altkom akademia

PYTH01

Podstawy programowania w języku Python poziom I

Ćwiczenia

Altkom Akademia S.A., materiały własne

Opracowane ćwiczenia pozostawiają uczestnikowi szkolenia swobodę wyboru sposobu ich realizacji.

Przystępując do wykonania ćwiczeń możesz zdecydować, czy chcesz je wykonać w oparciu o własne pomysły, czy też postępować według przygotowanego i sprawdzonego algorytmu.

Pamiętaj, że w każdym momencie wykonywania ćwiczenia możesz poprosić instruktora o pomoc oraz o dostarczenie wzorcowego rozwiązania.

1. WPROWADZENIE DO JĘZYKA PYTHON

ĆWICZENIE 1.1:

Instalacja środowiska z konsolą interaktywną

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - samodzielnego przygotowania środowiska do nauki i tworzenia aplikacji w Pythonie (konsoli interaktywnej)
  - weryfikacji, czy i jaka wersja Pythona jest zainstalowana

CELE I ZADANIA:

- Pobierz z Internetu i zainstaluj najnowszą wersję Pythona
- Sprawdź, czy i jakie wersje Pythona są zainstalowane
- Wyświetl *the Zen of Python*

ALGORYTM WYKONANIA:

- Ze strony `https://www.python.org/downloads/` pobierz instalator Pythona lub użyj pliku instalacyjnego dostarczonego przez instruktora
- Uruchom program instalacyjny i wskaż katalog, gdzie ma być zainstalowany Python
- Upewnij się, czy i jakie katalogi Pythona zostały dodane do zmiennej systemowej *PATH*
- Po zakończeniu instalacji, w wierszu linii poleceń wydaj polecenie sprawdzające, jakie wersje Pythona są zainstalowane
- Uruchom konsolę interaktywną Pythona
- Zapoznaj się z podstawowymi poleceniami
- Wyświetl listę aforyzmów określających filozofię Pythona (tzw. *the Zen of Python*)
- W jaki sposób można opuścić ten tryb?

ĆWICZENIE 1.2:

Instalacja IDE

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność samodzielnego przygotowania środowiska do nauki i tworzenia aplikacji w Pythonie (IDE)

CELE I ZADANIA:

- Pobierz z Internetu i zainstaluj najnowszą wersję PyCharm IDE dla Pythona

ALGORYTM WYKONANIA:

- Ze strony `https://www.jetbrains.com/pycharm/` pobierz IDE dla Pythona (*PyCharm*, wersja *Community*) lub użyj pliku instalacyjnego dostarczonego przez instruktora
- Uruchom program instalacyjny i zaznacz odpowiednie opcje konfiguracyjne
- Uruchom zainstalowane IDE
- Sprawdź, czy z poziomu IDE działa konsola Pythona
- Utwórz nowy projekt Pythona

2. PODSTAWOWE KONCEPCJE

ĆWICZENIE 2.1:

Praca z dokumentacją – kostka do gry

UMIEJĘTNOŚCI:

- Wykonanie ćwiczenia pozwoli na zapoznanie się, jak korzystać z dokumentacji

CELE I ZADANIA:

- Uruchom dokumentację Pythona
- Zapoznaj się z jej strukturą i opcjami
- Korzystając z dokumentacji napisz prosty program symulujący rzuty kostką do gry
- Uruchom go w konsoli, a następnie w IDE

ALGORYTM WYKONANIA:

- Uruchom dokumentację Pythona łącząc się ze stroną `https://docs.python.org/3/index.html`
- Zapoznaj się z dostępnymi informacjami
- Przejdź do dokumentacji biblioteki standardowej (link *Library Reference*)
- Korzystając z dokumentacji napisz program, który będzie symulował rzuty kostką do gry – w wyniku rzutu może wypaść od 1 do 6 oczek
- W sekcji modułów numerycznych i matematycznych odszukaj moduł umożliwiający generowanie liczb pseudolosowych
- Zapoznaj się z jego opisem i zastanów się, która z funkcji umożliwiłaby wylosowanie liczby całkowitej z zadanego przedziału
- Jak zastosować tę funkcję?
- Aby móc wykorzystać możliwości modułu, wpisz jako pierwszą instrukcję:
  `import` *`nazwa-modułu`*
- Nazwę wywołanej funkcji poprzedź nazwą modułu i kropką
- Sprawdź działanie programu w konsoli interaktywnej
- Uruchom ten sam program w zainstalowanym IDE

ĆWICZENIE 2.2:

Lata przestępne

UMIEJĘTNOŚCI:

- Wykonanie ćwiczenia pozwoli na zapoznanie się z różnymi typami operatorów

CELE I ZADANIA:

- Napisz program, określający, czy podany rok jest przestępny
- Wprowadź testowany rok z klawiatury
- Wypisz wynik na ekranie

ALGORYTM WYKONANIA:

- Napisz program, określający, czy podany rok jest przestępny
- Wprowadź z klawiatury rok
- Reguła przestępności:

  Rok jest przestępny, jeśli jest wielokrotnością 4, z wykluczeniem lat będących wielokrotnościami 100, chyba, że są wielokrotnościami 400 (te są przestępne)
- Zastanów się, jakie dwa niezależne warunki muszą być spełnione, aby rok był przestępny?
- Do zapisu warunków logicznych użyj operatora równości == oraz różności *!=*
- Zapisz powyższą regułę w postaci pojedynczego wyrażenia logicznego
- Oblicz wyrażenie, a wynik przedstaw na ekranie
- Przetestuj działanie programu dla różnych wartości lat, np.:

  rok 2000 – przestępny (podzielny przez 400)
  rok 1900 – nieprzestępny (podzielny przez 100 i niepodzielny przez 400)
  rok 2020 – przestępny (podzielny przez 4 i niepodzielny przez 100)
  rok 2019 – nieprzestępny (niepodzielny przez 4)

ĆWICZENIE 2.3:

Stan lokaty

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych i operatorów
  - interakcji programu z użytkownikiem (wczytywania danych z klawiatury i wypisywania wyników na ekranie)

CELE I ZADANIA:

- Napisz program, który wyliczy stan lokaty bankowej przy stałym oprocentowaniu po upływie zadanej ilości lat
- Wczytaj wszystkie niezbędne dane wejściowe z klawiatury
- Wypisz na ekranie końcowy stan na koncie

ALGORYTM WYKONANIA:

- Napisz program, który wyliczy stan lokaty bankowej przy stałym oprocentowaniu w skali roku, po upływie zadanej ilości lat (zakładamy, że po upływie każdego pełnego roku odsetki są dopisywane do kwoty wyjściowej)
- Zastanów się, jakie dane wejściowe będą potrzebne?
- Jakich typów będą te dane?
- Wprowadź niezbędne dane z klawiatury
- Pamiętaj o dokonaniu konwersji wczytanych danych na odpowiednie typy
- Napisz wyrażenie wyliczające stan lokaty
- Zastanów się, jak wynik końcowy zaokrąglić do dwóch miejsc w części ułamkowej
- Wypisz wynik na ekranie

ĆWICZENIE 2.4:

Losowanie lotto

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - pracy z dokumentacją
  - wyszukiwania potrzebnych informacji
  - korzystania z wbudowanych bibliotek

CELE I ZADANIA:

- Napisz program, który wyliczy szansę wylosowania k liczb spośród n różnych liczb (jak w lotto)

ALGORYTM WYKONANIA:

- Wczytaj z klawiatury dane wejściowe:
  - k – ilość skreślanych liczb (np. 6)
  - n – całkowita ilość liczb spośród których skreślamy (np. 49)
- Liczbę możliwych kombinacji wyboru k spośród n różnych liczb opisuje wzór:

$$\frac{n!}{k! * (n-k)!}$$

  gdzie: $m!$ oznacza silnię liczby m
- Odszukaj dokumentację modułu matematycznego, a następnie funkcję wyliczającą silnię (możesz też posłużyć się prezentacją)
- Podobnie, jak to robiłeś w ćwiczeniu 2.1, dodaj w skrypcie instrukcję importu modułu, a nazwę funkcji poprzedź nazwą modułu i kropką
- Korzystając z funkcji zastosuj powyższy wzór, a wynik przedstaw na ekranie

ĆWICZENIE 2.5:

BMI – Body Mass Index

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z instrukcji warunkowych
  - prezentacji danych na ekranie

CELE I ZADANIA:

- Napisz program, który wyliczy wartość indeksu masy ciała (*BMI – Body Mass Index*)
- Wczytaj wzrost (w metrach) i wagę (w kg)
- Wylicz indeks i wypisz diagnozę
- Dodatkowo określ przedział prawidłowej wagi

ALGORYTM WYKONANIA:

- Wczytaj z klawiatury:
  - wzrost (w metrach)
  - wagę (w kg)
- Wartość indeksu opisuje wzór:

$$BMI = \frac{waga}{wzrost^2}$$

- W oparciu o wyliczoną wartość postaw diagnozę (wykorzystaj do tego celu instrukcję warunkową):
  - $<$ 16.00 – wygłodzenie
  - 16.00–16.99 – wychudzenie
  - 17.00–18.49 – niedowaga
  - 18.50–24.99 – wartość prawidłowa
  - 25.00–29.99 – nadwaga
  - 30.00–34.99 – I stopień otyłości
  - 35.00–39.99 – II stopień otyłości (otyłość kliniczna)
  - $\geqslant$ 40.00 – III stopień otyłości (otyłość skrajna)
- Na ekranie wypisz wartość wyliczonego indeksu oraz diagnozę
- Odszukaj sposób zaokrąglenia wartości indeksu do 2 cyfr w części ułamkowej
- Dodatkowo podaj przedział prawidłowej wagi dla określonego wzrostu

ĆWICZENIE 2.6:

Wieczny kalendarz

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych i operatorów
  - stosowania instrukcji warunkowych

CELE I ZADANIA:

- Napisz program, który pełni rolę wiecznego kalendarza – na podstawie podanej daty (dnia, miesiąca i roku) wyznacza nazwę dnia tygodnia
- Wykorzystaj algorytm opracowany przez Mike'a Keith'a
- Elementy daty wczytaj z klawiatury

ALGORYTM WYKONANIA:

- Napisz program, który dla podanej daty zwróci nazwę dnia tygodnia
- Wczytaj z klawiatury: dzień, miesiąc i rok
- Do wyliczenia dnia tygodnia wykorzystaj algorytm opracowany przez Mike'a Keith'a:

  Jeśli przyjmiemy, że:
  - d oznacza dzień miesiąca (1..31)
  - m oznacza miesiąc (1..12)
  - y oznacza rok
  - *calk(x)* oznacza część całkowitą liczby x
  - *mod* oznacza operator modulo (resztę z dzielenia całkowitego)

  to należy obliczyć:
  - rok z poprawką: z = jeśli $m < 3$ to: y - *1*, w przeciwnym razie: y
  - korektę: c = jeśli $m < 3$ to: *0*, w przeciwnym razie: *2*
  - wyrażenie:
    *(calk(23*m/9) + d + 4 + y + calk(z/4) - calk(z/100) + calk(z/400) - c) mod 7*

  Interpretacja wyniku wyrażenia:
  - *0* – niedziela, *1* – poniedziałek, . . . , *6* – sobota
- Na ekranie wypisz datę i nazwę wyliczonego dnia tygodnia
- Do wypisania nazwy dnia tygodnia – podobnie jak w poprzednim ćwiczeniu – użyj instrukcji warunkowej
- Przetestuj działanie programu dla kilku wybranych dat

3. ZŁOŻONE TYPY DANYCH

ĆWICZENIE 3.1:

Skrót

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - wykorzystania pętli
  - korzystania z operacji na tekstach

