2330016056

October 6, 2025

1 Lab 1 Essential Libraries and Tools

Task 1: In the code cell below, complete the function `insertSecond` which takes two arguments, array `a` and element `b`, and returns an array which `b` is inserted before the second element in `a`.

[9]:

```python
# You should return your result.
import numpy as np


def insertSecond(a, b):
    res = a.tolist()
    res.insert(1, b)
    return np.array(res)


# Test cases
assert np.array_equal(insertSecond(
    np.array([-5, -10, -12, -6]), 5), np.array([-5, 5, -10, -12, -6]))
assert np.array_equal(insertSecond(
    np.array([1, 2, 3]), 7), np.array([1, 7, 2, 3]))
assert np.array_equal(insertSecond(
    np.array([-5, -10, -12, -6]), 8), np.array([-5, 8, -10, -12, -6]))
assert np.array_equal(insertSecond(
    np.array([1, 2, 3]), 12), np.array([1, 12, 2, 3]))
```

Task 2: In the code cell below, complete the function `mergeArrays` takes two array arguments `a` and `b`. The two arrays are merged, in which process the duplicate elements are removed. Finally the merged array are sorted.

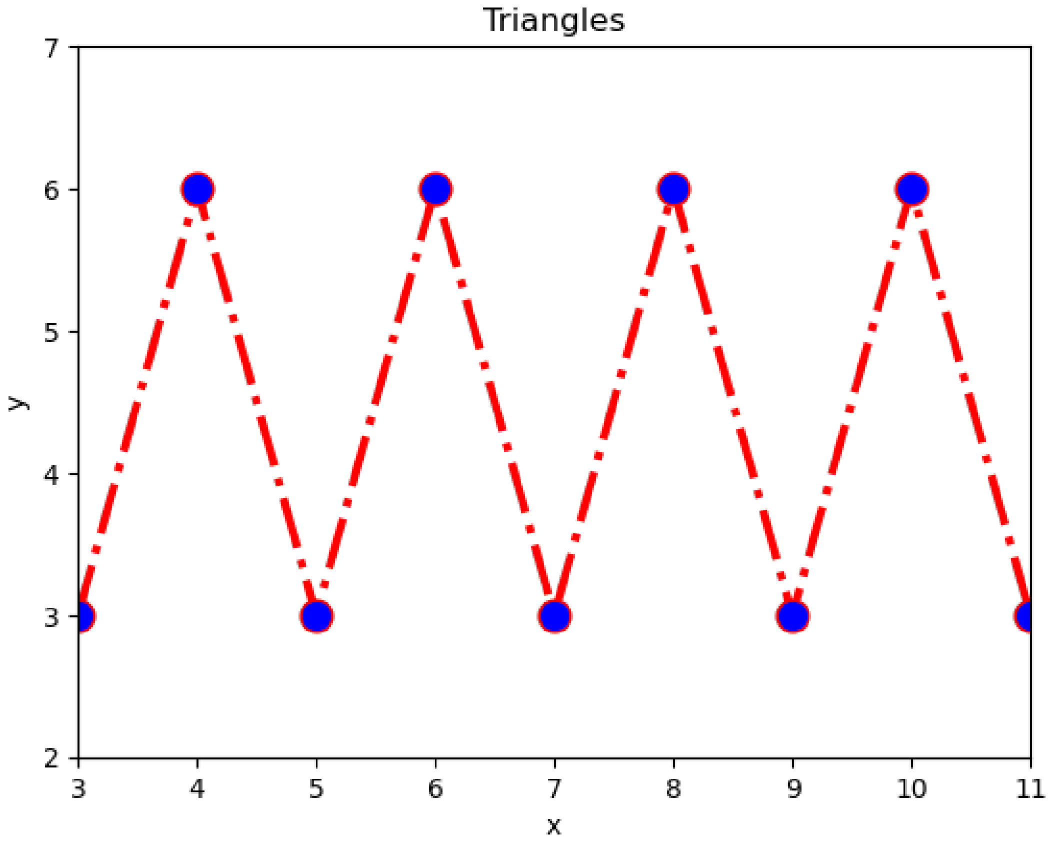
[8]:

```python
import numpy as np


def mergeArrays(a, b):
    res = np.concatenate((a, b))
    res = np.unique(res)
    return res
```

```
# Test cases
assert np.array_equal(mergeArrays(
    np.array([1, 1, 4, 8, 1]), np.array([2, 3])), np.array([1, 2, 3, 4, 8]))
assert np.array_equal(mergeArrays(np.array(
    [-5, -10, -10, -6]), np.array([-5, 8, -10, -12, -6])), np.array([-12, -10,␣
 ↪-6, -5, 8]))
assert np.array_equal(mergeArrays(
    np.array([1, 1, 6, 8, 1]), np.array([2, 3])), np.array([1, 2, 3, 6, 8]))
```

[1]:

```
import matplotlib.pyplot as plt

# x axis values
x = [3, 4, 5, 6, 7, 8, 9, 10, 11]

# y axis values
y = [3, 6, 3, 6, 3, 6, 3, 6, 3]

# plotting the points
plt.plot(
    x, y,
    color='red',           # line color
    linestyle='dashdot',   # line style
    linewidth=3,           # line thickness
    marker='o',            # circular markers
    markerfacecolor='blue',   # marker fill color
    markersize=12          # marker size
)

# Set the y-limits of the current axes
plt.ylim(2, 7)

# Set the x-limits of the current axes
plt.xlim(3, 11)

# naming the x axis
plt.xlabel('x')

# naming the y axis
plt.ylabel('y')

# giving a title to the graph
plt.title('Triangles')

# display the plot
plt.show()
```

