```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Sep 23 21:10:53 2019

@author: je
"""

# Åben og læs filen ...
import pandas as pd
import numpy as numpy
import matplotlib.pyplot as plt
import seaborn as sns
import re as re

# Besvarelser angående spm. 1-3 fra Portfolio 1 opdraget

# Min variable sættes til *titanic* - som jeg så læser ind som en Pandas
Dataframe.
# Jeg har valgt at indsætte ektra "print statements(strings) til bedre
overblik/forståelse.

print('\n\Indlæsningen af min csv fil')
# her indlæses min fil fra mit drev, data kan hentes
# via https://github.com/BA-ODS2019/dgk432/tree/master/data
titanic = pd.read_csv("../data/titanic.csv")

# Første step er at undersøge, hvilken type (datasæt) jeg har med at gøre

print('\n\Dokumentation for, at mit datasæt rent faktisk er en Pandas
DataFrame ' )
print(type(titanic))

# Derefter begynder jeg min analyse af datasættet, dels for at undersøge,
hvilke typer
# data jeg har med at gøre, men også for at lære data at kende, så jeg kan
# uddrage enkelte visualiseringer til sidst i processen

# Kilder der er anvendt er:
# https://www.shanelynn.ie/using-pandas-dataframe-creating-editing-
viewing-data-in-python/
# og https://pandas.pydata.org/pandas-docs/stable/index.html

print('\n\Overskrifter på kolonnerne: ' )
# Navnene/værdierne på de overskrifter, som findes i datasættet
# Disse værdier kan der beregnes og analyseres på senere hen
print(titanic.columns.values)

print('\n\Total antal rækker og kolonner i min Pandas DataFrame: ')
# Hvor mange rækker og kolonner beregnes med shape funktionen
print( '\n\Antal rækker og kolonner: ')
print(titanic.shape)
```

```
print('\n\Dimensionerne i min DataFrame: ')
print(titanic.ndim)

print('\n\Samlet antal rækker og total antal kolonner:  ')
# Samlet antal rækker/entries?
print(len(titanic))
print('\n\Samlet antal kolonner: ')
# Antal kolonner?
print(len(titanic.columns))

print('\n\Total antal celler: ')
#Hvor mange 'cells' i tabellen
print(titanic.size)

# Kolonnernes data type - (int, float, string, object...)
print( '\n\Datatypen fra kolonnerne (om det er integer, float eller
object/string) : ')
print( 'kort overblik via funktionen print(titanic.dtypes) :')
print(titanic.dtypes)

# .. med nedenstående kode snip (titanic.info) får man samme overblik som
alle
# de ovenstående koder, inkl type af data og overblik til videre
bearbejdning
print('\n\Hvilken type data min DataFrame indeholder: ')
print( 'mere dybdegående indsigt via funktionen >>>print(titanic.info) :')
print(titanic.info())

# Det er tydeligt, at funktionen >>>print(titanic.info) giver flere
informationer end
# >>>print(titanic.dtypes)

print('\n\nSidste for-analyse funktioner :')
# Beregninger til at undersøge data nærmere, inkl samlet antal, hele
rækker fra DataFrame
print(titanic.head()) # viser kun de 5 første hits/rækker i DataFrame
print(titanic.tail()) # viser de 5 sidste rækker i DataFrame
print(titanic.describe()) # større samlet overblik inkl kolonne data

# Break inden beregning, hvor der anvendes "Descriptive statistics"
print( '\n\nAntal overlevende: ')
# for at finde antal overlevende skal man navigere ned til den rette
kolonne
# og få output derfra - 'Survived' er enten 0 eller 1. 1 = overlevet.
print(titanic['Survived'].sum()) # Total sum of the column values -
overlevende

print( '\nGennemsnitsalder: ')
# sammen princip som ovenstående - nu blot med kolonnen AGE
print(titanic['Age'].mean()) # Mean of the column values - alder..

print( '\n\nMedian - alder: ')
print(titanic['Age'].median()) # Median of the column values - medianen..