CELE I ZADANIA:

- Napisz program, który w oparciu o podaną pełną nazwę utworzy jej skrót, np. zamieni *United Nations Educational, Scientific and Cultural Organization* na *UNESCO*

ALGORYTM WYKONANIA:

- Z podanej pełnej nazwy wyodrębnij słowa – wyszukaj odpowiednią funkcję
- Z każdego słowa weź pierwszy znak i upewnij się, że jest dużą literą
  - uwaga: jeśli zmienna *s* reprezentuje tekst, to pierwszy znak tego tekstu można otrzymać za pomocą konstrukcji *s[0]*
- Połącz ze sobą ciąg tych liter tworząc skrót
- Wypisz na ekranie oryginalną pełną nazwę i utworzony skrót

ĆWICZENIE 3.2:

Wieczny kalendarz

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych i operatorów
  - stosowania krotek

CELE I ZADANIA:

- Napisz program, który pełni rolę wiecznego kalendarza – na podstawie podanej daty (dnia, miesiąca i roku) wyznacza nazwę dnia tygodnia
- Wykorzystaj algorytm opracowany przez Mike'a Keith'a
- Elementy daty wczytaj z klawiatury

ALGORYTM WYKONANIA:

- Zmodyfikuj rozwiązanie ćwiczenia 2.6
- Utwórz krotkę z nazwami dni tygodnia
- Uprość kod usuwając instrukcję warunkową
- Wykorzystaj krotkę do powiązania indeksu krotki z numerem dnia tygodnia wyliczonym przez algorytm
- Zastanów się, jak będzie najprościej – który dzień tygodnia powinien być pierwszym elementem krotki?
- Na ekranie wypisz datę i nazwę wyliczonego dnia tygodnia
- Przetestuj działanie programu dla kilku wybranych dat

ĆWICZENIE 3.3:

Liczba "pechowych" dni

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych i operatorów
  - stosowania pętli

CELE I ZADANIA:

- Napisz program, który dla podanego roku wyliczy ile w nim jest piątków 13-tego
- Wykorzystaj algorytm z poprzedniego ćwiczenia
- Testowany rok wczytaj z klawiatury

ALGORYTM WYKONANIA:

- Napisz program, który dla podanego roku sprawdzi ile w nim jest piątków 13-tego
- Wczytaj z klawiatury: rok
- Utwórz licznik i wstępnie go zainicjalizuj (jaką wartością?)
- W pętli sprawdź, jakie dni tygodnia wypadały 13-tego dnia kolejnego miesiąca – jeśli był to piątek, to zinkrementuj licznik
- Do sprawdzenia dnia tygodnia wykorzystaj algorytm zaimplementowany w poprzednim ćwiczeniu
- Na koniec przedstaw wynik na ekranie

ĆWICZENIE 3.4:

Losowanie lotto

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - wykorzystania pętli
  - tworzenia zakresów

CELE I ZADANIA:

- Napisz program, który wyliczy szansę wylosowania k liczb spośród n różnych liczb (jak w lotto)
- W rozwiązaniu nie posługuj się zewnętrznymi modułami
- Wykorzystaj pętle i zakresy

ALGORYTM WYKONANIA:

- Podobnie jak w ćwiczeniu 2.4 wczytaj z klawiatury dane wejściowe:
  - k – ilość skreślanych liczb (np. 6)
  - n – całkowitą ilość liczb (np. 49)
- Liczbę możliwych kombinacji wyboru k spośród n różnych liczb opisuje wzór:

$$\frac{n!}{k!(n-k)!}$$

  gdzie: *m!* oznacza silnię liczby m
- Zauważ, że:

$$k! = 1 * 2 * \ldots * k$$

  zaś *n!* dzieli się bez reszty przez *(n-k)!* i da się wyliczyć jako iloczyn liczb

$$\frac{n!}{(n-k)!} = (n-k+1) * (n-k+2) * \ldots * n$$

- Powyższe iloczyny oblicz wykorzystując pętle i zakresy
- Podziel otrzymane wartości przez siebie i sprawdź, czy otrzymałeś identyczny wynik jak w ćwiczeniu 2.4

ĆWICZENIE 3.5:

Robot

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych i ich funkcji
  - stosowania pętli

CELE I ZADANIA:

- Napisz program, który będzie symulował ruch robota w 4 kierunkach (N/E/S/W)
- Program powinien działać w pętli i kontrolować położenie robota
- Po zakończeniu pracy program powinien obliczyć odległość robota w linii prostej od punktu startowego

ALGORYTM WYKONANIA:

- Napisz program, który w nieskończonej pętli będzie przyjmował z klawiatury polecenia ruchu dla robota
- Polecenia powinny mieć postać:

$$< kierunek > < ile\_krokow >$$

  np.:

  - *N 4* oznacza 4 kroki na północ
  - *W 1* oznacza 1 krok na zachód
  - *S 2* oznacza 2 kroki na południe
  - *E 3* oznacza 3 kroki na wschód, itd.
- Po każdym poleceniu program powinien wypisywać aktualne współrzędne położenia robota
- Wciśnięcie klawisza <Enter> powinno być sygnałem do zakończenia programu (i opuszczenia nieskończonej pętli)
- Na koniec program powinien obliczyć w jakiej odległości od punktu startowego znalazł się robot (możesz wykorzystać odpowiednią funkcję z biblioteki matematycznej)

ĆWICZENIE 3.6:

Skrót

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z operacji na tekstach
  - budowy wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który w oparciu o podaną pełną nazwę utworzy jej skrót, np. zamieni *United Nations Educational, Scientific and Cultural Organization* na *UNESCO*

ALGORYTM WYKONANIA:

- Zmodyfikuj rozwiązanie ćwiczenia 3.1
- Wykorzystując ten sam algorytm uprość rozwiązanie stosując "podejście pythonowe"
- Usuń ze skryptu pętlę i zastąp ją wyrażeniem listowym
- Sprawdź poprawność rozwiązania
- Wypisz na ekranie oryginalną pełną nazwę i utworzony skrót

