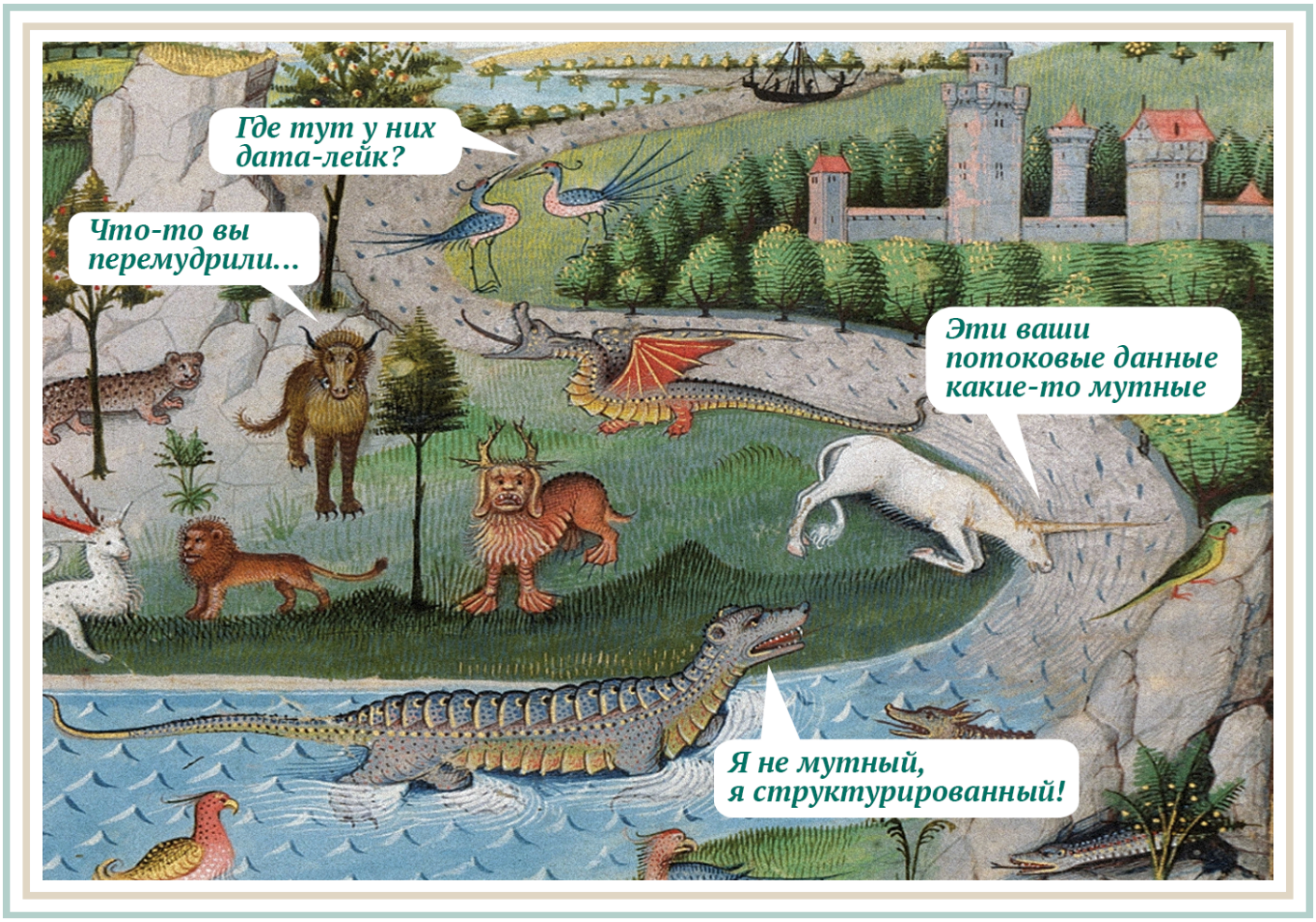
Занятие 4. Данные. Сбор данных



4.1 Поговорим о данных

4.1.1 А что такое данные?

Современный мир невозможно представить без **информации**. Человечество обменивается между собою знаниями, накапливает сведения об окружающем мире, получает данные... Очень много, казалось бы, схожих понятий, говорящих об одном и том же, но нет — между ними есть различия. Например, **данные** отличаются от информации: бумажная книга содержит массу полезной информации, но её нельзя назвать данными.

Если посмотреть толкование этих определений в Оксфордском словаре, то можем увидеть следующее:

- **Data**: 1. Известные факты, используемые для вывода или расчета. 2. Числовые и нечисловые значения характеристик кого-либо (чего-либо), с которыми выполняет операции компьютер или другое подобное устройство.
- **Information**: 1а. Что-то, что было сообщено; знания. 1б. Элементы знаний; новости. 2. Обвинение или жалоба, поданная в суд, и т. п.

