O prowadzącym

Kamil Musiał

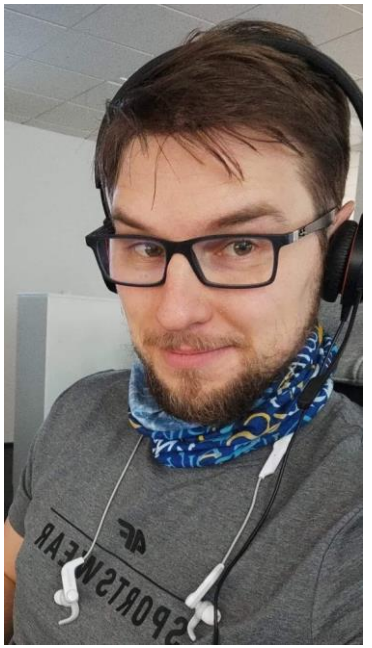
Tester w Tieto (wcześniej 7 lat tester / integrator / verification specialist w Nokia)

certyfikowany tester ISTQB (full advanced)

od 5 lat wykładowca WSB Wrocław (testowanie, telekomunikacja, sieci, IoT, Python)

doktorant Politechnika Wrocławska

fan morsowania zanim to było modne

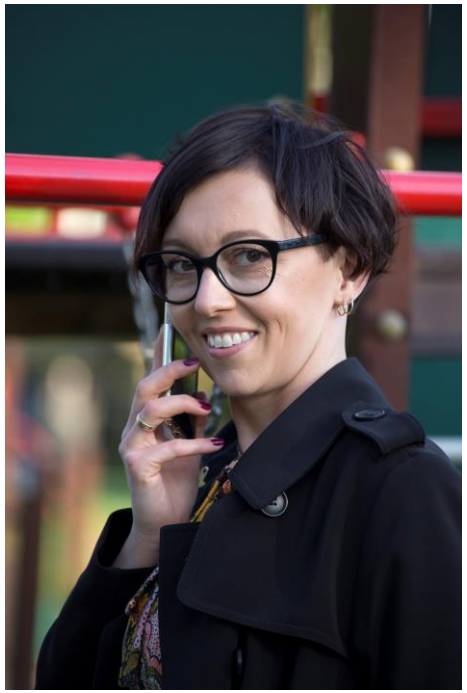


kamil.musial@akademia.altkom.pl
kamil.musial@wroclaw.merito.pl

O prowadzącej

Natalia Kniat

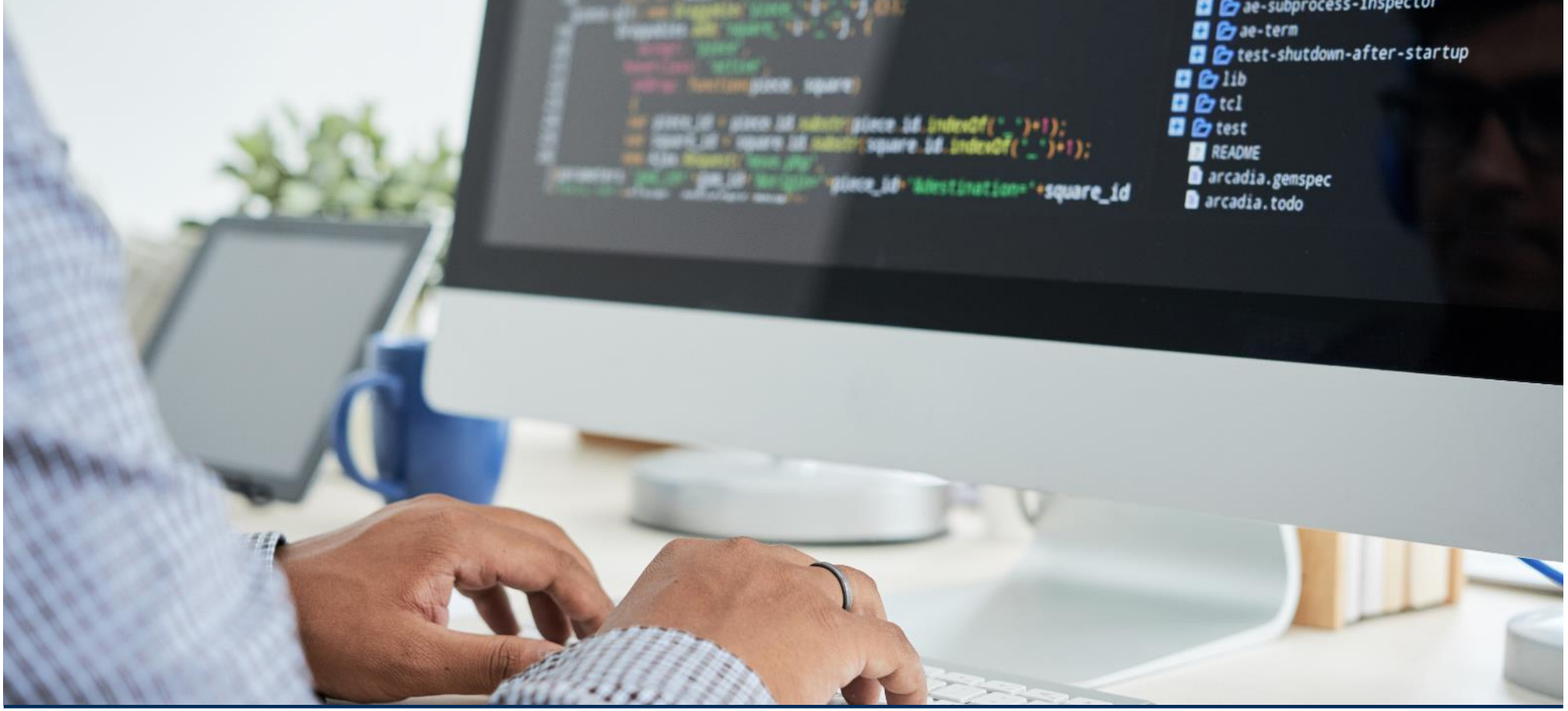