ĆWICZENIE 3.7:

Palindrom

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - wykorzystania wycinków sekwencji
  - korzystania z operacji na tekstach

CELE I ZADANIA:

- Napisz program, który zweryfikuje, czy podany tekst jest palindromem
- Palindrom brzmi tak samo, niezależnie od tego, czy jest czytany od przodu, czy wspak
- Wielkość liter, znaki separatorów i znaki interpunkcyjne nie mają znaczenia

ALGORYTM WYKONANIA:

- Zapisz w zmiennej treść tekstu z potencjalnym palindromem
- Palindrom brzmi tak samo, niezależnie od tego, czy jest czytany od przodu, czy wspak
- Do testów możesz użyć następującego tekstu: *Co mi dał duch? Cud, ład i moc.*
- Na podstawie powyższego tekstu utwórz listę zawierającą jedynie litery (wszystkie inne znaki usuwamy) – użyj do tego wyrażenia listowego
- W liście należy ujednolicić wielkość liter – zamień wszystkie litery na litery małe (lub na duże)
- Sprawdź, czy ta lista i lista o odwrotnej kolejności liter są identyczne
- Możesz "ulepszyć" rozwiązanie testując, czy pierwsza połowa listy jest identyczna z drugą połową napisaną wspak
- Wypisz na ekranie wynik porównania

ĆWICZENIE 3.8:

Początki kwartałów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - użycia list
  - wykorzystania pętli
  - wykorzystania wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który z podanej krotki nazw miesięcy wybierze te, które są początkami kwartałów
- Wykorzystaj wyrażenie listowe

ALGORYTM WYKONANIA:

- Utwórz krotkę zawierającą nazwy miesięcy
- Napisz wyrażenie listowe, które zwróci nową listę z nazwami pierwszych miesięcy w każdym kwartale
- Wypisz zawartość tej listy na ekranie
- Zastanów się, jak wypisać elementy listy, separując je przecinkami, bez użycia pętli
- Jak inaczej można rozwiązać to zadanie bez użycia wyrażeń listowych?

ĆWICZENIE 3.9:

Kwartały

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - użycia list
  - wykorzystania pętli
  - wykorzystania wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który z podanej krotki nazw miesięcy utworzy nową listę, w której miesiące będą pogrupowane w kwartały
- Wykorzystaj wyrażenie listowe

ALGORYTM WYKONANIA:

- Utwórz krotkę zawierającą nazwy miesięcy
- Napisz wyrażenie listowe, które zwróci nową listę, zawierającą 4 listy z nazwami miesięcy kolejnych kwartałów (macierz 4x3)
- Wypisz zawartość tej listy na ekranie

- Zwróć uwagę, że jeśli skrypt napiszesz uniwersalnie nie kodując w nim informacji o tym, że mamy 4 kwartały, a każdy z nich zawiera 3 miesiące, to można go będzie wykorzystać do prezentacji dowolnych danych w formie tabeli
- Opcjonalnie, spróbuj ulepszyć w ten sposób swój skrypt i wykorzystaj go do pogrupowania miesięcy w półrocza (sprawdź także, czy działa z kwartałami)

ĆWICZENIE 3.10:

Liczby rzymskie

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania i wykorzystania słowników
  - użycia instrukcji sterujących

CELE I ZADANIA:

- Napisz program, który podaną liczbę zapisze za pomocą cyfr rzymskich

ALGORYTM WYKONANIA:

- Zdefiniuj słownik, którego kluczami będą liczby arabskie, a wartościami ich odpowiedniki rzymskie (zastanów się, które?)
- Dla przypomnienia: I - 1, V - 5, X - 10, L - 50, C - 100, D - 500, M - 1000
- Klucze powinny być uporządkowane malejąco
- Wczytaj z klawiatury liczbę całkowitą z przedziału między 1 a 3999
- Jeśli wczytana liczba nie mieści się w tym zakresie, to program powinien o tym poinformować
- Korzystając z pętli pomniejszaj liczbę o kolejne wartości zdefiniowane jako klucze słownika (od największych do coraz mniejszych) mieszczące się w liczbie i zapisuj ich odpowiedniki rzymskie
- Uwaga: Algorytm postępowania przypomina problem wydania reszty z użyciem jak najmniejszej ilości nominałów

ĆWICZENIE 3.11:

"Baza" osób

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - symulowania kolekcji danych za pomocą słowników oraz list (namiastka bazy danych)
  - wydobywania potrzebnych danych z utworzonej struktury

CELE I ZADANIA:

- Za pomocą słowników utwórz dane opisujące kilka osób
- Utwórz z nich listę
- Napisz kilka konstrukcji za pomocą których utworzysz podzbiór osób spełniających zadane kryteria
- Zaproponuj kilka przykładów w których użyjesz funkcji agregujących dane (np. suma, średnia arytmetyczna, minimum, maksimum, liczba elementów)

ALGORYTM WYKONANIA:

- Zaprojektuj słownik opisujący osobę
- W tym celu zaproponuj atrybuty charakteryzujące osobę – w słowniku będą one pełniły rolę kluczy (np. imię, nazwisko, adres, płeć, wiek, itp.)
- Zastanów się, jakie typy danych będą powiązane z tymi kluczami (najlepiej, gdyby były różnorodne)
- Za pomocą zaprojektowanych słowników opisz kilka osób
- Zgrupuj te osoby, tworząc na ich podstawie listę
- W ten sposób utworzona została struktura danych przypominająca kolekcję obiektów lub trywialną bazę danych
- W oparciu o tak utworzone dane wejściowe zrealizuj kilka zapytań, np.:
  - utwórz listę mężczyzn mieszkających w podanym mieście
  - utwórz listę osób z danego przedziału wiekowego, np. 20-latków
  - oblicz średnią wieku kobiet o imionach rozpoczynających się literą A
  - itp.
- Spróbuj wymyśleć kilka przykładów podobnych zapytań i je zrealizować
- Uwaga: w rozwiązaniach mogą być przydatne wyrażenia listowe

