Matplotlib

Визуализация данных является ключевым навыком для будущих специалистов по анализу данных. `Matplotlib` упрощает создание содержательных и информативных графиков. В этом уроке вы узнаете, как создавать различные типы графиков и настраивать их для более привлекательного и понятного представления информации.

Линейный график

С помощью библиотеки `matplotlib` в `Python` можно создавать различные виды графиков. Самый простой из них — линейный график. Здесь представлен общий шаблон для его создания.

```
import pandas as pd

gapminder = pd.read_csv('datasets/gapminder_1.csv')
gapminder.columns
```

```
Index(['country', 'continent', 'year', 'life_exp', 'population',
'gdp_cap'], dtype='object')
```

```
print(gapminder)
```

```
          country continent  year  life_exp  population     gdp_cap
0     Afghanistan      Asia  1952    28.801     8425333  779.445314
1     Afghanistan      Asia  1957    30.332     9240934  820.853030
2     Afghanistan      Asia  1962    31.997    10267083  853.100710
3     Afghanistan      Asia  1967    34.020    11537966  836.197138
4     Afghanistan      Asia  1972    36.088    13079460  739.981106
...           ...       ...   ...       ...         ...         ...
1699     Zimbabwe    Africa  1987    62.351     9216418  706.157306
1700     Zimbabwe    Africa  1992    60.377    10704340  693.420786
1701     Zimbabwe    Africa  1997    46.809    11404948  792.449960
1702     Zimbabwe    Africa  2002    39.989    11926563  672.038623
1703     Zimbabwe    Africa  2007    43.487    12311143  469.709298

[1704 rows x 6 columns]
```

Инструкции

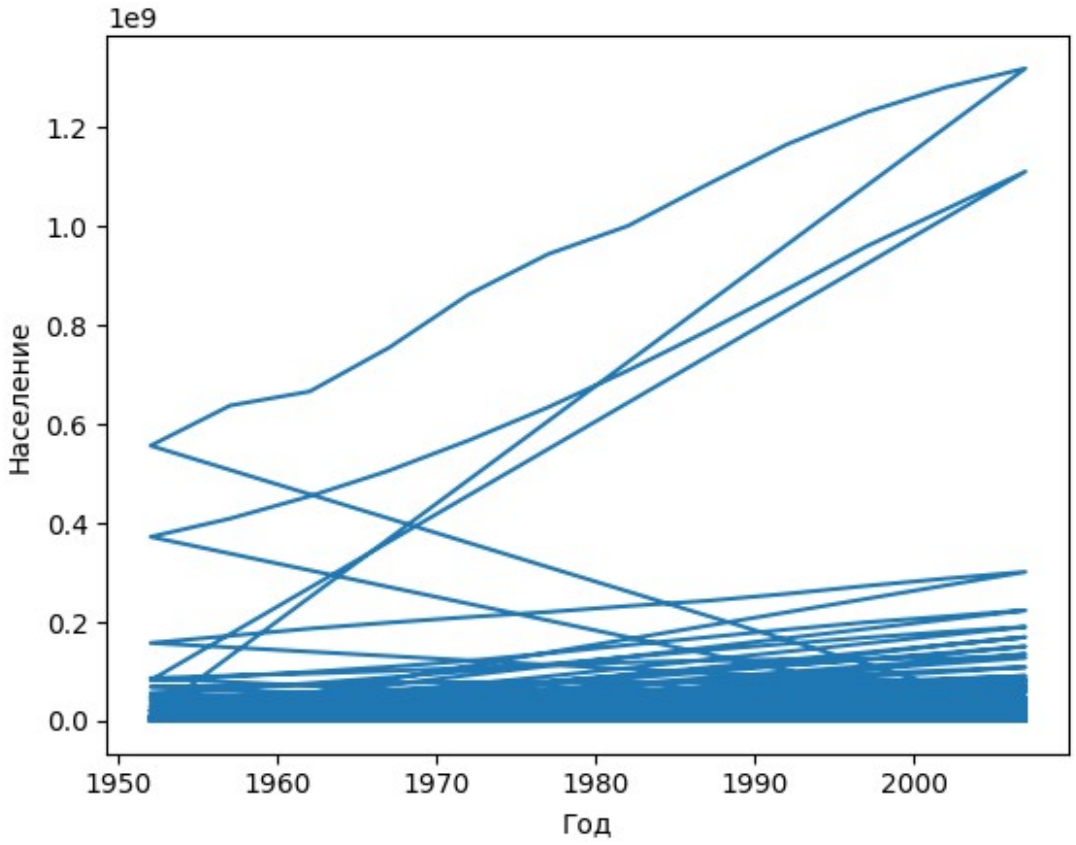
- Вывидите на экран последний элемент из списка `year` и списка `pop`, чтобы увидеть, какова прогнозируемая численность населения к 2100 году. Используйте две функции `print()`.
- Прежде чем начать, вам следует импортировать `matplotlib.pyplot as plt`. `pyplot` — это подпакет `matplotlib`, поэтому здесь используется точка.

- Используйте `plt.plot()`, чтобы создать линейный график. Год должен отображаться на горизонтальной оси, население — на вертикальной. Не забудьте завершить с `plt.show()` для отображения графика.

```
print(gapminder['year'].iloc[-1])  # Доступ к последнему элементу столбца 'year'

print(gapminder['population'].iloc[len(gapminder) - 1]) # Доступ к последнему элементу столбца 'population'
```

```
2007
12311143
```

```
# Импорт библиотеки matplotlib.pyplot как plt
import matplotlib.pyplot as plt

# Создание линейного графика: годы по оси x, население по оси y
plt.plot(gapminder['year'], gapminder['population'])

# Подписи осей
plt.xlabel('Год')
plt.ylabel('Население')

# Отображение графика с помощью plt.show()
plt.show()
```



```
# Фильтрация данных для Афганистана
afghanistan = gapminder[gapminder['country'] == 'Afghanistan']

# Создание линейного графика: годы по оси x, население по оси y для Афганистана
plt.plot(afghanistan['year'], afghanistan['population'])

# Подписи осей и заголовок
plt.xlabel('Год')
plt.ylabel('Население')
plt.title('Рост населения в Афганистане')

# Отображение графика
plt.show()
```

Линейный график (2)

Теперь, когда вы создали свой первый линейный график, давайте начнем работать с данными, которые использовал профессор Ханс Рослинг для создания своей красивой диаграммы-пузыря. Они были собраны в 2007 году. У вас есть два списка:

- `life_exp`, который содержит продолжительность жизни для каждой страны, и `gdp_cap`, который содержит ВВП на душу населения (т.е. на человека) для каждой

