Билет 7

1. Как распределить категориальные данные в Seaborn. Какой функцией можно объединить графики?

   График разброса по категориям удобен для небольших наборов данных, так как по мере увеличения количества точек они все равно начнут перекрывать друг друга и сливаться. Что бы преодолеть эти трудности, лучше воспользоваться графиками, которые сами содержат некоторую информацию о распределении внутри категорий. Один из таких графиков - это "ящик с усами" или boxplot. Его можно построить с помощью той же функции catplot с параметром kind, установленным в значение 'box'.

   Чтобы объединить несколько кривых на одном графике, просто перечислите их в функции plot(), при этом для каждой из них можно задать собственные парамеры кривых с помощью соответствующих "кодов". Следующий пример не только отрисовывает два графика в одном окне, но также делает подписи осей и отображает заголовок.

   ```
   plt.plot(X,Y, X,np.cos(X))       # объединение 2-х графиков
   plt.legend(('sin','cos'))           # подписи
   plt.title('Trigonometry')        # заголовок
   plt.xlabel('Time, s')               # наименование оси абсцисс
   plt.ylabel('Amplitude, c.u.')  # наименование оси ординат
   plt.show()
   ```

2. Если воспользоваться классами для управления сеткой ячеек, в которой располагаются графики, то можно...

   получить доступ к более тонким настройкам всей композиции графиков.

Практика

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_theme()


class DataAnalysis:
    '''
    Постройте простой график на основе встроенного набора данных flights,
    который содержит информацию о месячном объеме пассажироперевозок в период с
1945 по 1960 год.
    Строки – сколько пассажиров было перевезено в определенном месяце
определенного года,
    столбцы – год, месяц и количество.
```

```
    '''

    def __init__(self, filename):
        self._filename = filename
        self._flights = sns.load_dataset("flights")

    def buildGraph(self):
        sns.relplot(
            data=self._flights,
            x='year',
            y='passengers',
            kind='line'
            )
        plt.savefig(self._filename)
        plt.show(block=True)

    @property
    def data(self):
        return self._data

task = DataAnalysis('ticket7.png')
task.buildGraph()
```