4. PROGRAMOWANIE FUNKCYJNE

ĆWICZENIE 4.1:

Wieczny kalendarz – użycie funkcji

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - wykonywania operacji na tekstach
  - definiowania i korzystania z funkcji

CELE I ZADANIA:

- Napisz funkcję, która na podstawie daty podanej jako tekst, określi jaki to dzień tygodnia
- Wykorzystaj algorytm z ćwiczenia 3.2
- W rozwiązaniu zdefiniuj dwie funkcje pomocnicze (jedna – określająca numer dnia tygodnia, a druga – na podstawie numeru określająca nazwę dnia tygodnia)

ALGORYTM WYKONANIA:

- Korzystając z algorytmu zaimplementowanego w ćwiczeniu 3.2 utwórz funkcję pomocniczą *numer_dnia_tygodnia*
  - funkcja powinna posiadać dwa argumenty: tekst zawierający datę (w kolejności: dzień, miesiąc, rok) oraz użyty separator do oddzielenia elementów daty
  - zgodnie z zaimplementowanym algorytmem funkcja powinna zwrócić numer dnia tygodnia
- Zastanów się, jak efektywnie wyodrębnić z daty: dzień, miesiąc i rok i skonwertować je na wartości liczbowe
- Utwórz drugą funkcję pomocniczą *nazwa_dnia_tygodnia*
  - funkcja powinna poprzez argument przyjąć numer dnia tygodnia
  - w oparciu o dane na wejściu powinna zwrócić tekst z nazwą dnia tygodnia
- Utwórz funkcję *jaki_to_dzien*, która na podstawie tekstu z datą oraz użytego separatora wypisze nazwę dnia tygodnia
- Podanie parametru separatora powinno być opcjonalne
- W implementacji wykorzystaj utworzone wcześniej funkcje pomocnicze
- Wywołaj utworzoną funkcję dla wczytanej wcześniej daty i sprawdź jej działanie

ĆWICZENIE 4.2:

"Baza" osób – własne funkcje

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - symulowania kolekcji danych za pomocą słowników oraz list (namiastka bazy danych)
  - wydobywania potrzebnych danych z utworzonej struktury
  - definiowania i użycia funkcji

CELE I ZADANIA:

- Wzorując się na rozwiązaniu ćwiczenia 3.11 zrealizuj te same zapytania, tym razem wykorzystując wbudowane funkcje *filter* oraz *map*
- W rozwiązaniu wykorzystaj własne funkcje reprezentujące warunki (predykaty) przekazywane jako argument do funkcji *filter* oraz funkcję przekształcenia przekazywaną jako argument do funkcji *map*
- Sprawdź, czy wyniki są identyczne

ALGORYTM WYKONANIA:

- Z ćwiczenia 3.11 skopiuj dane wejściowe (słowniki i listę)
- Możesz też utworzyć dodatkową funkcję *utworz_osobe*, która pobierze wszystkie wymagane dane opisujące osobę i zwróci gotowy słownik – w ten sposób zmniejszymy ryzyko pomyłki podczas tworzenia słowników, gdyż funkcja zadba, aby we wszystkich słownikach były te same klucze
- Utwórz własne funkcje:
  - predykaty, które reprezentują warunki wykorzystywane w zapytaniach (występujące w wyrażeniach listowych)
  - funkcje, które przekształcają dane sekwencji (patrz: ostatnie zapytanie)
- Wykorzystaj utworzone funkcje oraz wbudowane funkcje *filter* oraz *map* do zrealizowania tych samych zapytań co w ćwiczeniu 3.11 – w ten sposób uda się wyeliminować wyrażenia listowe
- Porównaj otrzymane wyniki ze wcześniejszymi

ĆWICZENIE 4.3:

"Baza" osób – sortowanie

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - symulowania kolekcji danych za pomocą słowników oraz list (namiastka bazy danych)
  - sortowania danych wg zadanych kryteriów

CELE I ZADANIA:

- Korzystając z danych z poprzedniego ćwiczenia 4.2 wypisz na ekranie aktualne dane wszystkich osób znajdujących się w liście
- Uporządkuj dane wg różnych, zadanych kryteriów i wypisz je na ekranie, aby sprawdzić, czy zostały posortowane

ALGORYTM WYKONANIA:

- Skopiuj rozwiązanie poprzedniego ćwiczenia 4.2
- Pozostaw tylko dane (słowniki i listę)
- Wypisz na ekranie aktualny stan listy
- Posortuj dane wg zadanych kryteriów, np.:
  - wg nazwisk
  - wg płci i wieku
  - ...
- Wypisz ponownie te dane, aby sprawdzić, czy zostały one faktycznie posortowane

ĆWICZENIE 4.4:

"Baza" osób – funkcje lambda

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - symulowania kolekcji danych za pomocą słowników oraz list (namiastka bazy danych)
  - wydobywania potrzebnych danych z utworzonej struktury
  - definiowania i użycia wyrażeń lambda

CELE I ZADANIA:

- Wzorując się na rozwiązaniu ćwiczenia 4.2 zastąp funkcje przekazywane jako argumenty do funkcji *filter* i *map* funkcjami lambda

ALGORYTM WYKONANIA:

- Skopiuj rozwiązanie ćwiczenia 4.2
- Usuń funkcje predykatów oraz funkcje konwersji
- Zastąp je funkcjami lambda
- Sprawdź działanie tak zmodyfikowanego programu
- Porównaj otrzymane wyniki ze wcześniejszymi
- Jakie zalety/wady ma to rozwiązanie?

