ARTIFICIAL INTELLIGENCE PROJECT

Name: `akshat singh kushwaha`

Class & Section: `11th A`

Roll Number: `2`

Admission Number: `115046`

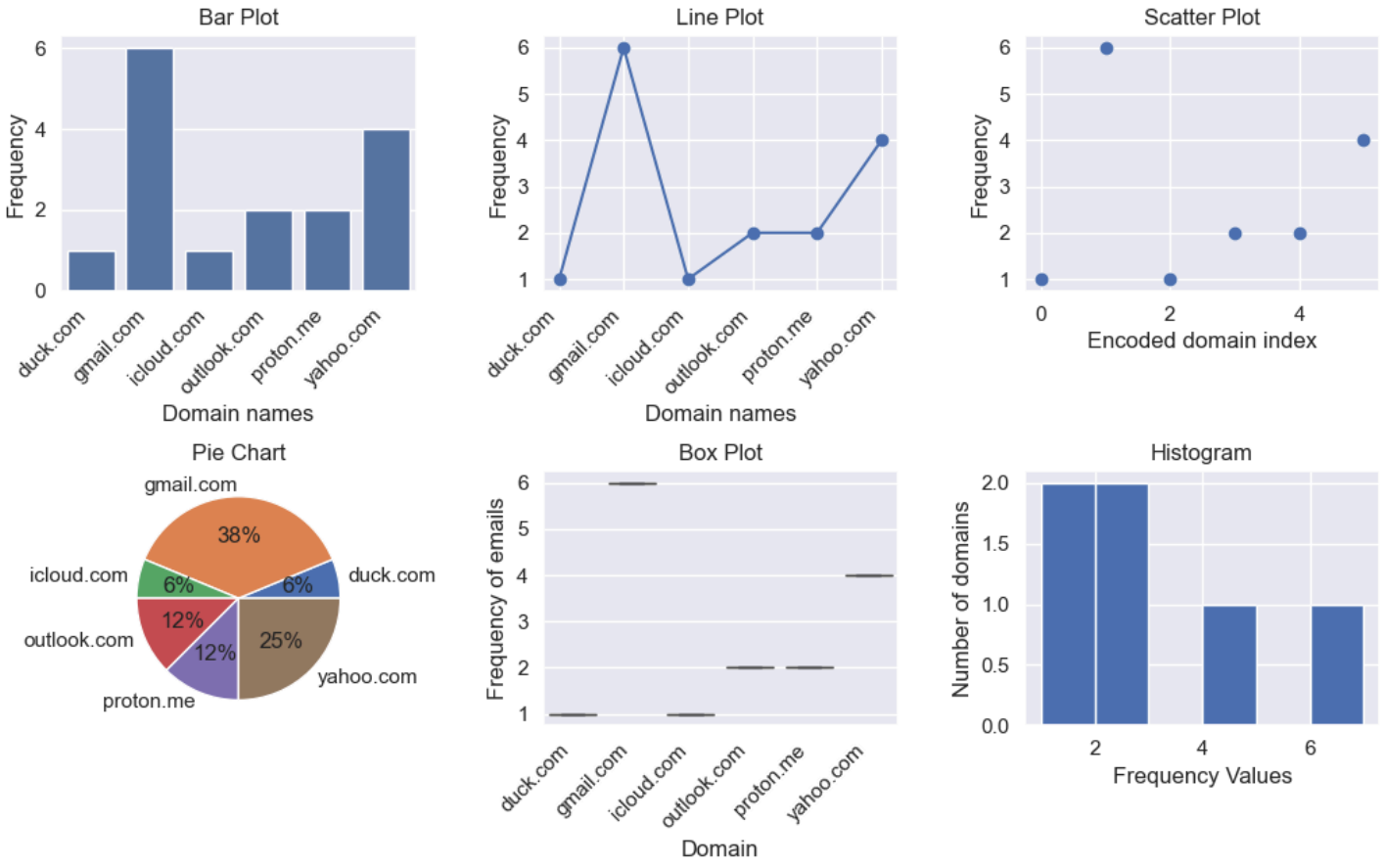
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import random
sns.set()
plt.rcParams['figure.figsize'] = (8.5, 3.8)
plt.rcParams['figure.autolayout'] = False
plt.rcParams['savefig.bbox'] = 'tight'
plt.rcParams['savefig.pad_inches'] = 0.02
plt.rcParams['axes.titlepad'] = 6
plt.rcParams['axes.labelpad'] = 4
plt.rcParams['xtick.major.pad'] = 2
plt.rcParams['ytick.major.pad'] = 2
random.seed(42)
np.random.seed(42)
```

1. Write a Python program that slices a list of email addresses to extract `usernames` and `domains`. Then, analyze and visualize the frequency of email domains using the following plots with `matplotlib` and `seaborn`:

a. **Bar Plot**
   X-axis: `Domain names` (e.g., gmail.com, yahoo.com)
   Y-axis: `Frequency` (number of times each domain appears)
b. **Line Plot**
   X-axis: `Domain names`
   Y-axis: `Frequency`
c. **Scatter Plot**
   X-axis: `Encoded domain index` (e.g., 0 for gmail.com, ...)
   Y-axis: `Frequency`
d. **Pie Chart**
   Slices: `Domain names`
   Size of each slice: `Frequency`
e. **Box Plot**
   X-axis: `Domain`
   Y-axis: `Frequency of emails`
f. **Histogram**
   X-axis: `Frequency Values`
   Y-axis: `Number of domains with those frequencies`