- CI Lead / testerka / integratorka w Nokia
- certyfikowana testerka ISTQB
- trenerka Nokia Academy (testowanie, telekomunikacja, Python)
- zorganizowana mama z zamiłowaniem do wyrobów własnych



natalia.kniat@akademia.altkom.pl
natalia.kniat@wroclaw.merito.pl

Agenda

1. Powtórka
2. Pętla *while* i *while True*
3. Funkcje
4. Słowniki
5. Zbiory

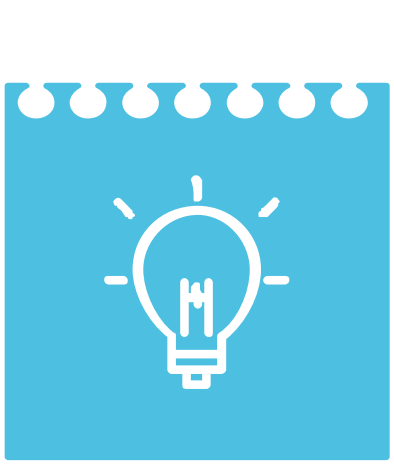


Powtórka

- Zmienne: *integer*, *float*, *string*
- Operacje na stringach
- Listy
- Pętla *for*
- Instrukcje warunkowe
- Rzutowanie
- Zaokrąglanie