```

```
# Find antal kvinder ...
females = titanic[titanic['Sex'] == 'female']
print('\n\nAntal kvinder: ')
print(females)

# Find antal mænd ...
males = titanic[titanic['Sex'] == 'male']
print('\n\nAntal mænd: ')
print(males)

print('\n\nHvordan er fordelingen mellem passagerklasse og overlevende:' )
# For at få fordelingen på passagere på rejseklasse skal man gruppere på..
og sorte værdien
# passagerer fra Pclass and Survived kolonnerne ..
print (titanic[['Pclass', 'Survived']].groupby(['Pclass'],
as_index=False).mean().sort_values(by='Survived', ascending=False))

print('\n\nHvordan er fordelingen så mellem køn og overlevende? : ')
# En lille ændring i forhold til ovenstående
# nu overlevende fra Sex og Survived kolonnerne -
print (titanic[['Sex', 'Survived']].groupby(['Sex'],
as_index=False).mean().sort_values(by='Survived', ascending=False))

# Antal personer på klasse 1, 2 og 3
print('\n\nantal personer på henholdsvis klasse 1, 2 og 3')
print(titanic.groupby('Pclass').count())

# Spørgsmål 4 i opdraget handler om, hvorvidt der er personer med samme
efternavn
# For at kunne foretage denne beregning, skal man anvende split funktionen
# dvs. først split på teksten(string), dernæst beregnes variablen i
forhold til value_count
print( '\n\nSortering pr efternavn: ')
print( '\n\nVed at tage sidste værdi/entry fra kolonnen Name har jeg fået
en liste over efternavne' )

last_names = titanic['Name'].str.split().str[-1]
print(last_names)

print('\n\nHvor mange rejsende har samme efternavn')
print(last_names.value_counts())

# Spørgsmål 5 i Portfolio 1 opgaven lød på at lave en Pivot tabel,
# som viser, hvor mange der rejste på henholdsvis 1, 2 og 3 klasse

print('\n\nAntal overlevende pr. klasse vises som Pivot tabel: ')
pt1 = titanic.pivot_table(columns = 'Pclass', values = 'Survived',
aggfunc='sum')
print(pt1)
pt1.plot(kind='bar')
plt.title("antal overlevende pr. klasse")
plt.savefig("SurvivorsPrClass.png")
```

```
print('\n\nAntal overlevende pr. køn vises som Pivot tabel: ' )
pt2 = titanic.pivot_table(columns = 'Sex', values = 'Survived',
aggfunc='sum')
print(pt2)
pt2.plot(kind='bar')
plt.title("antal overlevende pr. køn")
plt.savefig("SurvivorsPrSex.png")


# Ved også at anvende visualisering via Seaborn biblioteket kan man
tydeligt se, at der
# var mange flere mænd ombord end kvinder - inspiration snuppet fra Kaggle
# https://www.kaggle.com/startupsci/titanic-data-science-solutions
print('\n\nVed hjælp af Seaborn visualisering kan man se forskel i antal
mænd og kvinder om bord Titanic - fordelt på alder' )
t = sns.FacetGrid(titanic, col='Sex')
t.map(plt.hist, 'Age', bins=20)


print('\n\nSidste øvelse, BONUS i forhold til oplæg, er at få samlet de
mange forskellige titler, som passagererne har' )

# som lidt ekstra, kan man samle titler, denne inspiration er fundet via
netsøgning
# og ved brug af RegEx..

# En enkelt definition inkl. brug af RegEx
# kilde: https://regex101.com/ blandt andet..
# og https://www.kaggle.com/startupsci/titanic-data-science-solutions til
inspiration

def get_title(name):
        title_search = re.search('([A-Za-z]+)\.', name)
        # If the title exists, extract and return it.
        if title_search:
                return title_search.group(1)
        return ""

print('\n\nTitler, som de forskellige passagerer benytter sig af')
titanic['Title'] = titanic['Name'].apply(get_title)
print(pd.crosstab(titanic['Title'], titanic['Sex']))

print('\n\nSøg og erstat de mange titler så det forenkles: ' )
titanic['Title'] = titanic['Title'].replace(['Lady', 'Countess','Capt',
'Col',\
        'Don', 'Dr', 'Major', 'Rev', 'Sir', 'Jonkheer', 'Dona'], 'Other')
titanic['Title'] = titanic['Title'].replace('Mlle', 'Miss')
titanic['Title'] = titanic['Title'].replace('Ms','Miss')
titanic['Title'] = titanic['Title'].replace('Mme','Mrs')
titanic['Title'] = titanic['Title'].replace('Master', 'Mr')

print('\nPrint til sidst de samlede titler kombineret med hvorvidt de
overlevede:')
print (titanic[['Title', 'Survived']].groupby(['Title'],
```

```
  as_index=False).mean())
```