```
emails = [
    "alice@gmail.com","bob@yahoo.com","carol@gmail.com","dave@outlook.com",
    "erin@yahoo.com","frank@gmail.com","grace@proton.me","heidi@outlook.com",
    "ivan@gmail.com","judy@yahoo.com","mallory@icloud.com","oscar@gmail.com",
    "peggy@duck.com","trent@yahoo.com","victor@proton.me","walter@gmail.com"
]
usernames = [e.split('@')[0] for e in emails]
domains = [e.split('@')[1] for e in emails]
counts = pd.Series(domains).value_counts().sort_index()
idx = np.arange(len(counts))

fig, axes = plt.subplots(2, 3, figsize=(10.5, 6.5))
ax = axes[0,0]
sns.barplot(x=counts.index, y=counts.values, color='C0', ax=ax)
ax.set_xticks(np.arange(len(counts.index)))
ax.set_xticklabels(list(counts.index), rotation=45, ha='right')
```

```
ax.set_xlabel('Domain names'); ax.set_ylabel('Frequency'); ax.set_title('Bar Plot')

ax = axes[0,1]
ax.plot(idx, counts.values, marker='o')
ax.set_xticks(idx)
ax.set_xticklabels(list(counts.index), rotation=45, ha='right')
ax.set_xlabel('Domain names'); ax.set_ylabel('Frequency'); ax.set_title('Line Plot')

ax = axes[0,2]
ax.scatter(idx, counts.values)
ax.set_xlabel('Encoded domain index'); ax.set_ylabel('Frequency'); ax.set_title('Scatter Plot')

ax = axes[1,0]
ax.pie(counts.values, labels=counts.index, autopct='%1.0f%%')
ax.set_title('Pie Chart')

ax = axes[1,1]
sns.boxplot(x=counts.index, y=counts.values, ax=ax)
ax.set_xticks(np.arange(len(counts.index)))
ax.set_xticklabels(list(counts.index), rotation=45, ha='right')
ax.set_xlabel('Domain'); ax.set_ylabel('Frequency of emails'); ax.set_title('Box Plot')

ax = axes[1,2]
ax.hist(counts.values, bins=range(1, counts.max()+2))
ax.set_xlabel('Frequency Values'); ax.set_ylabel('Number of domains'); ax.set_title('Histogram')

plt.tight_layout(pad=0.6, w_pad=0.6, h_pad=0.8)
```



2. Write a Python program to print student database details based on Admission Number.

Display the following fields:

- `name`
- `roll no`

- `class`
- `percentage`

Input: `Admission Number`

Output: Print the student's details if found, otherwise print "Not found".