Powtórka

Zadanie



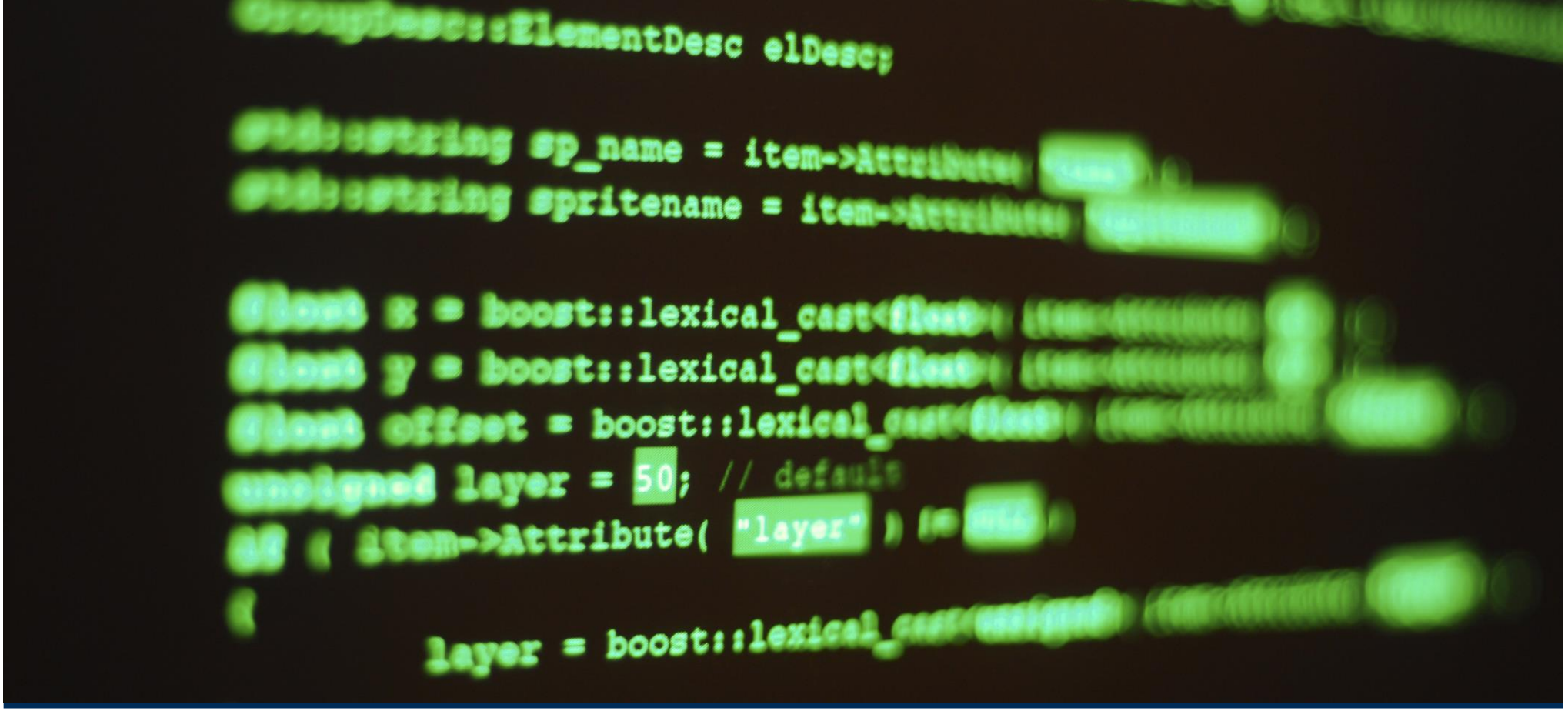
01

Dana jest lista zarobków brutto:

```
zarobki = ['5001PLN', '972EUR', '1043USD', '1897PLN',
'693EUR', '790USD’]
```

Napisz program, który:

- przeliczy zarobki na PLN,
- obliczy należny podatek (10% powyżej 3000 zł i 20% powyżej 5000 zł),
- zapisze w nowej liście zarobki netto w PLN,
- wypisze odpowiednie komunikaty.



Python

Pętla *while* oraz *while True*, funkcja, słownik, zbiór

Pętla *while*

Powtarzanie danej akcji, tak długo, jak warunek jest spełniony.

Przykłady:

1. Dodawaj elementy listy tak długo, aż nie skończy się lista.
2. Silnia – mnóż elementy tak długo, aż dojdziemy do finalnej wartości.
3. Szukaj rozwiązania tak długo, aż znajdziesz rozwiązanie spełniające dany warunek.