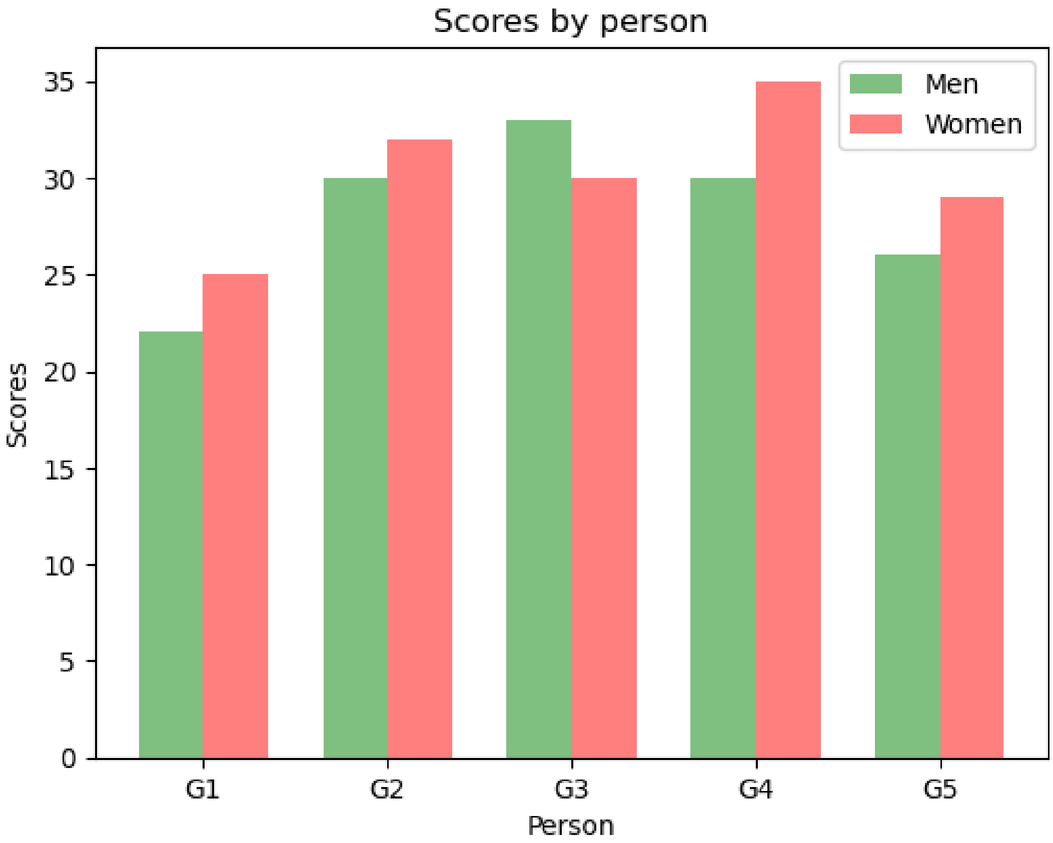


Task 3: Complete the code below to create a image which looks exactly the same as the follow one:

[3]:

```
import numpy as np
import matplotlib.pyplot as plt

n_groups = 5
men_means = (22, 30, 33, 30, 26)
women_means = (25, 32, 30, 35, 29)
alpha = 0.5

fig, ax = plt.subplots()

index = np.arange(n_groups)
bar_width = 0.35

rects1 = plt.bar(index, men_means, bar_width,
                 alpha=alpha, color='g', label='Men')
rects2 = plt.bar(index + bar_width, women_means, bar_width,
                 alpha=alpha, color='r', label='Women')
```

```python
plt.xlabel('Person')
plt.ylabel('Scores')
plt.title('Scores by person')
plt.xticks(index + bar_width / 2, ('G1', 'G2', 'G3', 'G4', 'G5'))
plt.legend()

plt.show()
```



Task 4: In the code cell below, complete the function `setDataFrameZeros` which takes a dataFrame `df` as an arguement, and returns a new dataFrame. The label and index in the new dataFrame is the same as those in `df` . If an element is 0 in `df` , set its entire row and column to 0 in the new dataFrame. All other elements are the same as those in `df` .

[7]:

```python
import pandas as pd


def setDataFrameZeros(df):
    res = df.copy()
    res.loc[df.index[df.eq(0).any(axis=1)], :] = 0
    res.loc[:, df.columns[df.eq(0).any(axis=0)]] = 0
    return res


# Test cases
df1 = pd.DataFrame({'c1': [1, 4, 7], 'c2': [2, 0, 8], 'c3': [3, 6, 9]})
df2 = pd.DataFrame({'c1': [1, 0, 7], 'c2': [0, 0, 0], 'c3': [3, 0, 9]})
assert (df2.equals(setDataFrameZeros(df1)))

df1 = pd.DataFrame({'c1': [0, 3, 1], 'c2': [1, 4, 3],
                    'c3': [2, 5, 1], 'c4': [0, 2, 5]})
df2 = pd.DataFrame({'c1': [0, 0, 0], 'c2': [0, 4, 3],
                    'c3': [0, 5, 1], 'c4': [0, 0, 0]})
assert (df2.equals(setDataFrameZeros(df1)))
```

[ ]: