ALTKOM

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

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - samodzielnego przygotowania środowiska do nauki i tworzenia aplikacji w Pythonie (konsolę interaktywną oraz IDE)
  - 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 *Easter Egg*
- Zainstaluj środowisko IDE

ALGORYTM WYKONANIA:

- Ze strony `https://www.python.org/downloads/` pobierz instalator Pythona
- Uruchom program instalacyjny i wskaż katalog, gdzie ma być zainstalowany Python
- Upewnij się, czy i jakie katalogi Pythona zostały dodane do zmiennej systemowej *PATH*
- 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
- W jaki sposób można opuścić ten tryb?
- Pobierz i zainstaluj wybrane środowisko IDE, np. *PyCharm*
- Sprawdź, czy z poziomu IDE działa konsola Pythona

2 Składnia języka

ĆWICZENIE 2.1:

Praca z dokumentacją

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- Uruchom dokumentację Pythona
- Zapoznaj się z jej możliwościami
- 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

3 Numeryczne typy danych

ĆWICZENIE 3.1:

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 po upływie zadanej ilości lat
- Zastanów się, jakie dane wejściowe będą potrzebne?
- Jakich typów będą te dane?
- Wprowadź niezbędne dane z klawiatury
- Napisz wyrażenie wyliczające stan lokaty
- Zastanów się, jak wynik końcowy zaokrąglić do dwóch miejsc w części ułamkowej
- Wypisz wynik na ekranie

ĆWICZENIE 3.2:

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)

- Zapisz powyższą regułę w postaci pojedynczego wyrażenia logicznego
- Wykonaj 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 3.3:

Losowanie lotto

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - pracy z dokumentacją
  - wyszukiwania potrzebnych informacji
  - korzystania z zewnętrznych 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 – 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
- Odszukaj dokumentację modułu matematycznego, a następnie funkcję wyliczającą silnię
- Podobnie, jak to robiłeś w ćwiczeniu 1.2, 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

4 Struktury sterujące

ĆWICZENIE 4.1:

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 ułamkowek
- Dodatkowo podaj przedział prawidłowej wagi dla określonego wzrostu

ĆWICZENIE 4.2:

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 x
  - *mod(...)* oznacza funkcję 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* – poniedzialek, ..., *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

5 Typ tekstowy

ĆWICZENIE 5.1:

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 5.2:

Robot

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - korzystania z wybranych typów danych 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 pętli z klawiatury będzie przyjmował 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 2 kroki na wschód, itd.

- Po każdym poleceniu program powinien wypisywać aktualne współrzędne położenia robota
- Wciśnięcie klawisza <Enter> powinno być sygnałem do zakończenia programu
- Na koniec program powinien obliczyć w jakiej odległości od punktu startowego znalazł się robot

6 Krotki, zakresy, listy

ĆWICZENIE 6.1:

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 4.2
- 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 6.2:

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 6.3:

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 3.3 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 3.3

7 Sekwencje

ĆWICZENIE 7.1:

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 5.1
- 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 7.2:

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 np. *Co mi dał duch? Cud, ład i moc.*
- Z powyższego utwórz nowy tekst zawierający jedynie litery (wszystkie inne znaki usuwamy)
- W tekście należy ujednolicić wielkość liter – zamień wszystkie litery w tekście na litery małe (lub na duże)
- Sprawdź, czy ten tekst i tekst o odwrotnej kolejności liter są identyczne
- uwaga: do utworzenia tekstu o odwrotnej kolejności znaków możesz użyć wycinków sekwencji

- Możesz "ulepszyć" rozwiązanie testując, czy pierwsza połowa tekstu jest identyczna z drugą połową napisaną wspak
- Wypisz na ekranie wynik porównania

ĆWICZENIE 7.3:

Początki kwartałów

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

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

ALGORYTM WYKONANIA:

- Utwórz listę 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
- Utwórz wyrażenie listowe, które utworzy macierz 4x3 z nazwami miesięcy pogrupowanych w kwartały

ĆWICZENIE 7.4:

Kwartały

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

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

ALGORYTM WYKONANIA:

- Utwórz listę 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 7.5:

Macierze

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- Napisz program, który utworzy macierz jednostkową
- Wykorzystaj wyrażenie listowe

ALGORYTM WYKONANIA:

- Macierz jednostkowa 4 x 4 ma postać:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Napisz wyrażenie listowe, które utworzy powyższą macierz (użyj zagnieżdżonych list)
- Powinna ona wyglądać następująco:

$$[[1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]$$

- Spróbuj napisać program uniwersalnie, tak, aby wymiar macierzy mógł być zadany jako parametr lub wczytany z klawiatury
- Sprawdź działanie programu
- Wypisz jej zawartość na ekranie

9 Słowniki

ĆWICZENIE 9.1:

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?)
- 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 monet

ĆWICZENIE 9.2:

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

10 Funkcje

ĆWICZENIE 10.1:

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 6.1
- 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 6.1 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 przyjąc 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
- W implementacji wykorzystaj utworzone wcześniej funkcje pomocnicze
- Wywołaj utworzoną funkcję dla wczytanej wcześniej daty i sprawdź jej działanie

ĆWICZENIE 10.2:

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 poprzedniego zadania zastępując typowanie dynamiczne typowaniem statycznym

ALGORYTM WYKONANIA:

- Skopiuj rozwiązanie poprzedniego ćwiczenia
- 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

ĆWICZENIE 10.3:

"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 9.2 utwórz uniwersalne funkcje potrafiące eliminować z sekwencji elementy niespełniające podanego predykatu oraz przekształcające elementy sekwencji zgodnie z podaną funkcją
- Wykorzystaj te funkcje do rozwiązania zadania

ALGORYTM WYKONANIA:

- Z ćwiczenia 9.2 skopiuj dane wejściowe (słowniki i listę)
- Utwórz funkcję *sito*, która:
  - przyjmie 2 argumenty: predykat (czyli funkcję zwracającą wartość logiczną) oraz sekwencję
  - funkcja powinna zwrócić te elementy sekwencji, które spełniają podany predykat
- Utwórz funkcję *transformacja*, która:
  - przyjmie 2 argumenty: funkcję (przekształcającą kolejny element i zwracającą wynik) oraz sekwencję
  - funkcja powinna zwrócić przekształcone elementy sekwencji
- Wykorzystaj te funkcje do zrealizowania tych samych zapytań co w ćwiczeniu 9.2
- Tam, gdzie to konieczne zdefiniuj funkcje dla argumentów funkcji *sito* oraz *transformacja*
- Porównaj otrzymane wyniki ze wcześniejszymi

ĆWICZENIE 10.4:

"Baza" osób – standardowe 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 wyrażeń lambda

CELE I ZADANIA:

- Wzorując się na rozwiązaniu poprzedniego ćwiczenia zastąp funkcje *sito* i *transformacja* funkcjami standardowymi *filter* i *map*
- Tam, gdzie to możliwe, użyj wyrażeń lambda

ALGORYTM WYKONANIA:

- Skopiuj rozwiązanie poprzedniego ćwiczenia
- Usuń funkcje *sito* i *transformacja* i zastąp je funkcjami standardowymi *filter* i *map*
- Jako argumenty dla tych funkcji (w miejsce predykatu oraz funkcji) przekaż wyrażenia lambda
- To powinno uprościć kod i spowodować, że jego użycie będzie bardziej uniwersalne (łatwiej jest wywołać tę samą funkcję przekazując inny predykat czy funkcję zapisaną jako wyrażenie lambda)
- Porównaj otrzymane wyniki ze wcześniejszymi

ĆWICZENIE 10.5:

"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 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
- Pozostaw tylko dane (słowniki i listę)
- Wypisz na ekranie katualny 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

11 Programowanie zorientowane obiektowo

ĆWICZENIE 11.1:

Definiowanie klas i obiektów

UMIEJĘTNOŚCI:

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

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 obiektu 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 przez wprowadzeniem bezsensownych wartości (np. roku urodzenia pochodzącego z przyszłości)?
- Zobacz, jak wtedy zachowa się program

ĆWICZENIE 11.2:

Hermetyzacja

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - hermetyzowania klas

CELE I ZADANIA:

- Dokonaj hermetyzacji klas z poprzedniego ćwiczenia

ALGORYTM WYKONANIA:

- Zmodyfikuj klasę z poprzedniego ćwiczenia (*Osoba*
- 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 11.3:

Wykorzystanie właściwości

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

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

ALGORYTM WYKONANIA:

- Zmodyfikuj klasę (*Osoba*) z poprzedniego ćwiczenia
- 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

12 Relacja dziedziczenia

ĆWICZENIE 12.1:

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
- Zdefiniuj w klasie kierownika metody umożliwiające poszerzenie lub zmniejszenie zespołu (np. metody: *dodaj_pracownika* i *usun_pracownika*)

- Wreszcie przedefiniuj metodę zwracającą opis kierownika – powinien on być zbieżny z opisem pracownika (również wykorzystaj delegację) i poszerzony o informację o odpowiedzialności
- Sprawdź działanie aplikacji

ĆWICZENIE 12.2:

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 11.1
- 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
- Przykładowo:

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

- Przetestuj działanie zdefiniowanych operacji

13 Wyjątki

ĆWICZENIE 13.1:

Osoby – użycie wyjątków standardowych

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- Wykorzystaj rozwiązanie ćwiczenia 11.3
- Użyj wyjątków standardowych do sygnalizacji sytuacji wyjątkowych
- Dodaj obsługę wyjątków

ALGORYTM WYKONANIA:

- Wykorzystaj rozwiązanie ćwiczenia 11.3
- 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 lub zamiast wartości logicznej podamy liczbę)?
- Dodaj obsługę nowego wyjątku i uruchom ponownie program

