

WBUDOWANE TYPY DANYCH, OPERATORY, WBUDOWANE FUNKCJE I METODY

ĆWICZENIA DODATKOWE MODUŁ 3

AltKom Akademia S.A., materiały własne

1 PODSTAWOWE WBUDOWANE TYPY DANYCH

ĆWICZENIE 1.1:

Odcinki

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
 - użycia instrukcji warunkowych

CELE I ZADANIA:

- Napisz program, który dla dwóch odcinków poziomych określi, czy są one rozłączne
- Jeśli odcinki nie są rozłączne, to trzeba wyznaczyć część wspólną obu odcinków oraz ich sumę
- Należy też sprawdzić, czy dane są poprawne

ALGORYTM WYKONANIA:

- Zdefiniuj i zainicjuj współrzędne obu odcinków – do ich opisu wystarczą dwie liczby (początek i koniec), gdyż odcinki są poziome
- Sprawdź, czy dane są poprawne, np.:
 - czy odcinek nie ma zerowej długości (początek pokrywa się z końcem)
 - czy współrzędna początku nie jest większa od współrzędnej końca
- Porównując współrzędne (użyj instrukcji warunkowych) wyznacz część wspólną oraz sumę obu odcinków
- Jeśli odcinki nie mają części wspólnej (są rozłączne), to wypisz stosowny komunikat
- Możesz trochę uprościć rozwiązanie zastępując niektóre instrukcje warunkowe, wykorzystaniem wbudowanych funkcji *min* i *max*

ĆWICZENIE 1.2:

NWD (algorytm klasyczny)

UMIEJĘTNOŚCI:

- Wykonanie ćwiczenia pozwoli na zapoznanie się z różnymi instrukcjami sterującymi

CELE I ZADANIA:

- Napisz program wyliczający największy wspólny dzielnik (NWD) dla dwóch podanych liczb całkowitych
- Wczytaj obie liczby z klawiatury
- Wypisz wynik na ekranie

ALGORYTM WYKONANIA:

- Wprowadź z klawiatury dwie liczby całkowite
- Wybierz większą z tych liczb i sprawdź, czy dzieli się bez reszty przez drugą z liczb – wtedy NWD będzie równy mniejszej liczbie
- W przeciwnym razie w pętli testuj podzielność obu liczb przez kolejne potencjalne dzielniki
- Rozpocznij od połowy wartości mniejszej z liczb, a w kolejnych iteracjach zmniejszaj ten dzielnik o 1
- Przerwij działanie pętli w momencie znalezienia pierwszego dzielnika obu liczb
- Jeśli dzielnik nie zostanie znaleziony to zwróć wartość 1
- Wypisz na ekranie obie liczby i ich największy wspólny dzielnik

ĆWICZENIE 1.3:

NWD (algorytm Euklidesa)

UMIEJĘTNOŚCI:

- Wykonanie ćwiczenia pozwoli na zapoznanie się z różnymi instrukcjami sterującymi

CELE I ZADANIA:

- Napisz program wyliczający największy wspólny dzielnik (NWD) dla dwóch podanych liczb całkowitych
- W rozwiązaniu wykorzystaj algorytm Euklidesa (opis algorytmu znajdziesz pod adresem: https://pl.wikipedia.org/wiki/Algorytm_Euklidesa)
- Wczytaj obie liczby z klawiatury
- Wypisz wynik na ekranie

ALGORYTM WYKONANIA:

- Wprowadź z klawiatury dwie liczby całkowite
- Wybierz większą z tych liczb i sprawdź, czy dzieli się bez reszty przez drugą z liczb – wtedy NWD będzie równy mniejszej liczbie
- W przeciwnym razie zmień role liczb:
 - oblicz resztę z dzielenia większej liczby przez mniejszą
 - podstaw mniejszą liczbę do większej
 - wyliczoną resztę z dzielenia przypisz mniejszej liczbie
- Ponownie sprawdź podzielność
- Postępuj tak, aż do momentu znalezienia wspólnego dzielnika
- Wypisz na ekranie obie liczby i ich największy wspólny dzielnik

2 ZŁOŻONE TYPY DANYCH

ĆWICZENIE 2.1:

Stacje pomiarowe*

Rozwiązanie zadania może się przydać do projektu zaliczeniowego :-)

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
 - wykorzystania kontenerów danych
 - odwoływania się do elementów kontenerów
 - wykorzystania wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który zdefiniuje zestaw danych charakteryzujących położenie stacji pomiarowych do monitorowania jakości powietrza
- Następnie przy użyciu wyrażeń listowych wybierz stacje spełniające zadane kryteria

ALGORYTM WYKONANIA:

- Wykorzystując poznane kontenery (takie jak: listy, słowniki, ...) zaprojektuj strukturę danych reprezentującą stacje pomiarowe służące do monitorowania jakości powietrza
- Każda stacja pomiarowa jest w pełni opisana przez:
 - *id*: unikalny identyfikator (liczba całkowita), jednoznacznie odnoszący się do danej stacji
 - *stationName*: nazwa stacji
 - *gegrLat*: szerokość geograficzna w stopniach (docelowo będzie to liczba z częścią ułamkową, ale na razie potraktujmy, że ta wartość jest zapisana jako tekst, np. '52.143250')
 - *gegrLon*: długość geograficzna w stopniach (docelowo będzie to liczba z częścią ułamkową, ale na razie potraktujmy, że ta wartość jest zapisana jako tekst, np. '19.233225')
 - *city*: opis miejscowości, na który składają się:
 - * *id*: unikalny identyfikator (liczba całkowita), jednoznacznie odnoszący się do danej miejscowości
 - * *name*: nazwa miejscowości
 - * *commune*: gmina, opisana przez:
 - *communeName*: nazwę gminy
 - *districtName*: nazwę powiatu
 - *provinceName*: nazwę województwa
 - *addressStreet*: nazwa ulicy

- Uwaga: mimo, że podane nazwy atrybutów stacji nie spełniają zalecanej konwencji nazewnictwa, należy je pozostawić – będzie mniej problemów przy dalszej rozbudowie funkcjonalności
- Zastanów się, jak te dane można byłoby opisać w Pythonie
- Następnie porównaj, czy Twoja koncepcja jest zbieżna z przykładowym zestawem danych dostarczonym przez instruktora
- Wykorzystując wyrażenia listowe zaprojektuj kilka “zapytań” (kwerend), które wybiorą stacje pomiarowe spełniające zadane kryteria
- Przykładowo:
 - wszystkie stacje pomiarowe znajdujące się w podanej miejscowości/ gminie/powiecie/...
 - spróbuj wymyśleć i zaimplementować jeszcze inne kryteria. . .

ĆWICZENIE 2.2:**Stanowiska pomiarowe***

Rozwiązanie zadania może się przydać do projektu zaliczeniowego :-)

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
 - wykorzystania kontenerów danych
 - odwoływania się do elementów kontenerów
 - wykorzystania wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który zdefiniuje zestaw danych charakteryzujących stanowiska pomiarowe znajdujące się w stacjach pomiarowych
- Następnie przy użyciu wyrażeń listowych wybierz stanowiska spełniające zadane kryteria

ALGORYTM WYKONANIA:

- Wykorzystując poznane kontenery (takie jak: listy, słowniki, ...) zaprojektuj strukturę danych reprezentującą stanowiska pomiarowe znajdujące się w stacjach pomiarowych
- Każda stacja może zawierać inną liczbę stanowisk pomiarowych (czujników) mierzących różne parametry jakości powietrza
- Każde stanowisko pomiarowe jest w pełni opisane przez:
 - *id*: unikalny identyfikator (liczba całkowita) jednoznacznie odnoszący się do danego stanowiska
 - *stationId*: identyfikator stacji (liczba całkowita) w której znajduje się to stanowisko
 - *param*: parametr mierzony na danym stanowisku, opisany przez:
 - * *paramName*: nazwa mierzonego parametru (np. dwutlenek azotu, pył zawieszony PM10, itp.)
 - * *paramFormula*: wzór parametru (np. NO₂, PM10, ...)
 - * *paramCode* – kod parametru (np. NO₂, PM10, ...)
 - * *idParam*: unikalny identyfikator parametru (liczba całkowita)
- Uwaga: mimo, że podane nazwy atrybutów stanowisk nie spełniają zalecanej konwencji nazewnictwa, należy je pozostawić – będzie mniej problemów przy dalszej rozbudowie funkcjonalności
- Zastanów się, jak te dane można byłoby opisać w Pythonie
- Następnie porównaj, czy Twoja koncepcja jest zbieżna z przykładowym zestawem danych dostarczonym przez instruktora
- Wykorzystując wyrażenia listowe zaprojektuj kilka “zapytań” (kwerend), które wybiorą stanowiska spełniające zadane kryteria

- Przykładowo:
 - wszystkie parametry, jakie jest w stanie mierzyć dana stacja pomiarowa
 - spróbuj wymyśleć jeszcze inne kryteria...

ĆWICZENIE 2.3:

Dane pomiarowe*

Rozwiązanie zadania może się przydać do projektu zaliczeniowego :-)

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
 - wykorzystania kontenerów danych
 - odwoływania się do elementów kontenerów
 - wykorzystania wyrażeń listowych

CELE I ZADANIA:

- Napisz program, który zdefiniuje zestaw danych pomiarowych zebranych przez dane stanowiskopomiarowe
- Następnie wykorzystując wyrażenia listowe oblicz wybrane charakterystyki statystyczne opisujące zestaw danych (np. wartość najmniejszą, największą, średnią, trend, itp.)