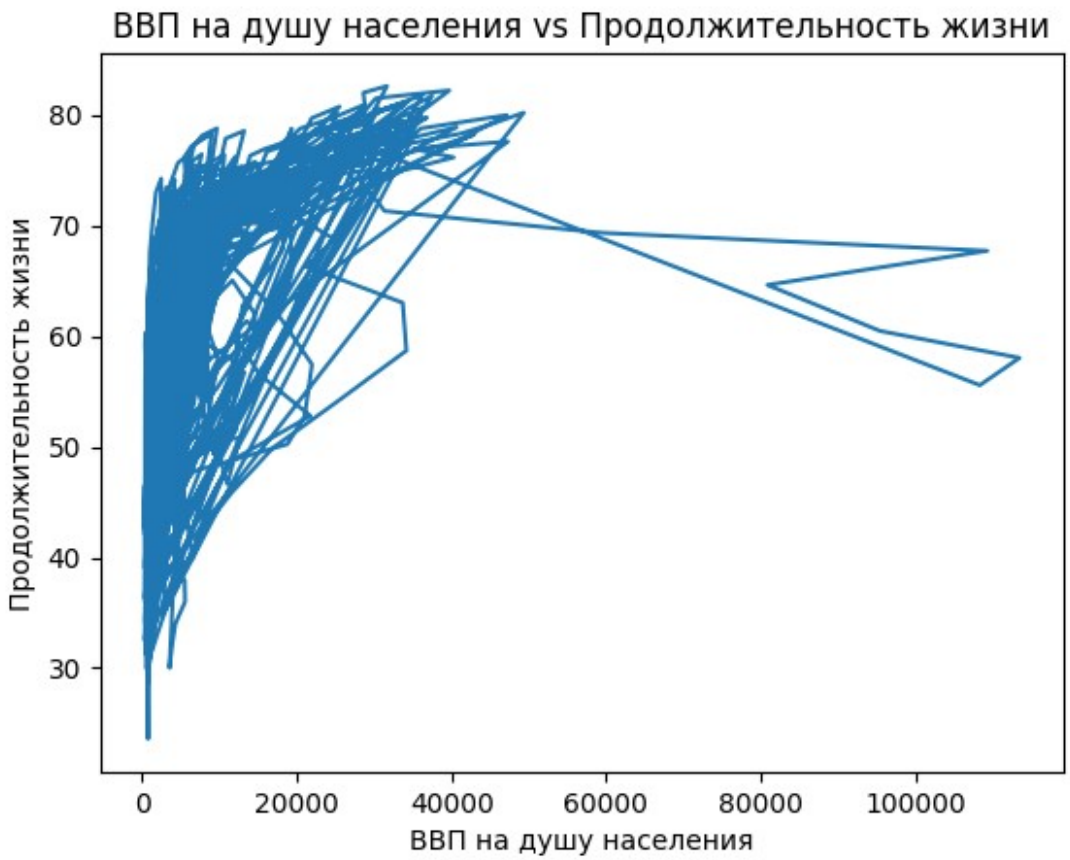
страны, выраженный в долларах США. ВВП означает валовой внутренний продукт. По сути, это представляет размер экономики страны. Разделите это на население, и вы получите ВВП на душу населения. `matplotlib.pyplot` уже импортирован как `plt`, так что вы можете начать работать прямо сейчас.

Инструкции

- Выведите последний элемент из списка `gdp_cap` и список `life_exp`; это информация о Зимбабве.
- Постройте линейный график с `gdp_cap` на оси x и `life_exp` на оси `y`. Может ли эти данные имеют смысл в представлении на линейном графике?
- Не забудьте завершить командой `plt.show()`, чтобы фактически отобразить график.

```
# Выведите последний элемент из столбцов gdp_cap и life_exp
print(gapminder['gdp_cap'].iloc[-1])
print(gapminder['life_exp'].iloc[-1])

# Постройте линейный график, gdp_cap на оси x, life_exp на оси y
plt.plot(gapminder['gdp_cap'], gapminder['life_exp'])

# Подписи осей
plt.xlabel('ВВП на душу населения')
plt.ylabel('Продолжительность жизни')
plt.title('ВВП на душу населения vs Продолжительность жизни')

# Отобразить график
plt.show()
```

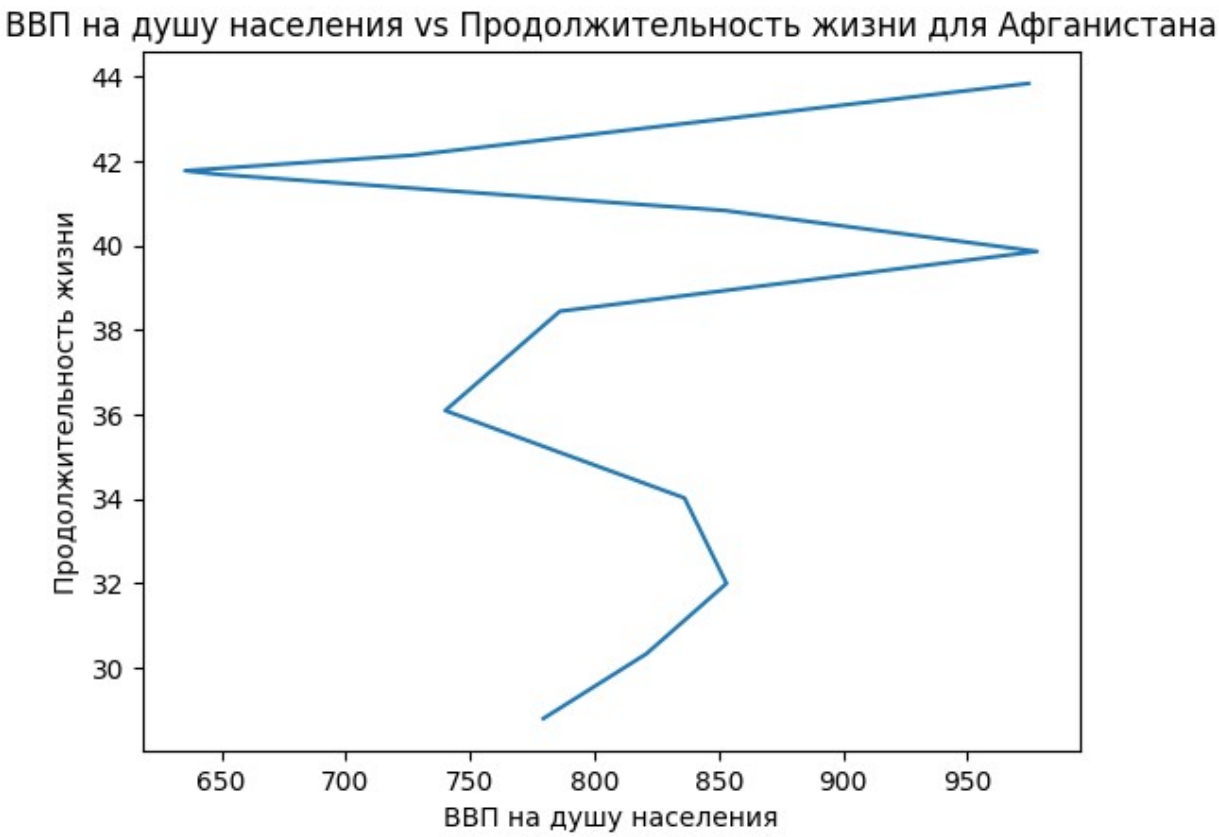
```
469.7092981
43.487
```



- Информация о Афганистане.

```
# Отфильтровать данные для Афганистана
afghanistan = gapminder[gapminder['country'] == 'Afghanistan']

# Вывести последний элемент gdpPercap и lifeExp для Афганистана
print(afghanistan['gdp_cap'].iloc[-1])
print(afghanistan['life_exp'].iloc[-1])

# Построить линейный график: ВВП на душу населения по оси x,
продолжительность жизни по оси y для Афганистана
plt.plot(afghanistan['gdp_cap'], afghanistan['life_exp'])

# Подписи осей
plt.xlabel('ВВП на душу населения')
plt.ylabel('Продолжительность жизни')
plt.title('ВВП на душу населения vs Продолжительность жизни для
Афганистана')

# Отобразить график
plt.show()
```

```
974.5803384
43.828
```



Точечная диаграмма (1)

Когда у вас есть временная шкала по горизонтальной оси, график линии - ваш друг. Но во многих других случаях, например, когда вы пытаетесь оценить наличие корреляции между двумя переменными, точечная диаграмма - более предпочтительный выбор. Ниже приведен пример построения точечной диаграммы.

```
import matplotlib.pyplot as plt
plt.scatter(x, y)
plt.show()
```