ĆWICZENIE 4.5:

Wieczny kalendarz – typowanie statyczne

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - wykonywania operacji na tekstach
  - definiowania i korzystania z funkcji
  - użycia typowania statycznego

CELE I ZADANIA:

- Zmodyfikuj rozwiązanie ćwiczenia 4.1, zastępując typowanie dynamiczne typowaniem statycznym

ALGORYTM WYKONANIA:

- Skopiuj rozwiązanie ćwiczenia 4.1
- Uzupełnij kod, tam gdzie to możliwe, o typowanie statyczne
- Określ typy danych dla parametrów funkcji, zwracanych wartości oraz użytych zmiennych
- Zobacz, jak zareaguje IDE, jeśli wbrew deklaracji użyjesz danych niezgodnych typów

5. KLASY I OBIEKTY

ĆWICZENIE 5.1:

Definiowanie klas i obiektów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - definiowania klas i atrybutów instancyjnych
  - tworzenia instancji na podstawie klas
  - dostępu do atrybutów instancyjnych

CELE I ZADANIA:

- Utwórz prostą klasę reprezentującą osoby
- Zaproponuj atrybuty, jakie powinna posiadać ta klasa
- Zdefiniuj sposób tworzenia obiektów i zachowanie klas

ALGORYTM WYKONANIA:

- Utwórz klasę o nazwie *Osoba*
  - każda osoba jest opisywana przez atrybuty reprezentujące: imię (tekst), nazwisko (tekst), płeć (wartość logiczna: True – mężczyzna, False – kobieta) oraz rok urodzenia (liczba)
  - możesz też zaproponować inne atrybuty...
  - podczas tworzenia instancji wszystkie dane muszą być podane
  - zdefiniuj metodę o nazwie *ile_lat*, która obliczy aktualny wiek osoby
    - aby odczytać aktualny rok zaimportuj moduł *datetime*
    - bieżący rok zwróci wyrażenie: *datetime.date.today().year*
  - zdefiniuj metodę (o nazwie *__str__*) zwracającą opis osoby, np.: *Jan Kowalski, płeć: M, wiek: 29 lat*
- Przetestuj działanie klasy
  - utwórz osobę, podając wszystkie niezbędne dane
  - wypisz wybrane atrybuty (np. imię, a następnie nazwisko)
  - wypisz wszystkie dane osoby, podając funkcji *print* zmienną reprezentującą utworzoną instancję osoby
  - zwiększ o 1 rok urodzenia utworzonej osoby i ponownie wypisz wszystkie informacje o niej
- Czy taka konstrukcja klas chroni nas przed wprowadzeniem bezsensownych wartości (np. roku urodzenia pochodzącego z przyszłości)?
- Zobacz, jak wtedy zachowa się program

ĆWICZENIE 5.2:

Hermetyzacja

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - hermetyzowania klas

CELE I ZADANIA:

- Dokonaj hermetyzacji klas z poprzedniego ćwiczenia 5.1

ALGORYTM WYKONANIA:

- Zmodyfikuj klasę *Osoba* z poprzedniego ćwiczenia 5.1
- Dokonaj jej hermetyzacji
- W tym celu atrybuty zadeklaruj jako silnie prywatne
- Modyfikacja wartości atrybutów będzie możliwa za pomocą dodatkowych metod dostępowych (np. *ustaw_imie*, *podaj_imię*, itd.)
- Zagwarantuj, że przyjmowane będą tylko wartości sensowne, tzn. imię i nazwisko nie mogą być puste, zaś data urodzenia musi być bieżąca lub przeszła
- Jak zmieni się kod testujący?
- Jakie widzisz tu utrudnienia?

ĆWICZENIE 5.3:

Wykorzystanie właściwości

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - tworzenia i wykorzystania właściwości

CELE I ZADANIA:

- Korzystając z rozwiązania poprzedniego ćwiczenia 5.2, zrezygnuj z hermetyzacji klasy
- Atrybuty klasy zdefiniuj jako właściwości

ALGORYTM WYKONANIA:

- Zmodyfikuj klasę (*Osoba*) z poprzedniego ćwiczenia 5.2
- Zrezygnuj z hermetyzacji klasy
- Atrybuty zadeklaruj jako właściwości – to spowoduje, że dostęp do nich, choć wygląda jak bezpośredni, faktycznie będzie się odbywał poprzez metody dostępowe
- Przetestuj działanie programu
- Upewnij się, że inaczej niż to miało miejsce w pierwszym ćwiczeniu, teraz można kontrolować (walidować) wartości atrybutów

ĆWICZENIE 5.4:

Dziedziczenie – pracownicy i kierownicy zespołów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania klas
  - wykorzystania relacji dziedziczenia

CELE I ZADANIA:

- Utwórz klasę opisującą pracownika, zaproponuj atrybuty i zainicjuj je
- Utwórz klasę kierownika zespołu wykorzystując relację dziedziczenia (kierownik jest też pracownikiem)
- Zmień zachowanie wybranych metod, wykorzystując zachowanie oryginalne

ALGORYTM WYKONANIA:

- Utwórz klasę o nazwie *Pracownik*, reprezentującą pracownika
- Dla każdego pracownika jesteśmy w stanie określić jego imię, nazwisko i zarobki
- Utwórz i zainicjuj te atrybuty w metodzie *__init__*
- Zastanów się, którą metodę należy przedefiniować, aby dostarczyć własnego opisu obiektu (przeznaczonego dla człowieka)
- Zmień tę reprezentację tak, aby po wykonaniu instrukcji:
```
p1 = Pracownik('Jan', 'Kowalski', 4_000)
print(p1)
```
otrzymać komunikat:
```
[Pracownik] Jan Kowalski, zarobki: 4000
```