ALGORYTM WYKONANIA:

- Wykorzystując poznane kontenery (listy, krotki, słowniki) zaprojektuj strukturę danych reprezentującą dane pomiarowe zebrane przez dane stanowisko pomiarowe
- Każde stanowisko zbiera dane pomiarowe z zadaną częstotliwością (np. co godzinę)
- Zestaw danych jest w pełni opisany przez:
 - *key*: kod parametru
 - *values*: sekwencję pojedynczych pomiarów, z których każdy jest opisany przez:
 - * *date*: datę i czas pomiaru (wartość tekstowa, np.: "2017-03-28 12:00:00")
 - * *value*: wartość pomiaru (liczba z częścią ułamkową)
- Uwaga: mimo, że podane nazwy atrybutów danych pomiarowych nie spełniają zalecanej konwencji nazewnictwa, należy je pozostawić – będzie mniej problemów przy dalszej rozbudowie funkcjonalności
- Zastanów się, jak te dane można byłoby opisać w Pythonie
- Następnie porównaj, czy Twoja koncepcja jest zbieżna z przykładowym zestawem danych dostarczonym przez instruktora
- W oparciu o posiadane dane spróbuj wyciągnąć jakieś wnioski
- Przykładowo:
 - utwórz listę danych pomiarowych z danego dnia
 - wyszukaj datę i godzinę, gdy mierzony parametr miał wartość najmniejszą i największą
 - oblicz wartość średnią w danym dniu
 - możesz spróbować znaleźć jeszcze inne dane charakteryzujące zebrany zestaw danych...

- Uwaga:
 - pamiętaj, że w zestawie danych mogą występować wartości *None*, gdy o danej porze pomiar się nie powiódł
 - zapoznaj się, ze składnią funkcji *min* i *max*
 - aby obliczyć wartość średnią możesz skorzystać z modułu *statistics*

ĆWICZENIE 2.4:

Czynniki pierwsze

UMIEJĘTNOŚCI:

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

CELE I ZADANIA:

- Napisz program, który podaną liczbę całkowitą zapisze jako iloczyn liczb pierwszych

ALGORYTM WYKONANIA:

- Wczytaj z klawiatury liczbę całkowitą
- Utwórz listę, w której będą zapamiętywane kolejne czynniki pierwsze
- Zainicjuj listę wartością 1 (każda liczba jest podzielna przez 1)
- W pętli testuj podzielność liczby przez potencjalne dzielniki (w porządku narastającym)
- Pierwszym testowanym dzielnikiem powinna być 2
- Zastanów się, jaką wartość powinni mieć maksymalny potencjalny dzielnik
- Uzasadnij odpowiedź, wyjaśniając, dlaczego przyjęte rozwiązanie jest najbardziej efektywne (potencjalnie najmniejsza liczba iteracji)
- W momencie znalezienia dzielnika:
 - dodaj go do listy
 - przerwij wykonywanie pętli
 - podziel liczbę przez znaleziony dzielnik i powtórz operacje wyszukiwania podzielników dla nowej liczby
- Operacja wyszukiwania dzielników zakończy się, jeśli w pętli nie zostanie znaleziony żaden dzielnik – wtedy do listy trzeba dołączyć jako ostatni element testowaną liczbę
- Wypisz na ekranie wczytaną liczbę i jej czynniki pierwsze
- Zastanów się, jak efektywnie wypisać elementy zawarte w liście

ĆWICZENIE 2.5:**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

ĆWICZENIE 2.6:

Liczba słownie

UMIEJĘTNOŚCI:

- Po wykonaniu ćwiczenia zdobędziesz umiejętności:
 - wykorzystania kontenerów danych
 - odwoływania się do elementów kontenerów
 - operacji na kontenerach

CELE I ZADANIA:

- Napisz program, który podaną liczbę całkowitą z zakresu od 0 do 999 zapisze słownie (po polsku)

ALGORYTM WYKONANIA:

- Celem zadania jest zamiana podanej liczby całkowitej z przedziału $[0; 999]$ na jej zapis słowny
- Utwórz krotki zawierające odpowiednie liczebniki słownie (zastanów się które)
- Z konwertowanej liczby wyodrębnij cyfry reprezentujące setki, dziesiątki i jedności
- W oparciu o nie zbuduj zapis słowny
- Kolejne liczebniki możesz dodać do listy, a następnie scalić jej elementy w jeden tekst
- Przykładowe działanie programu:

```
0 = zero
13 = trzynaście
80 = osiemdziesiąt
999 = dziewięćset dziewięćdziesiąt dziewięć
102 = sto dwa
```