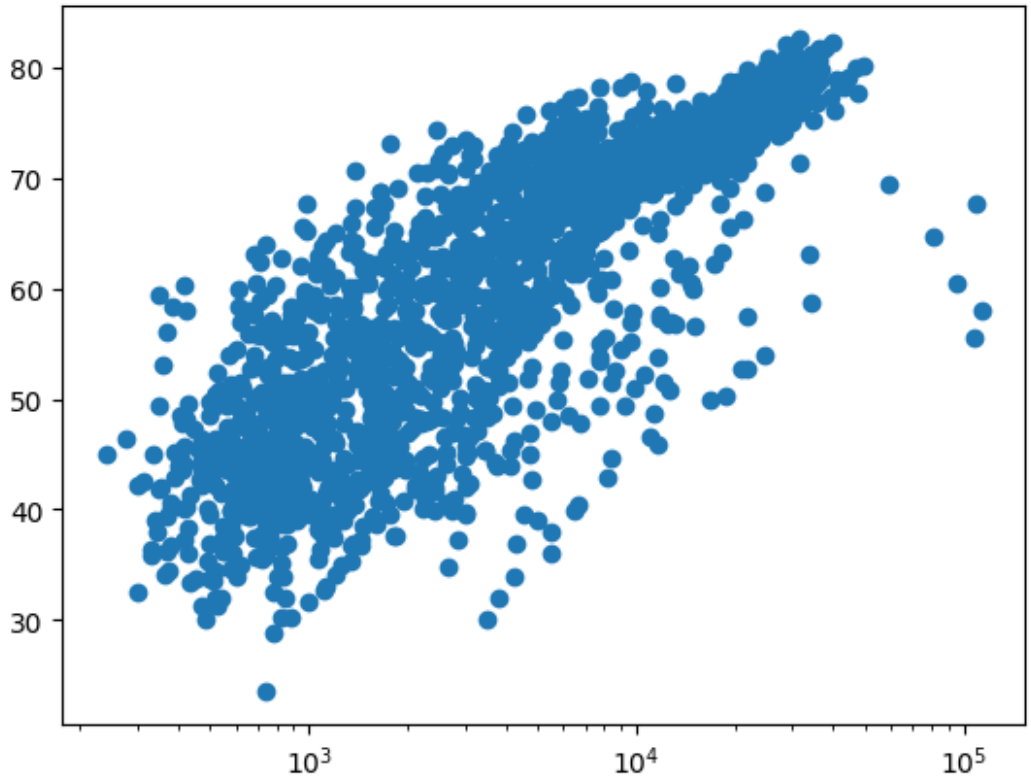
Давайте продолжим с графиком `gdp_cap` против `life_exp` - данные о ВВП и продолжительности жизни для разных стран в 2007 году. Возможно, точечная диаграмма будет лучшим вариантом?

Снова пакет `matplotlib.pyplot` доступен под именем `plt`.

Инструкции

- Измените график линии, который написан в скрипте, на точечную диаграмму.
- Корреляция станет яснее, когда вы отобразите ВВП на душу населения в логарифмической шкале. Добавьте строку `plt.xscale('log')`.
- Завершите ваш скрипт с `plt.show()`, чтобы отобразить график.

```
# Точечная диаграмма ВВП на душу населения относительно
продолжительности жизни
plt.scatter(gapminder['gdp_cap'], gapminder['life_exp'])

# Установка логарифмической шкалы для оси x
plt.xscale('log')

# Отображение графика
plt.show()
```




```
afghanistan = gapminder[gapminder['country'] == 'Afghanistan']

# Точечная диаграмма ВВП на душу населения относительно
продолжительности жизни для Афганистана
plt.scatter(afghanistan['gdp_cap'], afghanistan['life_exp'])

# Установка логарифмической шкалы для оси x
plt.xscale('log')

# Отображение графика
plt.show()
```

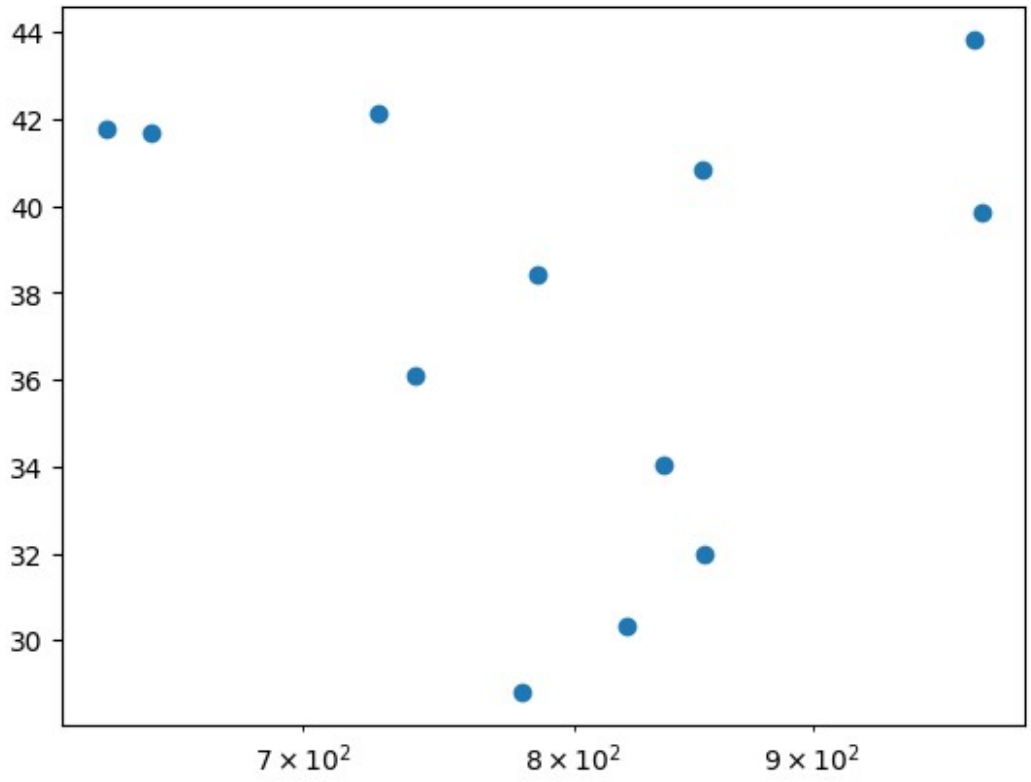


Диаграмма рассеяния (2)

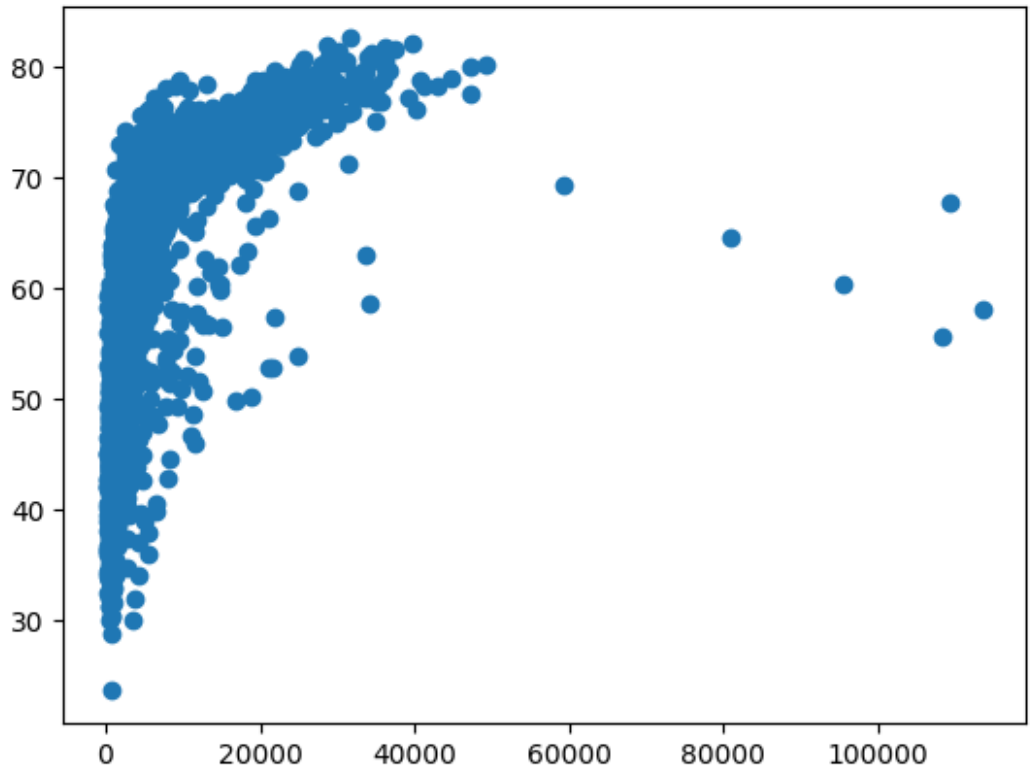
В предыдущем упражнении вы видели, что более высокий ВВП обычно соответствует более высокой продолжительности жизни. Другими словами, существует положительная корреляция.

Думаете ли вы, что есть связь между населением и продолжительностью жизни страны? Список `life_exp` из предыдущего упражнения уже доступен. Кроме того, теперь также доступен `pop`, в котором перечислены соответствующие население стран в 2007 году. Население представлено в миллионах человек.

