BIBLIOTEKA PANDAS

1. STRUKTURY DANYCH W BIBLIOTECE PANDAS

Serie danych (Series) – to jednowymiarowa tablica z etykietami, która może przechowywać dowolny typ danych.

Dokumentacja dla serii danych https://pandas.pydata.org/pandas-docs/stable/generated/pandas.Series.html#pandas.Series

Ramka danych (DataFrame) – dwuwymiarowa struktura z etykietami, mogąca przechowywać kolumny z różnymi typami danych.

Dokumentacja dla ramek - https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html#pandas.DataFrame

Listing 1 – tworzenie Series i DataFrames

```
import pandas as pd
import numpy as np

# Series
s = pd.Series([1, 3, 5, np.nan, 6, 8])
print(s)
s = pd.Series([10, 12, 8, 14], index=['Ala', 'Marek', 'Wiesiek',
'Eleonora'])
print(s)

# DataFrame
# tworzenie dataframe na podstawie słownika
data = {'Kraj': ['Belgia',  'Indie',  'Brazylia'],
'Stolica': ['Bruksela',  'New Delhi',  'Brasilia'],
'Populacja': [11190846, 1303171035, 207847528]}
df = pd.DataFrame(data,columns=['Kraj',  'Stolica',  'Populacja'])
print(df)
# DataFrame przechowuje typ dla każdej kolumny co możemy sprawdzić
wypisując
print(df.dtypes)
# możemy również w prosty sposób stworzyć serię danych - czyli próbki dla
kolejnych
# dat, pomiarów czasu
daty = pd.date_range('20180401', periods=5)
print(daty)
df = pd.DataFrame(np.random.randn(5,4), index=daty, columns=list('ABCD'))
print(df)

# biblioteka Pandas umożliwia również tworzenie DataFrame'ów z zewnętrznych
źródeł danych
# CSV, odczyt i zapis
```

```
df = pd.read_csv('dane.csv', header=None, nrows=2)
print(df)
df.to_csv('plik.csv', header=None, index=False)

# Excel - wymagana bibioteka xlrd oraz openpyxl
import xlrd
import openpyxl

xlsx = pd.ExcelFile('dane.xlsx')
df = pd.read_excel(xlsx,'Punktacja')
print(df)
df.to_excel('z_indeksami.xlsx',  sheet_name='Wydatki z indeksami')

# biblioteka Pandas może również dzięki bibliotece sqlalchemy odczytywać
# dane bezpośrednio z baz danych, lub zapisywać je do SQL-a
# ten temat wykracza jednak poza zakres aktualnej lekcji i może
# zostanie zaprezentowany w lekcji polegającej na wyświetlaniu wykresów na
postawie
# danych pochodzących z plików i zewnętrznych źródeł
```

2. POBIERANIE WYBRANYCH DANYCH

Pandas dostarcza wielu sposobów na pobieranie pojedynczych wartości, kolumn, wierszy lub zbiorów wartości na podstawie parametrów. Poniżej znajdują się 2 listingi z najbardziej popularnymi metodami.