- Utwórz klasę o nazwie *KierownikZespolu*
- Klasa powinna dziedziczyć po klasie *Pracownik* (gdyż każdy kierownik, też jest pracownikiem, ale na odwrót już nie)
- Kierownik powinien posiadać te same atrybuty, co pracownik oraz dodatkowo:
  - atrybut reprezentujący listę swoich pracowników (początkowo pustą)
  - atrybut reprezentujący odpowiedzialność (inicjowany podczas tworzenia obiektu)
- Inicjalizację atrybutów wspólnych dla obu klas deleguj do klasy pracownika
- Podczas tworzenia instancji kierownika, jego zespół powinien być pusty (nie ma potrzeby podawania zespołu jako argumentu podczas tworzenia instancji kierownika, ale atrybut powinien powstać)
- Zdefiniuj w klasie kierownika metody umożliwiające poszerzenie lub zmniejszenie zespołu (np. metody: *dodaj_pracownika* i *usun_pracownika*)

- Dodatkowo dodaj metodę *kto_w_zespole*, która wypisze listę pracowników należących do zespołu danego kierownika
- Wreszcie przedefiniuj metodę zwracającą opis kierownika – powinien on być zbieżny z opisem pracownika (również wykorzystaj delegację) i poszerzony o informację o odpowiedzialności (bez informacji o zespole)
- Sprawdź działanie aplikacji

ĆWICZENIE 5.5:

Przeciążanie operatorów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania klas
  - przeciążania operatorów

CELE I ZADANIA:

- Utwórz klasę i przeciąż w niej wybrane operatory

ALGORYTM WYKONANIA:

- Utwórz klasę o nazwie *Osoba* – możesz ją skopiować z ćwiczenia 5.1
- Utwórz kilka instancji tej klasy
- Sprawdź zachowanie programu, gdy spróbujesz porównać instancje tych osób za pomocą operatorów: <, ==, >
- Zastanów się, które metody należy przedefiniować, aby program działał poprawnie
- Zmień zachowanie tych metod
- Operatory trzeba przeciążyć tak, aby ustalały porządek dwóch osób – o kolejności powinno najpierw decydować nazwisko, a jeśli nie będzie to rozstrzygające, to imię
- Operator porównania powinien zwrócić wartość *True*, gdy dwie osoby mają to samo imię i nazwisko (mimo tego, że mogą się różnić innymi atrybutami)
- Przykładowo:

```
Jan Kowalski <  Adam Nowak
Jan Kowalski >  Dariusz Kowalski
Jan Kowalski >  Jan Jabłoński
Jan Kowalski == Jan Kowalski
```

- Przetestuj działanie zdefiniowanych operatorów

6. MODUŁY I PAKIETY

ĆWICZENIE 6.1:

Użycie modułów i pakietów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - tworzenia pakietów i modułów
  - wykorzystania modułów

CELE I ZADANIA:

- Utwórz klasę reprezentującą adres (ulica, kod pocztowy, miejscowość) i zapisz ją w module o tej samej nazwie
- Utwórz klasę reprezentującą osobę (imię, nazwisko, adres) i zapisz ją w module o tej samej nazwie
- W obu klasach zdefiniuj metody konwersji obiektów na tekst
- Z poziomu osobnego modułu przetestuj działanie obu klas

ALGORYTM WYKONANIA:

- Utwórz w module *adres.py* klasę *Adres*
  - utwórz w niej atrybuty instancyjne reprezentujące: ulicę, kod pocztowy i miejscowość
  - zdefiniuj metody umożliwiające reprezentację tekstową instancji
- Utwórz w module *osoba.py* klasę *Osoba*
  - utwórz w niej atrybuty instancyjne reprezentujące: imię, nazwisko i adres
  - zdefiniuj metody umożliwiające reprezentację tekstową obiektów
- Oba te moduły umieść w pakiecie *model*
- W module *osoby_test.py* utwórz listę kilku osób
- Wykorzystaj instrukcje importu w różnych wariantach
- Przedstaw dane tych osób, wykorzystując ich reprezentacje tekstowe

7. OPERACJE NA PLIKACH

ĆWICZENIE 7.1:

Metamorfoza – odczyt zawartości plików

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - prawidłowej pracy z plikami tekstowymi
  - odczytu danych z plików

CELE I ZADANIA:

- Wykorzystaj podane 2 pliki zawierające tekst piosenki w dwóch wersjach językowych (ang. i pl.)
- Przeczytaj i wypisz zawartość obu plików na dwa sposoby:
  - jeden plik za drugim
  - linie na przemian z jednego i drugiego pliku

ALGORYTM WYKONANIA:

- Umieść w bieżącym pakiecie dostarczone przez instruktora pliki z treścią piosenki
- Napisz program, który umożliwi przeczytanie zawartości obu plików (jeden za drugim)
- W rozwiązaniu możesz wykorzystać fakt, że obiekty plików (strumienie) są iterowalne
- Zwróć uwagę, czy znaki narodowe są poprawnie przedstawione – w plikach zostało użyte kodowanie UTF-8
- Pamiętaj, aby po zakończeniu pracy z plikami zamknąć otwarte strumienie (obiekty plików są menedżerami kontekstu)

- Utwórz drugi program (wzorując się na poprzednim) i wyświetl zawartość obu plików, prezentując linie z obu plików na przemian, tzn. pierwsza linia z pliku 1, pierwsza linia z pliku 2 (czyli jej tłumaczenie), druga linia z pliku 1, druga linia z pliku 2, itd.
- Zobacz, jak teraz prezentuje się treść...