Инструкции

- Начните с нуля: импортируйте `matplotlib.pyplot` как `plt`.
- Постройте точечную диаграмму, где `pop` отображается на горизонтальной оси, а `life_exp` - на вертикальной оси.
- Завершите скрипт с помощью `plt.show()` для отображения графика. Вы видите корреляцию?

```
# Точечная диаграмма ВВП на душу населения по отношению к ожидаемой продолжительности жизни
plt.scatter(gapminder['gdp_cap'], gapminder['life_exp'])

# Отображение графика
plt.show()
```



Построение гистограммы (1)

`life_exp`, список содержащий данные о продолжительности жизни для разных стран в 2007 году, доступен в вашем оболочке Python.

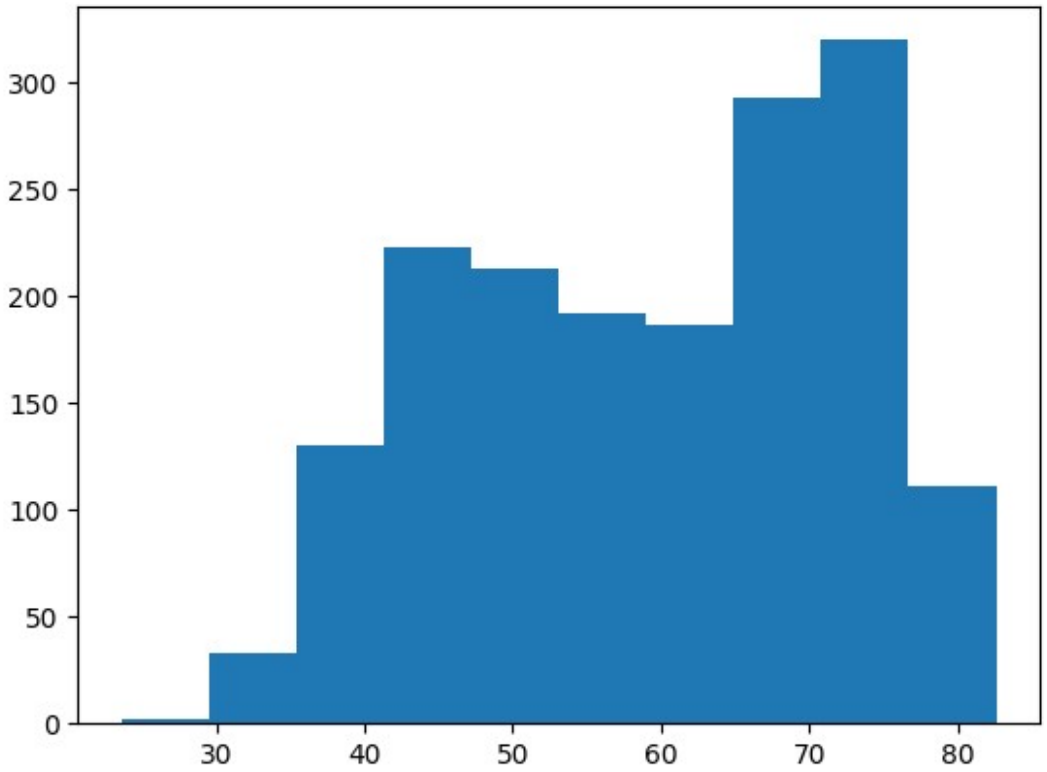
Чтобы увидеть, как распределена продолжительность жизни в разных странах, давайте создадим гистограмму `life_exp`.

`matplotlib.pyplot` уже доступен под именем `plt`.

Инструкции

- Используйте `plt.hist()`, чтобы создать гистограмму значений из `life_exp`. Не указывайте количество интервалов (бинов); Python установит количество интервалов по умолчанию равным 10.
- Добавьте `plt.show()` для отображения гистограммы. Можете ли вы определить, в каком интервале содержится больше всего наблюдений?

```
# Создание гистограммы данных по life_exp
plt.hist(gapminder['life_exp'])

# Отображение гистограммы
plt.show()
```



Построение гистограммы (1): ячейки

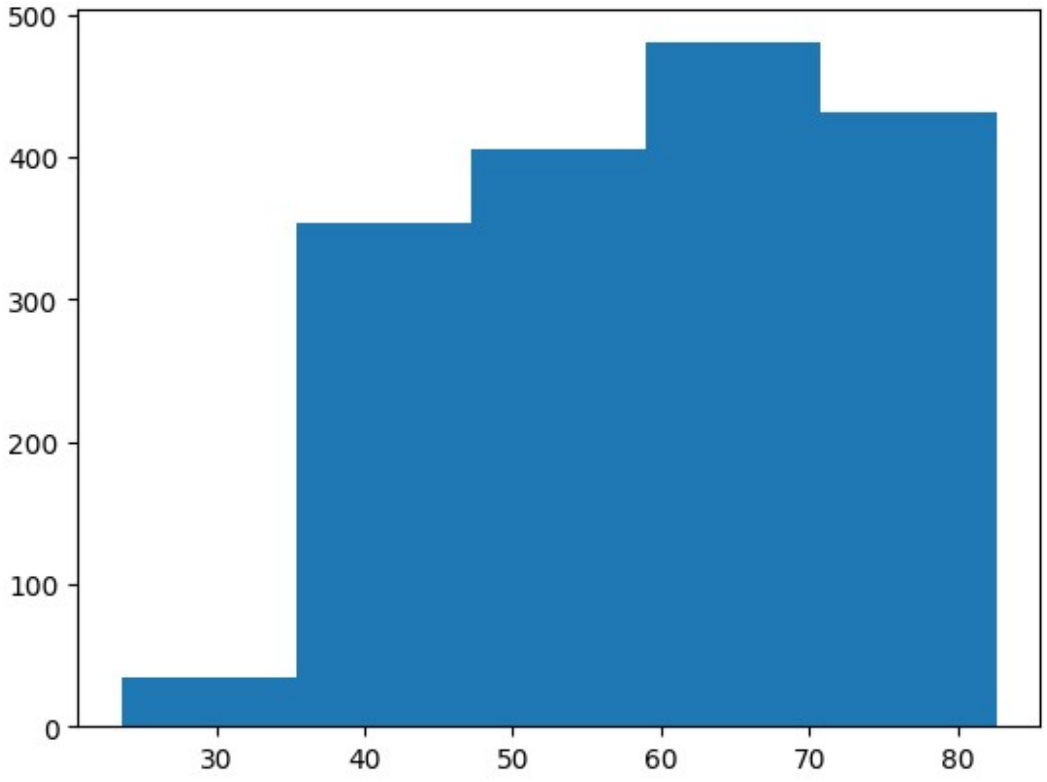
В предыдущем упражнении вы не указали количество ячеек. По умолчанию Python устанавливает количество ячейки на 10. Количество ячейки довольно важно. Слишком мало ячейки упростит реальность и не покажет вам деталей. Слишком много ячейки усложнит реальность и не покажет общую картину. Чтобы контролировать количество ячейки, на которые вы хотите разделить свои данные, вы можете установить аргумент `bins`. Именно это вы и сделаете в этом упражнении. Здесь вы будете создавать две гистограммы. Код в скрипте уже содержит вызовы `plt.show()` и `plt.clf()`; `plt.show()` отображает график; `plt.clf()` снова очищает его, чтобы вы могли начать заново. Как и раньше, `life_exp` доступен, и `matplotlib.pyplot` импортирован как `plt`.

