Pandas Exercises 2 ¶

TASK: Import pandas

In [3]:
```
# CODE HERE
import pandas as pd
import numpy as np
```

TASK: Read the bank.csv file

In [4]:
```
# CODE HERE
df = pd.read_csv('bank.csv')
type(df)
```

Out[4]:
```
pandas.core.frame.DataFrame
```

TASK: Display the first 5 rows of the data set

In [5]:
```
# CODE HERE
df_5 = df.head(5)
df_5
```

Out[5]:

	age	job	marital	education	default	balance	housing	loan	contact	day
0	30	unemployed	married	primary	no	1787	no	no	cellular	19
1	33	services	married	secondary	no	4789	yes	yes	cellular	11
2	35	management	single	tertiary	no	1350	yes	no	cellular	16
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5

TASK: What is the average (mean) age of the people in the dataset?

In [6]:
```
df_5['age'].mean()
```

Out[6]:
```
37.4
```

TASK: What is the marital status of the youngest person in the dataset?

HINT (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.idxmin.html)

In [7]:
```
s=df.sort_values(by=['age'])
s.iloc[0].marital
```

Out[7]:
```
'single'
```

TASK: How many unique job categories are there?

In [26]:
```
len(df['job'].unique())
```

Out[26]: 12

TASK: How many people are there per job category? (Take a peek at the expected output)

In [24]:
```
df.groupby('job').size()
```

Out[24]:
```
job
admin.           478
blue-collar      946
entrepreneur     168
housemaid        112
management       969
retired          230
self-employed    183
services         417
student           84
technician       768
unemployed       128
unknown           38
dtype: int64
```

TASK: What percent of people in the dataset were married?

In [30]:
```
married_count = df[df['marital'] == 'married'].shape[0]

total_people = df.shape[0]

percentage_married = (married_count / total_people) * 100
print(percentage_married)
```

```
61.86684361866843
```

TASK: There is a column labeled "default". Use pandas' .map() method to create a new column called "default code" which contains a 0 if there was no default, or a 1 if there was a default. Then show the head of the dataframe with this new column.

Helpful Hint Link One (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.map.html)

Helpful Hint Link Two (https://stackoverflow.com/questions/19798153/difference-between-map-applymap-and-apply-methods-in-pandas)

In [44]:
```
def map_default(default_value):
    if default_value == 'no':
        return 0
    elif default_value == 'yes':
        return 1
    else:
        return None

df['default_code'] = df['default'].map(map_default)

df.head()
```

Out[44]:

	age	job	marital	education	default	balance	housing	loan	contact	day
0	30	unemployed	married	primary	no	1787	no	no	cellular	19
1	33	services	married	secondary	no	4789	yes	yes	cellular	11
2	35	management	single	tertiary	no	1350	yes	no	cellular	16
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5

TASK: Using pandas .apply() method, create a new column called "marital code". This column will only contained a shortened code of the possible marital status first letter. (For example "m" for "married" , "s" for "single" etc... See if you can do this with a lambda expression. Lots of ways to do this one!

Hint Link (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.apply.html)

In [46]:
```
df['marital_code'] = df['marital'].apply(lambda x: x[0].lower())

# Show the head of the DataFrame with the new column
df.head()
```

Out[46]:

	age	job	marital	education	default	balance	housing	loan	contact	day
0	30	unemployed	married	primary	no	1787	no	no	cellular	19
1	33	services	married	secondary	no	4789	yes	yes	cellular	11
2	35	management	single	tertiary	no	1350	yes	no	cellular	16
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5

TASK: What was the longest lasting duration?

In [47]:
```
df['duration'].max()
```

Out[47]: 3025

TASK: What is the most common education level for people who are unemployed?

In [52]:
```
unemployed_df = df[df['job'] == 'unemployed']

unemployed_df['education'].value_counts()
```

Out[52]:
```
education
secondary    68
tertiary     32
primary      26
unknown       2
Name: count, dtype: int64
```

TASK: What is the average (mean) age for being unemployed?

In [54]:
```
unemployed_df = df[df['job'] == 'unemployed']

unemployed_df['age'].mean()
```

Out[54]:
```
40.90625
```