2024 BMED223 :: Hands-on instructions

- **Filename: week13_studentID_studentName.py**
- **Deadline: 2024/06/03 23:59**
- **No excuse for late submission**. Prepare for submission in advance.
- There is no limit on the number of submissions, but grading will be based on the final file.
- Use comments (#, #%%, etc.) to separate problems.

서동휘 andy1200@korea.ac.kr
박영주 lime2514@korea.ac.kr

[Problem 1]

1. Load the data

A. The data ***passenger.csv*** includes passenger data related to ship sinking accidents. The data structure is as shown in the table below.

Columns	Descriptions	Values
PassengerId	Passenger ID	
Survived	Survival	0 = No, 1 = Yes
Pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
Gender	Biological sex	
Age	Age in years	
SibSp	# of siblings / spouses aboard the ship	
Parch	# of parents / children aboard the ship	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southamption

B. Load ***passenger.csv*** into a Pandas dataframe.

C. Print information about the dataframe.

D. Print the top 10 rows of the dataframe.

E. Print all columns and index of the dataframe.

2. Data manipulation

A. Remove the columns '*SibSp*' and '*Parch*' from the dataframe.

B. Extract the rows where '*Gender*' is female from the dataframe.

C. Extract the rows with surviving women from the result of 2.B.

D. Calculate the ratio of passengers survived among female passengers and print the result in the form below.

```
>> {%.2f} of the women survived.
```

E. In the same way, calculate and print the ratio of non-surviving male passengers.
```
>> {%.2f} of the men could not survive.
```

F. Divide the dataframe into three separate dataframes, ***p1, p2,*** and ***p3***, based on the value of the '*pclass*' column (without memory sharing).

G. Sort the dataframes ***p1, p2,*** and ***p3*** by *'Age'* column in ascending order.

H. Calculate the mean and standard deviation of *'Age'* for each dataframe (***p1***, ***p2***, and ***p3***), and print them in the following format to express those values to the second decimal place.
```
>> Average age of grade 1: {%.2f} +/- {%.2f}.
>> Average age of grade 2: {%.2f} +/- {%.2f}.
>> Average age of grade 3: {%.2f} +/- {%.2f}.
```

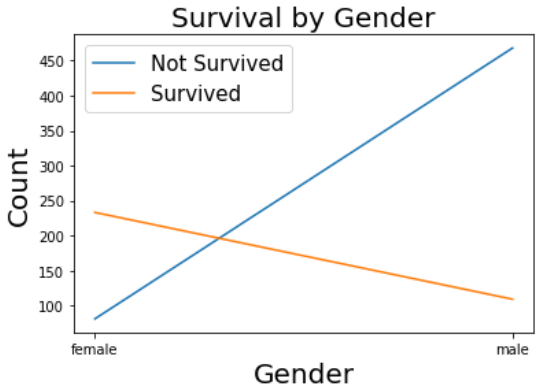
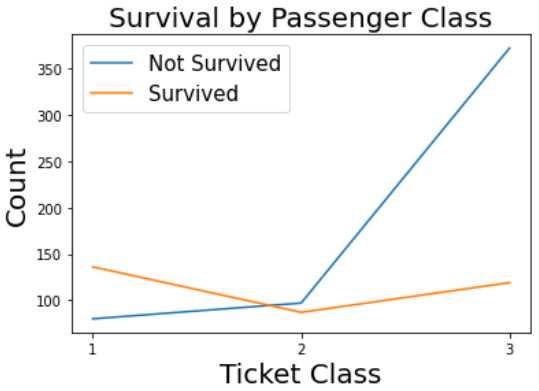
I. Extract the column '*Age*' and '*Embarked*' from ***p1***, ***p2*** and ***p3***, combine them into one dataframe.

J. If the column '*Age*' contains no value (*NaN*), remove the row.

3. Data visualization

A. Plot the gender distribution according to survival status (Survival count vs. Gender)

B. Plot survival depending on age (Survival count vs. Ticket class)

3.A	3.B
	

4. Data storage

A. Save the result of 2.J as an Excel file (***result.xlsx***).

Example for ***result.xlsx***

	Age	Embarked
0	38	C
1	35	S
2	54	S
3	58	S
4	28	S
5	19	S

...

[Problem 2]

Write the program to draw the energy spectrum from ***'Ba133_for_Ehisto.csv'*** by following the step below. The csv file contains total 555,170 rows. All data are in units of eV.

1. Read the CSV file into the dataframe ***'df'***.
2. Add a column ***'E'*** to ***df*** for energy of photons (rows).
   Calculate the energy (E) for each row using the equation below. E is in units of eV. (1 MeV = 1,000 keV = 1,000,000 eV)

$$E = a(X_+ + X_- + Y_+ + Y_-) + b$$
$$\text{where } a = 7.42344 \text{ and } b = -147632.12$$

3. Select the rows that contain the energy between 0 and 1 **MeV**. Then associate it into a dataframe ***'df2'***.
4. Generate the energy histogram (frequency vs energy) from the column ***'E'*** and convert it a dataseries ***'ehist'***. Set the index as the energy in **keV**.
5. Plot the histogram ***'ehist'*** as shown below.

<table>
<tr><th>'Ba133_for_Ehisto.csv'</th><th>2.(2)</th></tr>
<tr><td>

X-	X+	Y-	Y+
9639	9097	15257	6781
11700	6429	7448	10490
12644	20272	12949	20257
5160	11111	7174	6968
5473	8100	6592	7246
6518	7429	6036	8586
13323	6301	6108	19625
19116	6634	7775	17751
8281	31968	10094	20250
4898	16998	5838	13350

...

</td><td>

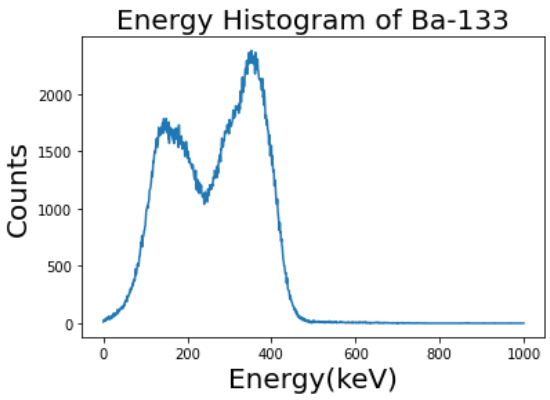
	X-	X+	Y-	Y+	E
0	9639	9097	15257	6781	155051.22256
1	11700	6429	7448	10490	120109.09048
2	12644	20272	12949	20257	343220.57968
3	5160	11111	7174	6968	78136.96072
4	5473	8100	6592	7246	55851.79384
...	...	...	...	...	...
555165	21943	10488	24236	10209	348817.85344
555166	15575	16742	24105	12269	362291.39704
555167	7564	8018	6562	9691	88693.09240
555168	8082	25160	13175	12067	286520.34496
555169	11408	4886	7563	6433	77223.87760

[555170 rows x 5 columns]

</td></tr>
<tr><th>2.(4) : ehist</th><th>2.(5)</th></tr>
<tr><td>

```
Out[736]:
0       12.0
1       29.0
2       32.0
3       30.0
4       22.0
        ...
996      1.0
997      0.0
998      0.0
999      0.0
1000     0.0
Length: 1001, dtype: float64
```

</td><td>



</td></tr>
</table>

fin.