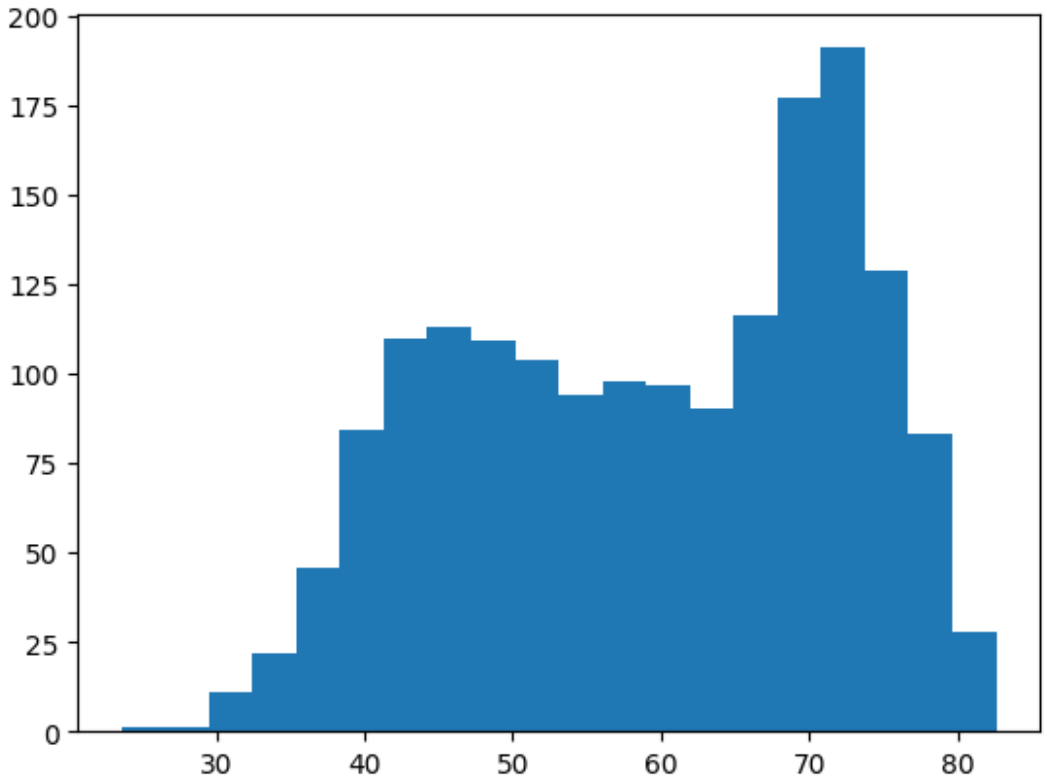
Instructions

- Постройте гистограмму `life_exp` с 5 корзинами. Можете ли вы сказать, в какой корзине содержится больше всего наблюдений?
- Постройте другую гистограмму `life_exp`, на этот раз с 20 корзинами. Лучше ли это?

```
# Построение гистограммы с 5 корзинами
plt.hist(gapminder['life_exp'], 5)

# Показать и очистить график
plt.show()
plt.clf()
```

```
# Построение гистограммы с 20 корзинами
plt.hist(gapminder['life_exp'], 20)

# Показать и очистить снова
plt.show()
plt.clf()
```






```
<Figure size 640x480 with 0 Axes>
```

Построение гистограммы (3): сравнение

В видео вы видели пирамиды населения настоящего и будущего. Потому что мы использовали гистограмму, было очень легко сделать сравнение.

Давайте сделаем похожее сравнение. `life_exp` содержит данные о продолжительности жизни для разных стран в 2007 году. У вас также есть доступ ко второму списку `life_exp1950`, содержащему аналогичные данные за 1950 год. Можете ли вы создать гистограмму для обоих наборов данных?

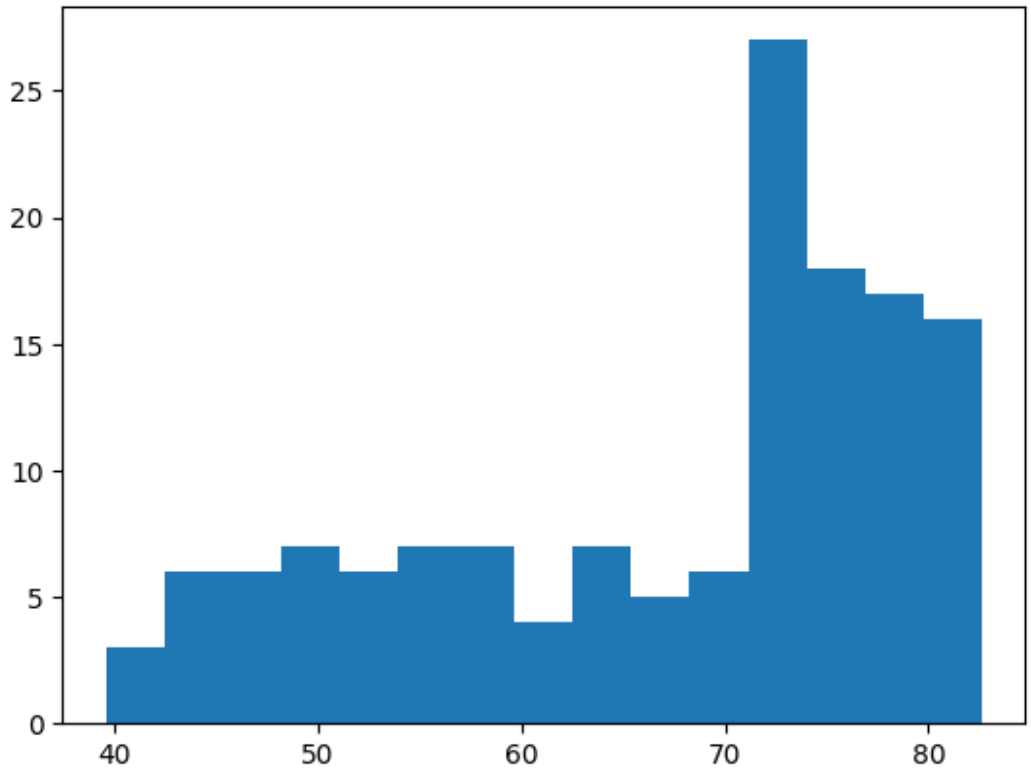
Вы снова будете создавать два графика. Команды `plt.show()` и `plt.clf()` для красивого отображения уже включены. Также `matplotlib.pyplot` импортирован для вас под именем `plt`.

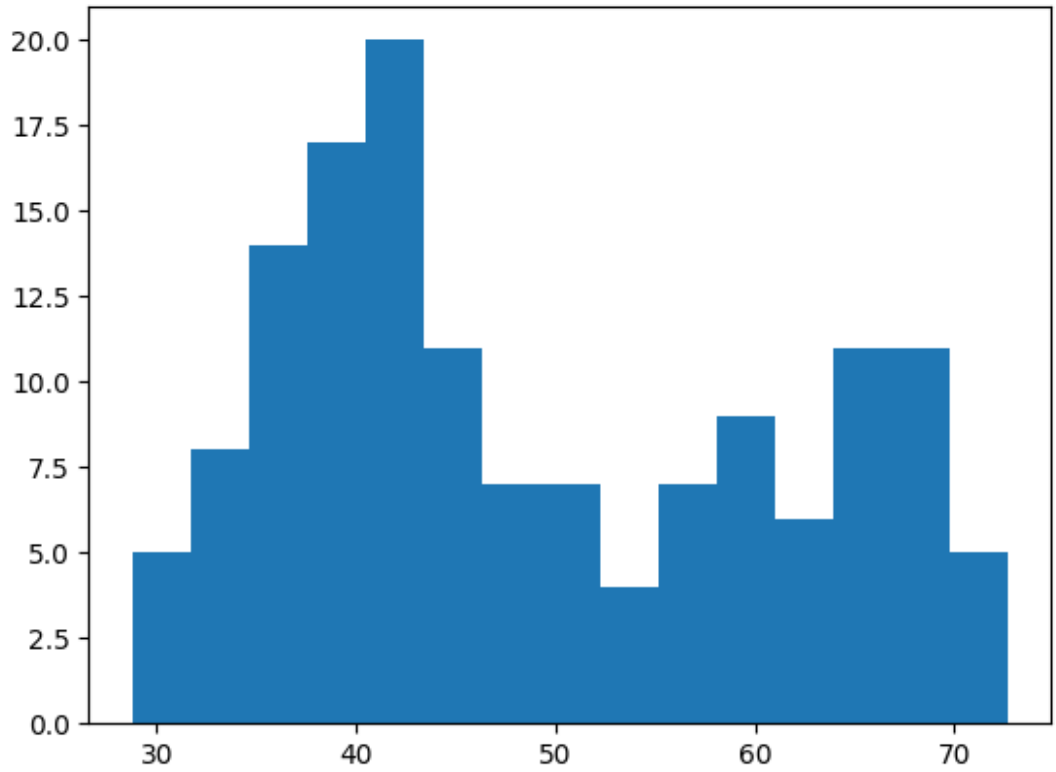
Инструкции

- Постройте гистограмму для `life_exp` с 15 корзинами.
- Постройте гистограмму для `life_exp1950` также с 15 корзинами. Есть ли большая разница с гистограммой для данных за 2007 год?

```
# Фильтрация данных для 2007 года
data_2007 = gapminder[gapminder['year'] == 2007]

# Создание гистограммы продолжительности жизни для 2007 года с 15
```

```
интервалами
plt.hist(data_2007['life_exp'], bins=15)
# Отображение и очистка графика
plt.show()
plt.clf()

# Фильтрация данных для 1950 года
data_1950 = gapminder[gapminder['year'] == 1952]

# Создание гистограммы продолжительности жизни для 2007 года с 15
интервалами
plt.hist(data_1950['life_exp'], bins=15)

# Отображение и очистка графика
plt.show()
plt.clf()
```






```
<Figure size 640x480 with 0 Axes>
```