ĆWICZENIE 13.2:

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
- Zastąp standardową klasę sygnalizującą błędne wartości – własnymi klasami

ALGORYTM WYKONANIA:

- Zmodyfikuj rozwiązanie poprzedniego ćwiczenia
- Utwórz dedykowane klasę wyjątku 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 obiektu wyjątku
- Zastąp standardową klasę wyjątku własnymi i sprawdź działanie programu

ĆWICZENIE 13.3:

Telefon na kartę – sytuacje wyjątkowe

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
  - definiowania i ywkorzystania 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

14 Moduły i pakiety

ĆWICZENIE 14.1:

Użycie modułów i pakietów

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- Utwórz klasę reprezentującą adres (ulica, kod pocztowy, miasto) 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 *address.py* klasę *Address*
- Zadeklaruj w niej atrybuty instancyjne reprezentujące: ulicę, kod pocztowy i miasto
- Zdefiniuj metody umożliwiające reprezentację tekstową obiektów
- Utwórz w module *person.py* klasę *Person*
- Zadeklaruj w niej atrybuty instancyjne reprezentujące: imię, nazwisko i adres
- Zdefiniuj metody umożliwiające reprezentację tekstową obiektów
- W module *people.py* utwórz listę kilku osób
- Wykorzystaj instrukcje importu
- Przedstaw dane tych osób, wykorzystując ich reprezentacje tekstowe

15 Operacje wejścia-wyjścia

ĆWICZENIE 15.1:

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 wiersz w dwóch częściach
- 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 wiersza
- Napisz program, który umożliwi przeczytanie zawartości obu plików (jeden za drugim)
- Wyświetl treść na ekranie
- 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

- 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, druga linia z pliku 1, druga linia z pliku 2, itd.
- Zobacz, jaką metamorfozę przeszła prezentowana treść...

ĆWICZENIE 15.2:

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 rozwiązanie poprzedniego zadania
- Przeczytaj zawartość dwóch części wiersza
- Zapisz ją w nowym pliku
- Zadbaj o zamknięcie wszystkich strumieni

ĆWICZENIE 15.3:

Utrwalanie i odtwarzanie obiektów

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętność:
  - konwersji obiektów na postać tekstową
  - zapisu i odczytu danych z pliku
  - odtwarzania obiektów na podstawie ich reprezentacji tekstowej

CELE I ZADANIA:

- Wykorzystaj kod z ćwiczenia 14.1
- Utwórz listę osób, a następnie skonwertuj ją na tekst
- Zapisz tę informację w pliku
- Odczytaj tekst z pliku
- Odtwórz instancje na podstawie reprezentacji tekstowych

ALGORYTM WYKONANIA:

- Wykorzystaj kod z ćwiczenia 14.1 (klasy *Address* oraz *Person*)
- Utwórz listę zawierającą dane kilku osób
- Zapisz reprezentacje tekstowe tych osób w pliku
- Następnie odczytaj te dane z pliku
- Wykorzystując funkcję *eval* odtwórz obiekty na podstawie ich reprezentacji
- Przedstaw dane tych osób na ekranie

16 Dołączone baterie (stdlib)

ĆWICZENIE 16.1:

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 6.1
- 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 16.2:

Lotto

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- W oparciu o dokumentację funkcji specjalnych biblioteki *scipy* napisz program, który wyliczy szansę wylosowania k liczb spośród n różnych liczb

ALGORYTM WYKONANIA:

- Napisz program o podobnej funkcjonalności do programu z ćwiczenia 3.3
- Wykorzystaj możliwości funkcji specjalnych (dot. kombinatoryki) biblioteki *scipy*
- Jeśli moduł nie jest zainstalowany:
  - to z wiersza linii poleceń wydaj polecenie:
    `python -m pip install scipy`
  - w przypadku IDE *PyCharm* trzeba wybrać z menu opcję: *File* → *Settings...* → *Project* → *Project Interpreter*
  - można też dodać instrukcję importu, a gdy pojawi się błąd wykorzystać proponowane rozwiązanie problemu – instalację pakietu
- Dokumentację znajdziesz pod adresem:
  `https://docs.scipy.org/doc/scipy/reference/special.html`
- W programie dodaj na początku instrukcję importu:
  import scipy.special
- Porównaj otrzymane wyniki z tymi z wcześniejszego ćwiczenia

ĆWICZENIE 16.3:

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 7.4
- Wykorzystaj możliwości funkcji biblioteki *numpy*
- Jeśli moduł nie jest zainstalowany, to go zainstaluj postępując podobnie jak w poprzednim ćwiczeniu
- 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 listę z nazwami miesięcy
- Przekształć ją na tablicę
- Przeorganizuj tablicę jednowymiarową (1x12) na dwuwymiarową (4x3)
- Dokonaj konwersji na listę
- Porównaj otrzymane wyniki z tymi z wcześniejszego ćwiczenia