```
db = {
    101: {"name":"Aarav","roll":1,"class":"XI-A","percentage":89.2},
    102: {"name":"Isha","roll":2,"class":"XI-A","percentage":92.5},
    103: {"name":"Kabir","roll":3,"class":"XI-B","percentage":76.8},
    104: {"name":"Meera","roll":4,"class":"XI-B","percentage":84.1}
}
adm = 102
rec = db.get(adm)
if rec:
    print("Admission Number:", adm)
    print("name:", rec["name"])
    print("roll no:", rec["roll"])
    print("class:", rec["class"])
    print("percentage:", rec["percentage"])
else:
    print("Not found")
```

```
Admission Number: 102
name: Isha
roll no: 2
class: XI-A
percentage: 92.5
```

3. Write a Python program to analyze climate using `matplotlib` and `seaborn`.

a. **Line Plot**

Plot the temperature trend over months.

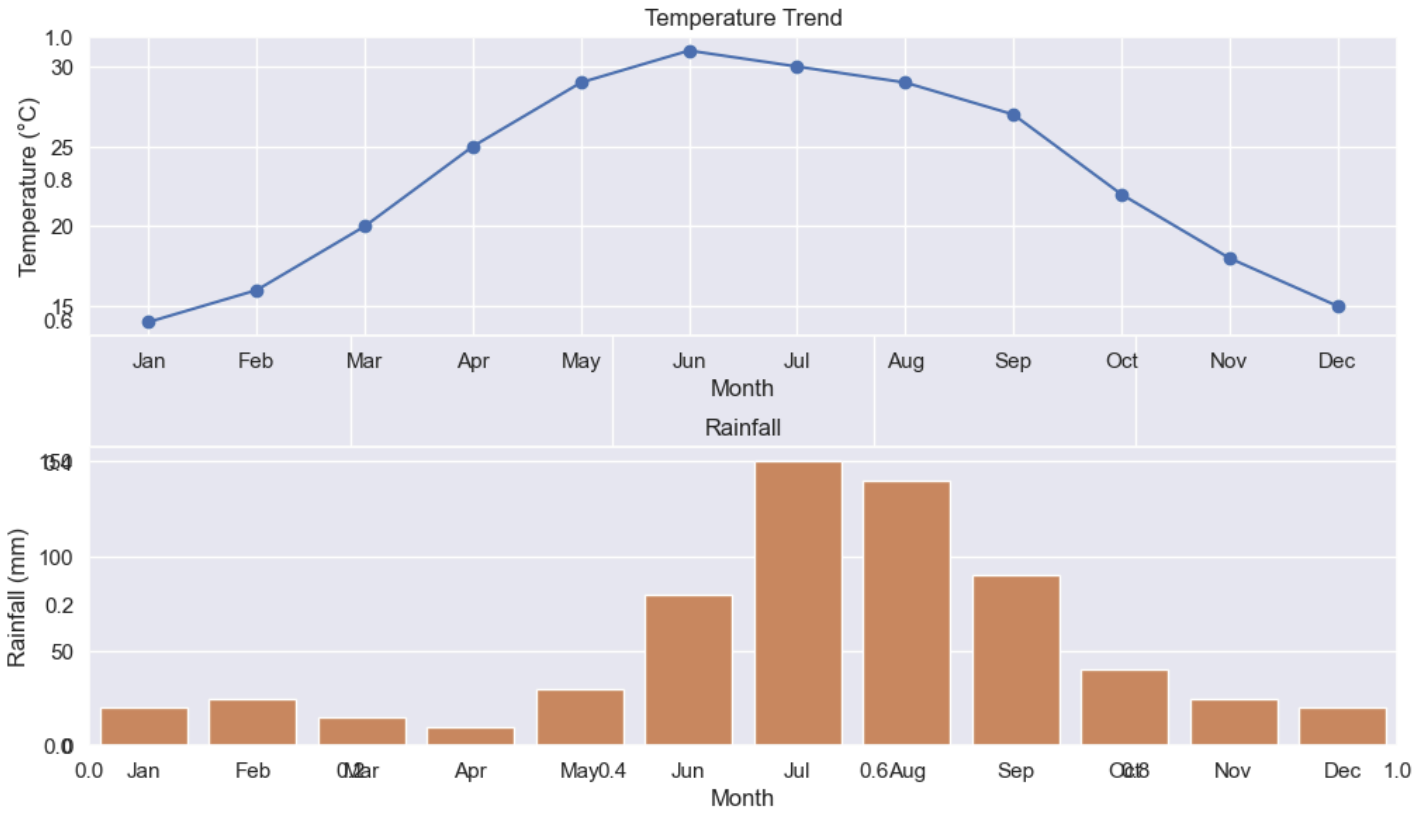
b. **Bar Plot**

Visualize rainfall for each month.

c. **Combined Chart**

Display both temperature and rainfall together for comparison.