Метки

Пришло время настроить свою собственную диаграмму. Это самая интересная часть, ты увидишь, как твоя диаграмма оживет!

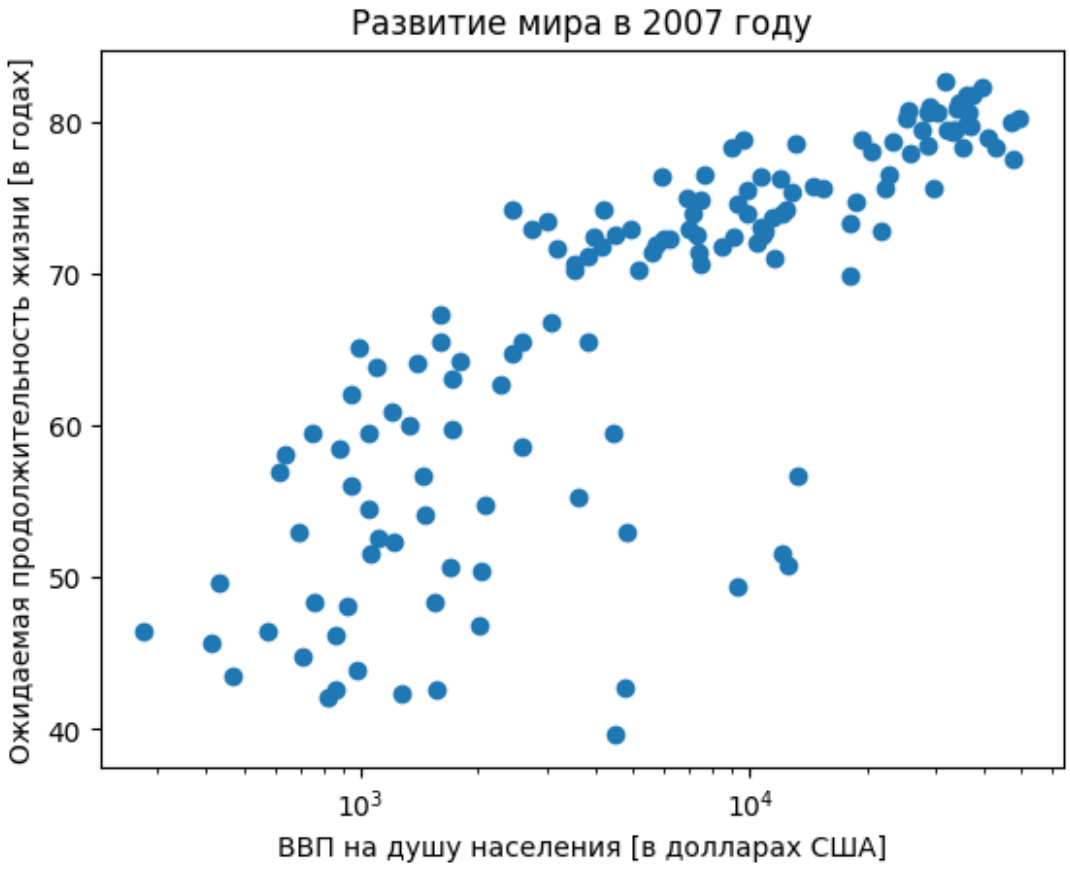
Ты собираешься работать с диаграммой рассеяния с данными о мировом развитии: ВВП на душу населения по оси `x` (логарифмическая шкала), ожидаемая продолжительность жизни по оси `y`. Код для этой диаграммы уже доступен в скрипте.

В качестве первого шага добавь подписи осей и заголовок к диаграмме. Ты можешь сделать это с помощью функций `xlabel(), ylabel() и title()`, доступных в `matplotlib.pyplot`. Этот подпакет уже импортирован как plt.

Инструкции

- Строки `xlab` и `ylab` уже предварительно заданы для тебя. Используй эти переменные, чтобы установить подписи для осей x и y соответственно.
- Строка `title` также уже задана для тебя. Используй ее для добавления заголовка к диаграмме.
- После этих настроек заверши скрипт с помощью `plt.show()`, чтобы действительно отобразить диаграмму.

```
# Базовая точечная диаграмма, логарифмический масштаб
data_2007 = gapminder[gapminder['year'] == 2007]

plt.scatter(data_2007['gdp_cap'], data_2007['life_exp'])

plt.xscale('log')

# Строки
xlab = 'ВВП на душу населения [в долларах США]'
ylab = 'Ожидаемая продолжительность жизни [в годах]'
title = 'Развитие мира в 2007 году'

# Добавить подписи осей
plt.xlabel(xlab)
plt.ylabel(ylab)

# Добавить заголовок
plt.title(title)

# После настройки, показать график
plt.show()
```



Отметки

Настройки, которые вы уже настроили в скрипте, приведены в более краткой форме.

Демонстрация как можно управлять y-отметками, указав два аргумента:

```
plt.yticks([0,1,2], ["один", "два", "три"])
```

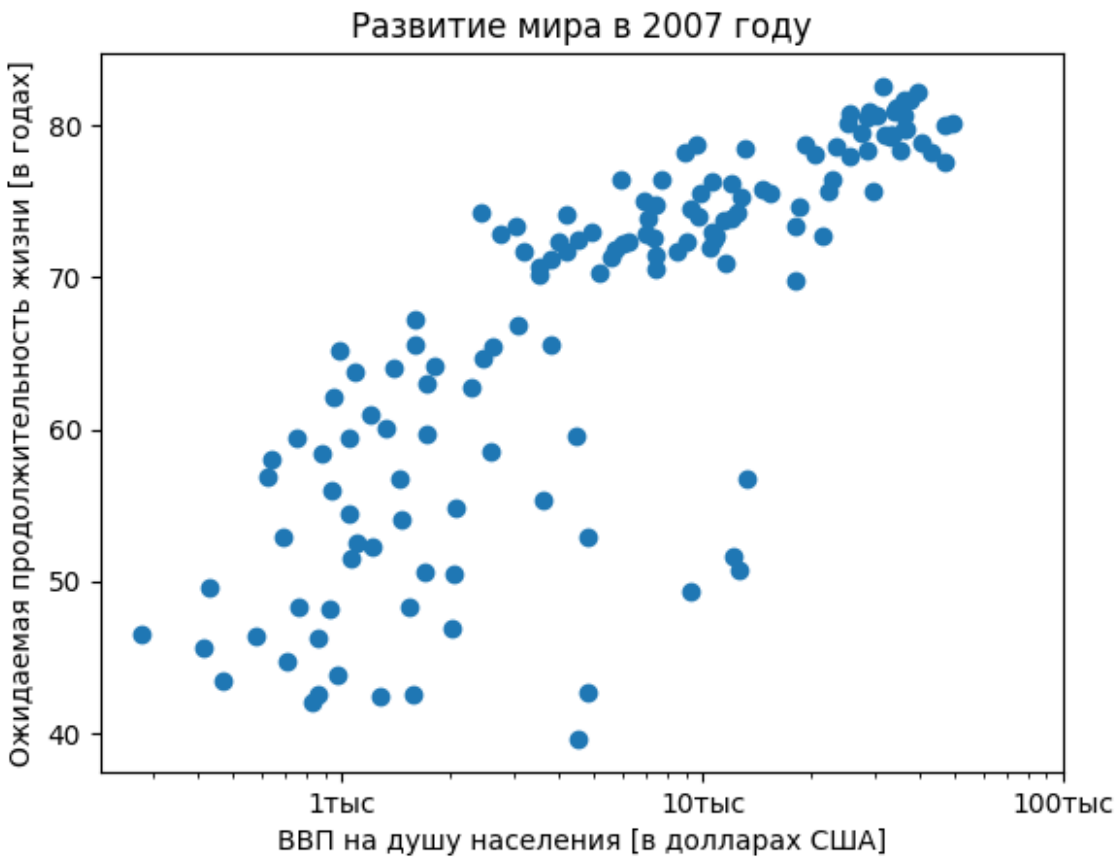
В этом примере отметки, соответствующие числам 0, 1 и 2, будут заменены на один, два и три соответственно.

