej liczby w kolejności malejącej np. 12345 zamieni na 54321, a a 1998 na 8991

PIĄTA PRACA DOMOWA

UWAGA! UWAGA!

- Napisz funkcję, która zamieni każdy jeden dodatni element listy na ujemny, a ujemny na dodatni. Wykorzystaj skróconego ifa.

  [1, 2, 3, -4] => [-1, -2, -3, 4]

- Przygotuj funkcję, która będzie zamieni wszystkie elementy listy na stringi oraz drugą, która zamieni każdy element listy na integer.

- Napisz funkcję, która rekurencyjnie policzy sumę wszystkich wartości od 1 do n.. dla przykładu sum_it(5) wykona 5 + 4 + 3 + 2 + 1

- PIN może składać się wyłącznie z czterech cyfr. Napisz funkcję która zwróci prawdę lub fałsz w zależności od tego czy to wymaganie jest spełnione.