Pętla *while True*

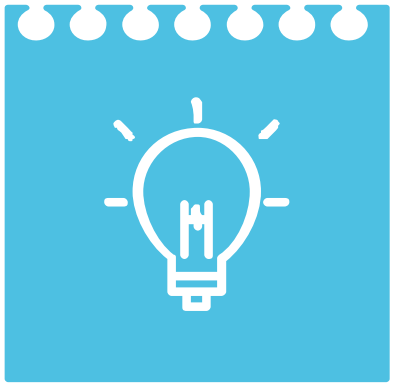
Powtarzanie danej akcji, aż do informacji o przerwaniu działania.

Przykłady:

1. Wprowadzenie hasła.
2. Weryfikacja poprawności danych.

Pętle

Zadanie



01

Napisz program, który przyjmuje numer PESEL i weryfikuje jego poprawność.

Program pyta tak długo, aż zostanie wprowadzony poprawny PESEL bądź użytkownik wyjdzie z programu.

Przyjmij, że:

- *PESEL jest czterocyfrowy,*
- *w poprawnym numerze PESEL suma cyfr jest podzielna przez 3 lub przez 4.*

Pętla while - zadanie

Napisz program, który w ustalonej posortowanej liście wyszukuje indeks konkretnego elementu.

lista = [-88, -3, 11, 23, 56, 78, 98, 154, 176, 198, 275, 375, 416, 524, 564, 627, 738, 873, 924]

Wejście:

- Wartość elementu do znalezienia

Wyjście:

- Indeks na którym indeksie listy jest ten element, lub jeśli go nie ma informacja o tym na jakim indeksie by się znajdował

Wymagania:

Użyj wyszukiwania binarnego z użyciem pętli while

Pętla while - zadanie

Napisz program, który czeka aż będzie pełna minuta i o tym poinformuje.

Napisz program, który czeka aż pojawi się określony plik i o tym poinformuje.

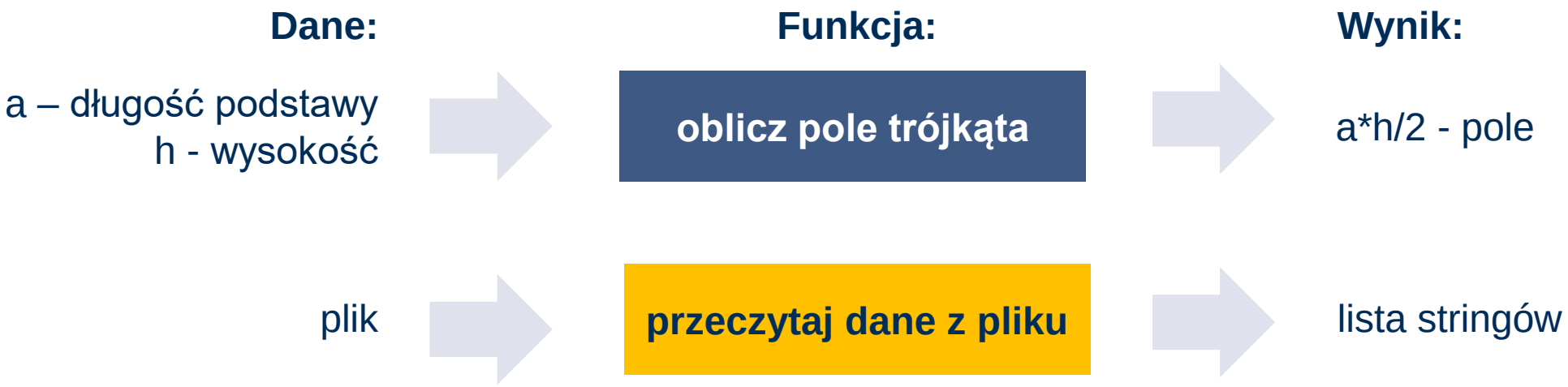
Pętla while – zadanie rozszerzenie

Niech poprzedni program sam generuje losową listę o zadanej długości, posortuje ją i dokona wyszukiwania.

Porównaj czasu wyszukiwania binarnego vs. Wyszukiwania liniowego/brute force w zależności od długości listy.

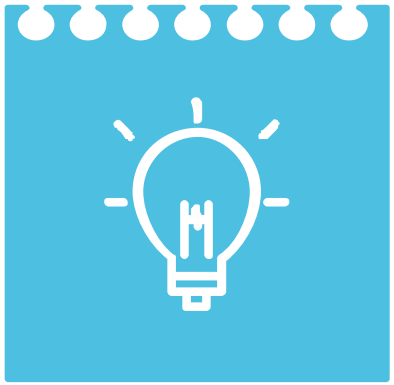
Funkcja

Część programu, zgrupowany kod programu odpowiadający za pewną funkcjonalność (możliwość wielokrotnego użycia).



Funkcja

Zadania



01 Napisz funkcję, która przyjmuje metraż, wiek oraz typ budynku i zwraca informację, czy można w nim mieszkać.

02 Napisz funkcje, która przyjmuje dane użytkownika i zwraca informację, czy użytkownik może dalej używać programu.

03 Napisz funkcję, która pozwoli użytkownikowi zmienić hasło użytkownika (jeśli zna poprawne hasło).

Funkcja zadania

Napisz funkcje która oczekuje aż użytkownik poda liczbę z zadanego zakresu. Użyj tej funkcji w programie, który pyta o datę urodzenia i oblicza ile ktoś ma lat, miesięcy, dni.

Napisz funkcje która oczekuje aż użytkownik zgodzi się na warunki działania programu, jeśli odp użytkownika jest „tak” to się zgodził i program ma działać dalej. Jeśli użytkownik odp „nie” program ma się zakończyć. Każda inna odp ma powodować powtórzenie pytania. Dodaj tą funkcję do poprzedniego programu.

Słownik (typ zmiennych)

Słownik zawiera pary klucz - wartość; klucze muszą być unikalne.

Tworzenie słownika – składnia (UWAGA! nawiasy klamrowe i dwukropek):

```
nazwa_słownika = { klucz1: wartość1 , klucz2: wartość2 , ... }
```

Przykład:

```
kolory_napojow = { 'kawa': 'biala', 'herbata' : 'zielona', 'cola' : 'czarna'}
```

`slownik.keys()`	zwraca klucze w słowniku
`slownik.values()`	zwraca wartości w słowniku
`slownik.items()`	zwraca klucz i wartość
„klucz" in słownik	zwróci prawdę, jeśli klucz występuje w słowniku.

Słownik (typ zmiennych)

Słownik zawiera pary klucz - wartość; klucze muszą być unikalne.

Tworzenie słownika – składnia (UWAGA! nawiasy klamrowe i dwukropek):

```
nazwa_słownika = { klucz1: wartość1 , klucz2: wartość2 , ... }
```

Przykład:

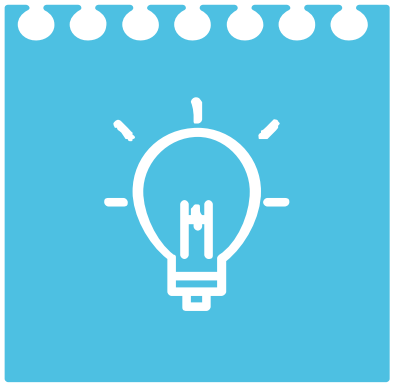
```
kursy_walut_do_PLN = {'EURO' : 4.6, 'USD' : 4.5, 'BTC' : 1000000}
```

Odwołanie do elementu słownika - analogicznie jak w listach (rolę “indeksu” pełni zdefiniowany klucz):

```
print(kursy_walut_do_PLN['EURO’])      >>>> 4.6
print(kursy_walut_do_PLN['dolar’])      >>>> błąd, brak klucza
print(kursy_walut_do_PLN[1])            >>>> błąd, brak klucza
```

Słownik (typ zmiennych)

Zadania



01

Rozwiń poprzedni program.
Program pozwala zmienić hasło użytkownika – pobiera hasło ze słownika i porównuje.

02

Napisz program, który przeczyta plik (txt) i policzy ile raz powtarza się dane słowo.