Давайте сделаем что-то подобное для x-оси вашей диаграммы мирового развития с помощью функции `xticks()`. Значения отметок 1000, 10000 и 100000 должны быть заменены на 1k, 10k и 100k. Для этого уже созданы два списка: `tick_val` и `tick_lab`.

Инструкции

- Используйте `tick_val` и `tick_lab` в качестве входных данных для функции `xticks()`, чтобы сделать график более читаемым.
- Как обычно, отобразите график с помощью `plt.show()` после того, как добавите настройки.

```
# График рассеяния
plt.scatter(data_2007['gdp_cap'], data_2007['life_exp'])

# Предыдущие настройки
plt.xscale('log')
plt.xlabel('ВВП на душу населения [в долларах США]')
plt.ylabel('Ожидаемая продолжительность жизни [в годах]')
plt.title('Развитие мира в 2007 году')

# Определение значений и подписей для делений на оси x
tick_val = [1000, 10000, 100000]
tick_lab = ['1тыс', '10тыс', '100тыс']

# Адаптация делений на оси x
plt.xticks(tick_val, tick_lab)

# После настройки отобразите график
plt.show()
```



Размеры

Прямо сейчас диаграмма рассеяния представляет собой просто облако синих точек, неотличимых друг от друга. Давайте это изменить. Не было бы здорово, если бы размер точек соответствовал населению?

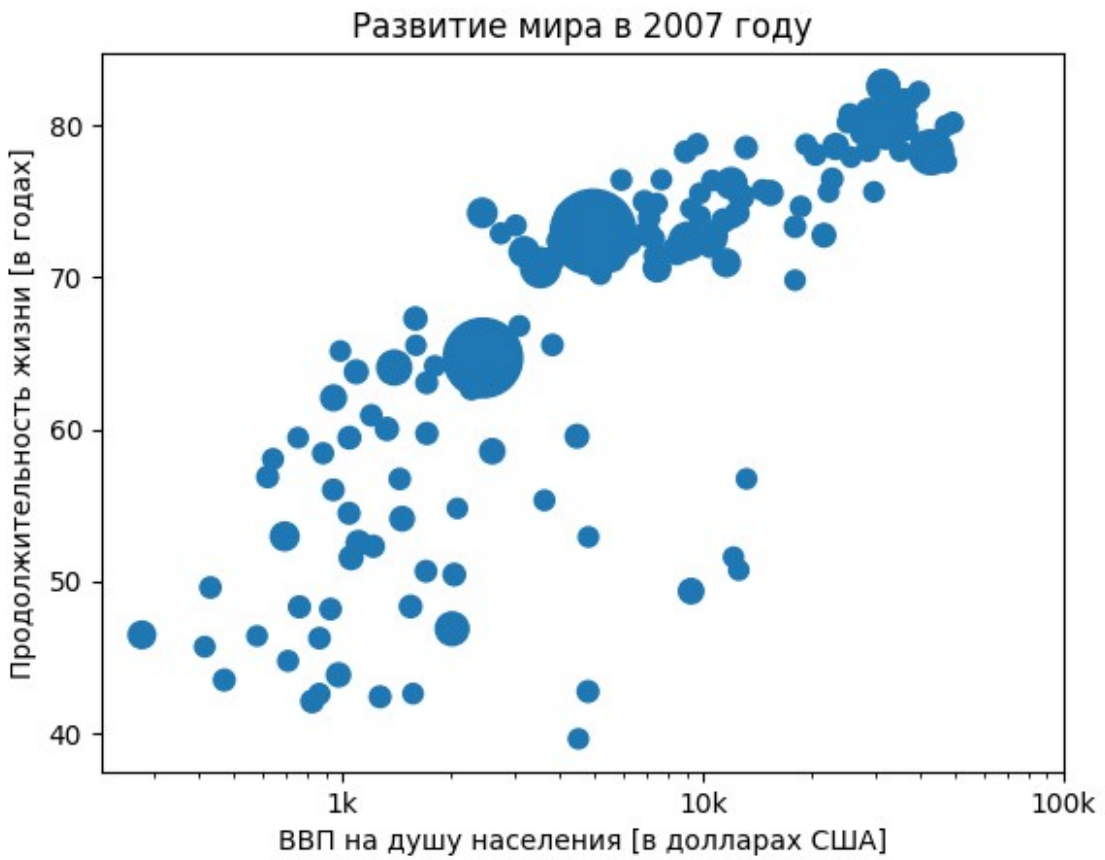
Для этого в вашей рабочей области загружен список '`pop`'. Он содержит численность населения для каждой страны, выраженную в миллионах. Вы можете видеть, что этот список добавлен в метод scatter в качестве аргумента `'s'` для размера.

Инструкции

- Запустите скрипт, чтобы увидеть, как изменится график.
- Выглядит неплохо, но увеличение размера пузырей сделает все более заметным.
- Импортируйте пакет '`numpy`' как '`np`'.
- Используйте '`np.array()`', чтобы создать массив '`numpy`' из списка '`pop`'. Назовите этот массив NumPy '`np_pop`'.
- Удвойте значения в 'np_pop', установив значение 'np_pop' равным 'np_pop * 2'. Поскольку `np_pop` - это массив NumPy, каждый элемент массива будет удвоен.
- Измените аргумент '`s`' внутри '`plt.scatter()`' на '`np_pop`' вместо '`pop`'.

```
import numpy as np
```

```
# Нормализация значений населения для отображения размеров
pop_min = data_2007['population'].min()
pop_max = data_2007['population'].max()

# Нормализация данных о населении
normalized_pop = (data_2007['population'] - pop_min) / (pop_max -
pop_min)

# Настройка отображения размеров точек в подходящем диапазоне,
учитывая масштаб диаграммы рассеяния
size_range = (50, 1000)  # Установка диапазона размеров точек

# Расчет размеров точек на основе нормализованного населения
dot_sizes = np.interp(normalized_pop, (0, 1), size_range)

# Диаграмма рассеяния с размерами точек, соответствующими
нормализованным данным о населении
plt.scatter(data_2007['gdp_cap'], data_2007['life_exp'], s=dot_sizes)

# Предыдущие настройки
plt.xscale('log')
plt.xlabel('ВВП на душу населения [в долларах США]')
plt.ylabel('Продолжительность жизни [в годах]')
plt.title('Развитие мира в 2007 году')
plt.xticks([1000, 10000, 100000],['1k', '10k', '100k'])

# Отображение графика
plt.show()
```



Цвета

Код, который вы написали до сих пор, доступен в скрипте.

Следующий шаг - сделать график более красочным! Для этого для вас создан список col. Это список с цветом для каждой страны в зависимости от континента, к которому принадлежит страна.