Listing 2 – wyświetlanie danych

```
import pandas as pd
import numpy as np

seria = pd.Series([10, 12, 8, 14], index=['Ala', 'Marek', 'Wiesiek',
'Eleonora'])

data = {'Kraj': ['Belgia',  'Indie',  'Brazylia'],
'Stolica': ['Bruksela',  'New Delhi',  'Brasilia'],
'Populacja': [11190846, 1303171035, 207847528]}
df = pd.DataFrame(data, columns=['Kraj',  'Stolica',  'Populacja'])


# pojedynczy element Series
print(seria['Wiesiek'])
# możemy również dostać się do wartości serii jak do pola klasy
print(seria.Wiesiek)

# tak jak przy cięciu list
print(df[1:])

# kolumna po etykiecie
print(df['Populacja'])
# pojedyncza kolumna ze zbioru kolumn
print(df.ix[:, 'Stolica'])

# pobieranie pojedynczej wartości po indeksie kolumny i wiersza
print(df.iloc[[0], [0]])
```

```
# pobieranie wartości po indeksie wiersza i etykiecie kolumny
print(df.loc[[0],  ['Kraj']])
print(df.at[0, 'Kraj'])

# pojedynczy wiersz
print(df.ix[2])

# wybiera kolumnę z podanego wiersza
print(df.ix[1, 'Stolica'])

# podobnie jak w przypadku serii można odwoływać się do kolumn jak do pól
klasy
# dodatkowo print jest wywoływany jak w pętli dla każdego elementu danej
kolumny
print('kraj: ' + df.Kraj)

# Pandas posiada również funckje pozwalające na losowe pobieranie elementów
lub
# w odniesieniu do procentowej wielkości całego zbioru

# jeden losowy element
print(df.sample())
# n losowych elementów
print(df.sample(2))
# ilość elementów procentowo - uwaga na zaokrąglanie
print(df.sample(frac=0.5))

# jeżeli potrzeba nam więcej próbek niż znajduje się w zbiorze i
dopuszczamy duplikaty
# to możemy użyć parametru replace, który będzie losował z powtórzeniami
print(df.sample(n=10, replace=True))

# zamiast wyświetlać całą kolekcję możemy wyświetlić tylko określoną liczbę
pierwszych lub ostatnich elementów
print(df.head())
print(df.head(2))
print(df.tail(1))

# sprawdź wartości atrybutów index, columns, values dla Series lub
DataFrame
# Pandas jest też w stanie wyświetlić statystykę dla wartości, które dana
kolekcja zawiera, o ile są jakieś kolumny
# z danymi numerycznymi

print(df.describe())
# transpozycja to zmienna T kolekcji, podobnie jak w Numpy
print(df.T)
```

Więcej przykładów pobierania elementów poprzez indeksowanie można znaleźć pod adresem
https://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-integer