ĆWICZENIE 7.2:

Łączenie plików

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - odczytu danych z plików
  - zapisu danych do pliku

CELE I ZADANIA:

- Skopiuj zawartość 2 plików i zapisz ją w nowym pliku

ALGORYTM WYKONANIA:

- Wykorzystaj przykładowe dane (2 wersje językowe piosenki) z poprzedniego ćwiczenia 7.1
- Przeczytaj zawartość z obu plików i zapisz ją w nowym pliku
- Ponieważ pliki są niewielkie, możesz spróbować przeczytać je jednym poleceniem
- Zadbaj o zamknięcie wszystkich strumieni

8. WYJĄTKI

ĆWICZENIE 8.1:

Osoby – użycie wyjątków standardowych

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - użycia wyjątków standardowych
  - obsługi wyjątków

CELE I ZADANIA:

- Wykorzystaj rozwiązanie ćwiczenia 5.3
- Użyj wyjątków standardowych do sygnalizacji sytuacji wyjątkowych
- Dodaj obsługę wyjątków

ALGORYTM WYKONANIA:

- Wykorzystaj rozwiązanie ćwiczenia 5.3
- Użyj mechanizmu wyjątków do zasygnalizowania wystąpienia sytuacji wyjątkowej
- W tym przypadku sytuacją wyjątkową będzie próba ustawienia niedozwolonej wartości atrybutu (np. pustego tekstu imienia lub nazwiska, czy też podanie roku urodzenia z przyszłości)
- Zastanów się, która standardowa klasa wyjątku może być użyta do sygnalizacji powyższych sytuacji
- Wykryj taką sytuację i wyrzuć obiekt wyjątku, aby nie dopuścić do przyjęcia błędnych danych
- Uruchom program i zobacz, co się stanie, gdy spróbujesz podać błędne lub niekompletne dane
- Dodaj obsługę wyjątku
- W jaki sposób można związać obiekt wyjątku z komunikatem?
- Jak można dotrzeć do tego komunikatu?
- Uruchom ponownie program i sprawdź jego działanie
- Co się stanie, jeśli podamy wartość niewłaściwego typu (np. rok urodzenia podamy słownie)?
- Dodaj obsługę nowego wyjątku i uruchom ponownie program

ĆWICZENIE 8.2:

Osoby – własne klasy wyjątków

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania własnych klas wyjątków
  - wyrzucania wyjątków i ich obsługi

CELE I ZADANIA:

- Wykorzystaj rozwiązanie poprzedniego ćwiczenia 8.1
- Zastąp standardowe klasy sygnalizujące błędne wartości – własnymi klasami

ALGORYTM WYKONANIA:

- Zmodyfikuj rozwiązanie poprzedniego ćwiczenia 8.1
- Utwórz dedykowane klasy wyjątków o nazwach *NoValueError* oraz *BirthDateError*
- Klasy powinny mieć wbudowany komunikat błędu, informujący o niewłaściwej wartości
- Nazwę błędnego atrybutu (w przypadku klasy *NoValueError*) należy podać podczas tworzenia instancji wyjątku
- Zastąp standardowe klasy wyjątków własnymi i sprawdź działanie programu

ĆWICZENIE 8.3:

Telefon na kartę – sytuacje wyjątkowe

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania i wykorzystania w sposób praktyczny własnych klas wyjątków

CELE I ZADANIA:

- Napisz program symulujący działanie telefonu na kartę
- Wykorzystaj własną klasę wyjątku do sygnalizacji sytuacji niedozwolonych

ALGORYTM WYKONANIA:

- Napisz program symulujący działanie telefonu na kartę
- W tym celu utwórz klasę o nazwie *PrepaidPhone*
- Klasa powinna przechować informację o ilości dostępnych minut na rozmowy
- Ta wartość powinna zostać zainicjowana w czasie tworzenia obiektu (starter)
- Dodaj metody umożliwiające:
  - *get_limit* – sprawdzenie aktualnego stanu minut
  - *add_to_limit* – doładowanie konta zadaną liczbą minut
- Dodaj metodę *call* symulującą rozmowę trwającą przez zadany czas
- W wyniku wywołania tej metody stan konta powinien zostać pomniejszony
- Przetestuj działanie programu wywołując kilkukrotnie metodę rozmowy i metodę doładowania
- Co się stanie, gdy czasy trwania rozmów będą przewyższały wartości doładowań i limit minut się wyczerpie?

- Utwórz klasę wyjątku *PrepaidPhoneError* sygnalizującą taką sytuację
- W metodzie rozmowy wykryj sytuację wyczerpania limitu minut i zgłoś ten fakt, poprzez wyrzucenie obiektu wyjątku
- Wykryj w programie wystąpienie wyjątku, a w jego obsłudze doładuj konto i wypisz aktualny stan minut
- Przetestuj działanie programu

9. WAŻNE WBUDOWANE MODUŁY I

BIBLIOTEKI

ĆWICZENIE 9.1:

Wieczny kalendarz

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - pracy z dokumentacją
  - zapoznania się z pakietem *datetime*

CELE I ZADANIA:

- W oparciu o dokumentację modułu *datetime* napisz program, który dla podanej daty zwróci nazwę dnia tygodnia

ALGORYTM WYKONANIA:

- Napisz program o podobnej funkcjonalności do ćwiczenia 3.2
- Zapoznaj się z pakietem *datetime* i klasą *date*
- Wyszukaj metody, które dla danej daty zwracają dzień tygodnia
- Czym one się różnią?
- Uruchom program i porównaj wyniki z wcześniejszym rozwiązaniem

ĆWICZENIE 9.2:

Kwartały

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - pracy z dokumentacją
  - zapoznania się z funkcjami specjalnymi biblioteki *numpy*

CELE I ZADANIA:

- W oparciu o dokumentację funkcji specjalnych biblioteki *numpy* napisz program, który wyliczy przekształci listę zawierającą nazwy miesięcy na nową listę zawierającą kwartały
- Każdy kwartał powinien być listą zawierającą 3 miesiące

ALGORYTM WYKONANIA:

- Napisz program o podobnej funkcjonalności do programu z ćwiczenia 3.9
- Wykorzystaj możliwości funkcji biblioteki *numpy*
- Zapoznaj się z podstawowymi możliwościami pakietu:
  `https://docs.scipy.org/doc/numpy/user/quickstart.html`
- W programie dodaj na początku instrukcję importu:
  import numpy
- Utwórz krotkę z nazwami miesięcy
- Przekształć ją na tablicę
- Przeorganizuj tablicę jednowymiarową (1 x 12) na dwuwymiarową (4 x 3)
- Dokonaj konwersji na listę
- Porównaj otrzymane wyniki z tymi z wcześniejszego ćwiczenia