Zbiór (typ zmiennych)

Rzutowanie

Zbiór zawiera nieuporządkowane elementy bez powtórzeń.

Tworzenie słownika – składnia (UWAGA! nawiasy klamrowe):

```
nazwa_zmiennej = {element1, element2, element3}
```

Przykład:

```
zbior = {1, 2, 4}
```

Zbiór (typ zmiennych)

Rzutowanie

Operacje na zbiorach:

- `print("Suma:",A.union(B))` wszystkie elementy zbiorów A i B
- `print("Część wspólna:",A.intersection(B))` część wspólna zbiorów A i B
- `print("Różnica A-B:", A.difference(B))`
- `print("Różnica B-A:", B.difference(A))`
- `print("Różnica symetryczna:", A.symmetric_difference(B))`

Python

Zadania

Zadanie

Załóżmy, że pesel ma 4 cyfry.

- stwórzmy zbiór `NFZ` – ludzie w bazie pacjentów NFZ,
- stwórzmy zbiór `chorzy_rok` – ludzie chorzy w ostatnim roku,
- stwórzmy zbiór `chorzy_miesiac` – ludzie chorzy w ostatnim miesiącu,
- stwórzmy zbiór `krzyki` – wszystkich ludzi mieszkających na Krzykach,
- stwórzmy zbiór `centrum` – wszystkich ludzi mieszkających w Centrum.

```
NFZ = {1234, 3476, 4544, 3423, 3254, 8769, 5436, 2345, 6532, 1243, 6435, 1298, 6732,
7688, 7648, 2345, 2356}

chorzy_rok = set([1234, 3476, 4544, 3423, 3254, 8769, 5436])

chorzy_miesiac = set([1234, 3476, 3098, 4544, 3423])

krzyki = {4544, 3423, 3254, 8769, 5436, 2345, 6532, 1243}

centrum = {7648, 2345, 2356, 3987, 1234, 3476, 3254}

zbior_pusty = set()
```

Zadanie

1. Sprawdźmy:
   a) ile jest osób, które chorowały w ostatnim roku na Krzykach.
   b) ile jest osób z Krzyków chorowało w ostatnim roku.
   c) ile osób chorowało w ostatnim miesiącu w centrum.
   d) ile osób mieszka w sumie w centrum i na krzykach.
2. Sprawdźmy poprawność danych:
   a) każdy, kto chorował w ostatnim miesiącu, jednocześnie chorował w ostatnim roku.
   b) nikt nie powinien mieszkać jednocześnie w Centrum i na Krzykach – jeśli tak, trzeba usunąć.
   c) każdy: chory, zdrowy, z Krzyków i z Centrum, powinien być w bazie NFZ, jeśli nie ma, trzeba dopisać.
3. Żeńskie Numery PESEL mają ostatnią cyfrę parzystą, męskie – nieparzystą. Zróbmy nowe zbiory, osobne dla mężczyzn i kobiet (w NFZ).

Zadanie

4. Wypiszmy:
   a) kobiety z Krzyków, które były chore w ostatnim roku.
   b) wypiszmy mężczyzn z Centrum, którzy NIE byli chorzy w ostatnim roku.
5. Napiszmy program przyjmujący PESEL i zwracający wszystkie dostępne informacje o tym peselu.
6. Zabezpieczmy program:
   a) wszystkie dane, które wprowadzamy muszą być poprawne.
   b) sprawdźmy, czy wszystkie numery PESEL w zbiorach są poprawne (czterocyfrowe liczby).

Zadanie

7. Program lista rzeczy do zrobienia. Program ma oferować opcje:
    1. Dodaj rzecz do zrobienia
    2. Oznacz rzecz jako zrobioną
    3. Pokaż rzeczy do zrobienia i zrobione
    4. Usuń rzecz do zrobienia
    5. Ustaw konkretną rzecz na początek
    6. Wyjdź z programu

8. Dla zadanego tekstu zlicz występowanie każdego słowa. Wykorzystaj defaultdict.

WYŻSZA SZKOŁA BANKOWA
Warszawa

Dziękujemy za uwagę!

01.09.2023