Как мы создали список `col`, вы спросите? В данных `Gapminder` есть список `continent` с континентом, к которому принадлежит каждая страна. Составляется словарь, который отображает континенты на цвета:

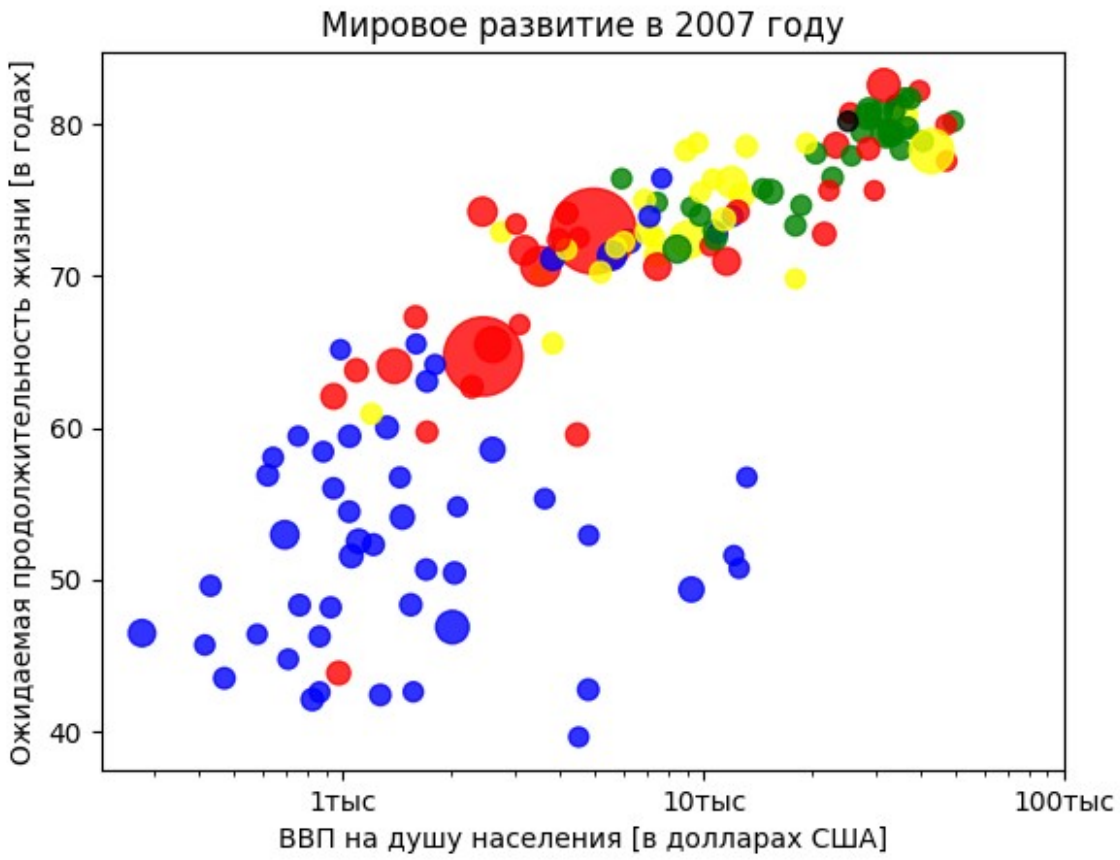
```
dict = {
    'Asia':'red',
    'Europe':'green',
    'Africa':'blue',
    'Americas':'yellow',
    'Oceania':'black'
}
```

Инструкции:

- Добавьте c = col в аргументы функции `plt.scatter()`.

- Измените прозрачность пузырей, установив аргумент `alpha` на `0.8` внутри `plt.scatter()`. Альфа может быть установлена от нуля до единицы, где ноль полностью прозрачен, а единица совсем не прозрачна."

```
# Предполагается, что столбец 'continent' содержит информацию о континенте в вашем DataFrame

# Диаграмма рассеяния с цветами на основе континента и размером по населению
plt.scatter(data_2007['gdp_cap'], data_2007['life_exp'], s=dot_sizes, c=colors, alpha=0.8)

# Предыдущие настройки
plt.xscale('log')
plt.xlabel('ВВП на душу населения [в долларах США]')
plt.ylabel('Ожидаемая продолжительность жизни [в годах]')
plt.title('Мировое развитие в 2007 году')
plt.xticks([1000, 10000, 100000], ['1тыс', '10тыс', '100тыс'])

# Показать график
plt.show()
```



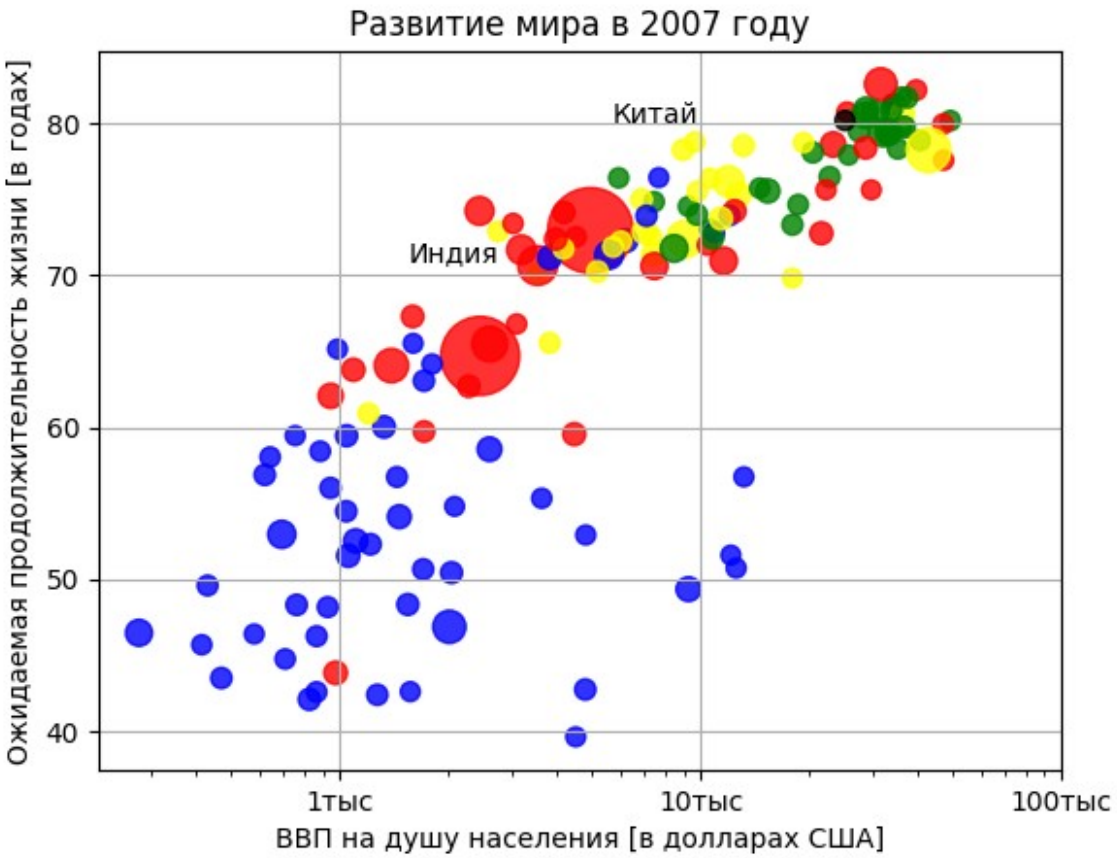
Дополнительные настройки

Если вы еще раз взглянете на скрипт, в разделе `# Дополнительные настройки`, вы увидите две функции `plt.text()`. Они добавляют слова `India` и `China` на график.

Инструкции

Добавьте `plt.grid(True)` после вызовов `plt.text()`, чтобы на графике были нарисованы линии сетки.

```
# Разброс данных
plt.scatter(data_2007['gdp_cap'], data_2007['life_exp'], s=dot_sizes,
c=colors, alpha=0.8)

# Предыдущие настройки
plt.xscale('log')
plt.xlabel('ВВП на душу населения [в долларах США]')
plt.ylabel('Ожидаемая продолжительность жизни [в годах]')
plt.title('Развитие мира в 2007 году')
plt.xticks([1000, 10000, 100000], ['1тыс', '10тыс', '100тыс'])

# Дополнительные настройки
plt.text(1550, 71, 'Индия')
plt.text(5700, 80, 'Китай')

# Добавление сетки
plt.grid(True)

# Отображение графика
plt.show()
```


Развитие мира в 2007 году
Китай
Индия
Ожидаемая продолжительность жизни [в годах]
80
70
60
50
40
1тыс
10тыс
100тыс
ВВП на душу населения [в долларах США]