```
months = ["Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"]
temp = [14,16,20,25,29,31,30,29,27,22,18,15]
rain = [20,25,15,10,30,80,150,140,90,40,25,20]
fig = plt.subplots(figsize=(10.5, 6.2))[0]
ax1 = plt.subplot(2,1,1)
ax1.plot(months, temp, marker='o')
ax1.set_xlabel('Month'); ax1.set_ylabel('Temperature (°C)'); ax1.set_title('Temperature Trend')
ax2 = plt.subplot(2,1,2)
sns.barplot(x=months, y=rain, color='C1', ax=ax2)
ax2.set_xlabel('Month'); ax2.set_ylabel('Rainfall (mm)'); ax2.set_title('Rainfall')
plt.tight_layout(pad=0.6, h_pad=0.8)
```

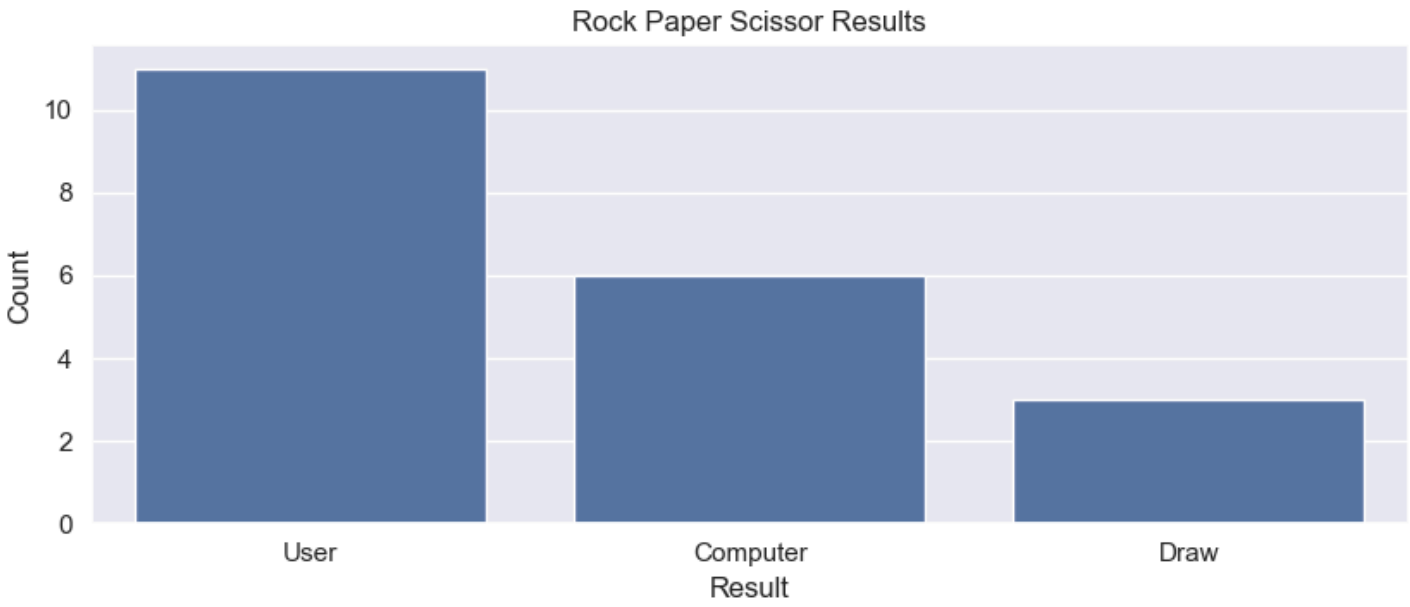


4. Write a Python program to create a **Rock, Paper, Scissor** game and visualize the results using `matplotlib`.

Requirements:

- Simulate multiple rounds of the game between a user and computer.
- For each round, randomly select moves for both user and computer from `['Rock', 'Paper', 'Scissor']`.
- Determine the winner for each round.
- Count the number of wins for `User`, `Computer`, and `Draw`.
- Visualize the results using a **bar plot** with `matplotlib` and `seaborn`.

```
choices = ['Rock','Paper','Scissor']
rounds = 20
user_moves = [random.choice(choices) for _ in range(rounds)]
comp_moves = [random.choice(choices) for _ in range(rounds)]
res = []
for u,c in zip(user_moves, comp_moves):
    if u == c:
        res.append('Draw')
    elif (u=='Rock' and c=='Scissor') or (u=='Paper' and c=='Rock') or (u=='Scissor' and c=='Paper'):
        res.append('User')
    else:
        res.append('Computer')
summary = pd.Series(res).value_counts().reindex(['User','Computer','Draw']).fillna(0)
plt.figure(); sns.barplot(x=summary.index, y=summary.values, color='C0'); plt.xlabel('Result');
plt.ylabel('Count'); plt.title('Rock Paper Scissor Results'); plt.tight_layout()
```



5. Write a Python program to analyze and visualize daily screen time usage using `Seaborn` and `Matplotlib`.

The program should display:

a. **Line Plot**
   Show the number of hours spent on screens each day over a week.

b. **Bar Plot**
   Compare screen time across different days.

c. **Histogram**
   Show the frequency distribution of screen time.

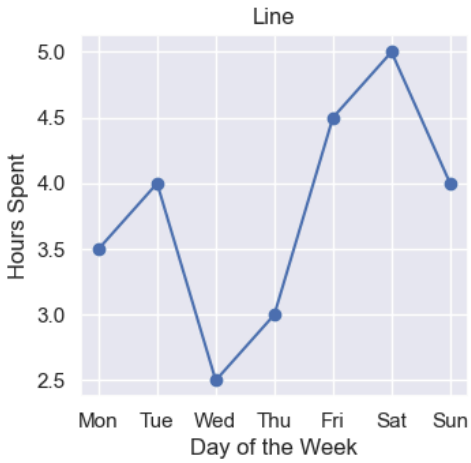
d. **Box Plot**
   Observe screen time variations and detect outliers.

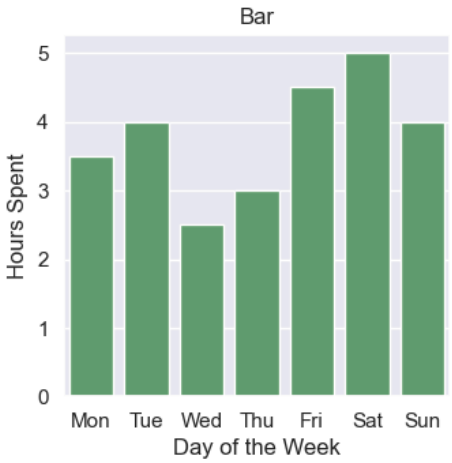
e. **Pie Chart** (using `Matplotlib`)
   Show the proportion of total weekly screen time spent on each day.

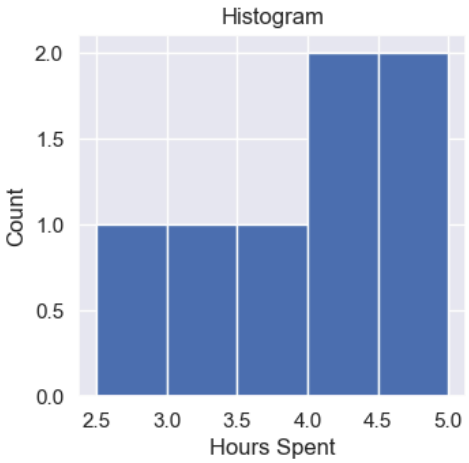
Label the X-axis as `Day of the Week` and the Y-axis as `Hours Spent`.

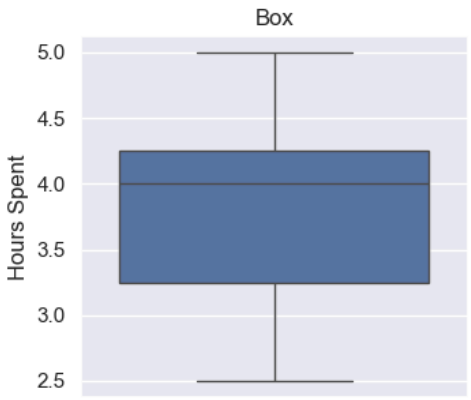
Ensure your graphs are clearly titled and easy to interpret.

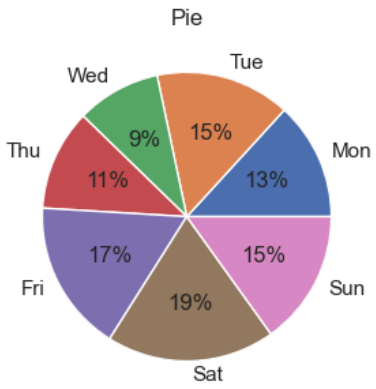
```
days = ['Mon','Tue','Wed','Thu','Fri','Sat','Sun']
hours = [3.5, 4, 2.5, 3, 4.5, 5, 4]
fig, axes = plt.subplots(2, 3, figsize=(10.5, 6.5))
ax = axes[0,0]; ax.plot(days, hours, marker='o'); ax.set_xlabel('Day of the Week');
ax.set_ylabel('Hours Spent'); ax.set_title('Line')
ax = axes[0,1]; sns.barplot(x=days, y=hours, color='C2', ax=ax); ax.set_xlabel('Day of the Week');
ax.set_ylabel('Hours Spent'); ax.set_title('Bar')
ax = axes[0,2]; ax.hist(hours, bins=5); ax.set_xlabel('Hours Spent'); ax.set_ylabel('Count');
ax.set_title('Histogram')
ax = axes[1,0]; sns.boxplot(y=hours, ax=ax); ax.set_ylabel('Hours Spent'); ax.set_title('Box')
ax = axes[1,1]; ax.pie(hours, labels=days, autopct='%1.0f%%'); ax.set_title('Pie')
axes[1,2].axis('off')
plt.tight_layout(pad=0.6, w_pad=0.6, h_pad=0.8)
```


Line
Hours Spent
5.0
4.5
4.0
3.5
3.0
2.5
Mon
Tue
Wed
Thu
Fri
Sat
Sun
Day of the Week


Bar
Hours Spent
5
4
3
2
1
0
Mon
Tue
Wed
Thu
Fri
Sat
Sun
Day of the Week


Histogram
Count
2.0
1.5
1.0
0.5
0.0
2.5
3.0
3.5
4.0
4.5
5.0
Hours Spent


Box
Hours Spent
5.0
4.5
4.0
3.5
3.0
2.5


Pie
Tue
Wed
Mon
Thu
Fri
Sun
Sat
15%
9%
13%
11%
17%
15%
19%