Listing 3 – filtrowanie, grupowanie i agregowanie danych

```
import pandas as pd
import numpy as np

seria = pd.Series([10, 12, 8, 14], index=['Ala', 'Marek', 'Wiesiek',
'Eleonora'])

data = {'Kraj': ['Belgia',  'Indie',  'Brazylia'],
'Stolica': ['Bruksela',  'New Delhi',  'Brasilia'],
'Populacja': [11190846, 1303171035, 207847528]}
df = pd.DataFrame(data, columns=['Kraj',  'Stolica',  'Populacja'])

# wyświetla dane z serii gdzie wartość większa od 1
print(seria[(seria > 1)])
# nieco inny efekt można osiągnąć qykorzystując funkcję where, która zwraca
kolekcję w oryginalnej wielkości (liczebności)
# elementów, ale wartości nie spełniające warunku uzupełnia wartością NaN
print(seria.where(seria > 10))
# możemy również podać wartość, która zostanie podstawiona zamiast NaN
print(seria.where(seria > 10, 'za duze'))
# jeszcze inna własność where pozwala na modyfikowanie oryginalnego zbioru
(domyślnie zwaracana jest kopia)
seria2 = seria.copy()
seria2.where(seria2 > 10, 'za duze', inplace=True)
print('############')
print(seria2)

# wyświetla dane z serii gdzie wartość nie jest większa od 10
print(seria[~(seria > 10)])
# warunki możemy ze sobą łączyć
print(seria[(seria < 13) & (seria > 8)])

# warunki pobierania danych dla DataFrame
print(df[df['Populacja']>1200000000])
# bardziej skomplikowane warunki
print(df[((df.Populacja > 1000000) & (df.index.isin([0,2])))])

# inny przykład z listą dopuszczalnych wartości oraz isin zwracająca
wartości boolowskie
print('######')
szukaj = ['Belgia', 'Brasilia']
print(df.isin(szukaj))


# zmmiana, usuwanie i dodawanie danych

# w przypadku serii możemy dodać/zmienić wartość poprzez odwołanie sie do
elementu serii przez klucz (indeks)
seria['Wiesiek'] = 15
print(seria.Wiesiek)
seria['Alan'] = 16
print(seria)

# podobna operacja dla DataFrame ma nieco inny efekt - wartość ustawiona
dla wszystkich kolumn
df.loc[4] = 'dodane'
print(df)
# ale można dodać wiersz w postaci listy
df.loc[5] = ['Polska', 'Warszawa', 38675467]
print(df)
```

```
# usuwanie danych można wykonać przez funkcję drop, ale pamiętajmy, że
operacja nie wykonuje się in-place więc
# zwracana jest kopia DataFrame z usuniętymi wartościami
new_df = df.drop([4])
print(new_df)
# więc jeżeli chcemy zmienić pierwotny zbiór dodajemy parametr inplace=True
df.drop([4], inplace=True)
# można usuwać również całe kolumny po nazwie indeksu, ale wykonanie tej
komendy uniemożliwi
# wykonanie dalszego kodu (można przetstować po zakomentowaniu dalszej
części listingu)
# df.drop('Kraj', axis=1, inplace=True)
print(df)

# do DataFrame możemy dodawać również kolumny zamiast wierszy
df['Kontynent'] = ['Europa', 'Azja', 'Ameryka Południowa', 'Europa']
print(df)

# Pandas ma również własne funkcje sortowania danych
print(df.sort_values(by='Kraj'))

# grupowania
grouped = df.groupby(['Kontynent'])
print(grouped.get_group('Europa'))
# można też jak w SQL czy Excelu uruchomić funkcje agregujące na danej
kolumnie
print(df.groupby(['Kontynent']).agg({'Populacja':['sum']}))
# więcej przykładów można znaleźć pod adresem
https://pandas.pydata.org/pandas-
docs/stable/generated/pandas.core.groupby.DataFrameGroupBy.agg.html

# podobny mechanizm to sumy częściowe i tabele przestawne znane z Excela,
które w Pandas również mają swoje odpowiedniki
print("_____ sumy częściowe _______")
tabela =
pd.pivot_table(df,values=['Populacja'],index=['Kontynent'],columns=['Kraj']
,aggfunc=np.sum,margins=True)
print(tabela.stack('Kraj'))
```

Więcej przykładów i możliwości biblioteki Pandas można znaleźć w obszernym dokumencie dostępnym pod adresem https://pandas.pydata.org/pandas-docs/stable/cookbook.html#dataframes

Zadanie 1

Wczytaj do DataFrame arkusz z narodzinami dzieci w Polsce dostępny pod adresem https://dane.gov.pl/dataset/219.

Zadanie 2

Z danych z zadania 1 wyświetl (korzystając w miarę możliwości z funkcji biblioteki Pandas):

- tylko te rekordy gdzie liczba nadanych imion była większa niż 1000 w danym roku

- tylko rekordy gdzie nadane imię jest takie jak Twoje

- sumę wszystkich urodzonych dzieci w całym danym okresie,

- sumę dzieci urodzonych w latach 2000-2005

- sumę urodzonych chłopców i dziewczynek,

- najbardziej popularne imię dziewczynki i chłopca w danym roku (czyli po 2 rekordy na rok),

- najbardziej popularne imię dziewczynki i chłopca w całym danym okresie,

Zadanie 3

Wczytaj plik http://wmii.uwm.edu.pl/~kropiak/wd/zamowienia.csv a następnie wyświetl:

- listę unikalnych nazwisk sprzedawców (przetwarzając zwróconą pojedynczą kolumnę z DataFrame)

- 5 najwyższych wartości zamówień

- ilość zamówień złożonych przez każdego sprzedawcę

- sumę zamówień dla każdego kraju

- sumę zamówień dla roku 2005, dla sprzedawców z Polski

- średnią kwotę zamówienia w 2004 roku,

- zapisz dane za 2004 rok do pliku zamówienia_2004.csv a dane za 2005 do pliku zamówienia_2005.csv