Глядя на эти определения, вопрос "что первичнее: ~~курица или яйцо?~~ ~~дух или материя?~~ данные или информация?" отпадает сам собой, поскольку получается, что **данные** — это результаты измерений, просто необработанный сигнал, который не несёт смысла, а **информация** — это обработанные данные: полезный, наполненный смыслом, сигнал.

4.1.2 Признаки и наблюдения

Пока мы не двинулись дальше, коротко остановимся на паре нюансов. В науке о данных факты называют признаками, а то, к чему эти факты относятся, — наблюдениями. Да, это сложная формулировка, но примеры ниже расставят всё по своим местам:

Наблюдение	Признак
Человек	Возраст
Кошка	Привита или нет
6Б класс лицея №3	Средний балл по истории
Книга «Война и мир»	Количество символов
Пассажир «Титаника»	Класс каюты

Наблюдением может быть:

- индивид;
- группа индивидов;
- единичные природные явления;
- тексты;
- изображения, аудио или видео;
- и т. д.

А вообще, какие признаки о наблюдении нужно фиксировать? И какие наблюдения нужно выбрать? Исследователь определяет это сам — как раз после операционализации предположения. Главное, чтобы для каждого наблюдения собирался идентичный набор признаков. Мы можем сравнивать между собой зелёное и тёплое, но будет лучше, если мы сравним зелёное с красным, а тёплое с холодным.

4.1.3 Формализация данных

Предположим, что мы собрали некоторые данные, то есть произвели измерения. Что дальше? Дальше наблюдения и их признаки нужно записать в структурированной форме, чтобы их можно было анализировать. Это называется формализация. **Формализация** — представление данных в структурированной форме, подходящей для анализа. Например, если мы захотим зафиксировать судебную статистику, то нужно будет сделать таблицу, в которой будет несколько колонок:

Страна	Год	Количество судебных решений по уголовным преступлениям
Германия	1985	---
Германия	1986	---
---	---	---
Франция	1985	---
Франция	1986	---

Практически всегда формализация данных предполагает сохранение данных в каком-либо формате. Например:

- простой и неструктурированный текст (форматы .txt, .doc);
- таблица (.csv);
- структурированный текстовый файл (.xml, .json).

4.1.4 Переменная и её шкала

Напоследок мы добавим ещё одно определение, без которого картина не была бы полной. Но сначала зафиксируем, что уже усвоили.

Если рассмотреть таблицу выше, то:

- сама таблица соответствует всем доступным данным по теме;
- каждая отдельная строка — это одно наблюдение;
- каждая отдельная ячейка в строке — признак наблюдения;
- а чему соответствует колонка?

Переменной. Если объяснять на пальцах, то переменная — это «коробка», в которую мы кладём признаки отдельных наблюдений. В этом плане данные похожи на котиков — они тоже любят коробки. Если говорить более строго, то **переменная** — общая характеристика, которую можно измерить или посчитать. У переменной есть свой диапазон значений, который называется *шкалой*. **Шкала** — это система отношений между реальными объектами, ситуациями и значениями и условными значениями, которые им присвоены.

Мы делим переменные по типу шкалы на группы. Вот несколько примеров:

Шкала (Тип переменной)	Пример	Возможные значения	Описание
Номинальная	Регион проживания	г. Санкт-Петербург, Тюменская область	Неупорядоченные текстовые или числовые значения
Ранговая	Уровень счастья человека	Высокий, средний, низкий	Упорядоченные текстовые или числовые значения
Интервальная	Возраст человека, округлённый до 10 лет	11–20, 21–30	Упорядоченные числовые значения, которые разделены на равные интервалы
Непрерывная или абсолютная	Рост человека	167, 178, 203	Упорядоченные числовые значения

Это важно запомнить, потому что тип переменной определяет, что можно и что нельзя с ней делать. Иногда нам даже приходится преобразовывать переменную в другой тип, чтобы проверить своё предположение. Подробнее об этом мы будем говорить дальше, а сейчас даём такое вот простое пояснение, чтобы сильно вас не путать, — за него нам наверняка прилетит от коллег-статистиков.

Хотя шкала ассоциируется с линейкой и последовательным расположением элементов на ней, некоторые шкалы нельзя упорядочить — например, номинальную. Так, в Санкт-Петербурге есть восемнадцать районов, которые мы можем использовать для кодирования адреса наблюдений, но их нельзя расставить по порядку от большего к меньшему.

Ещё, чтобы сделать свою жизнь проще, мы часто называем первые три типа категориальными (т. е. дискретными) шкалами. **Дискретность** — свойство, противопоставляемое непрерывности, прерывистость. Например, расстояния между городами или рост человека могут быть выражены любым произвольным значением в метрах. А дискретность — это любое нарушение непрерывности. Вот несколько примеров дискретных переменных: цвет глаз, жанр книги, школьная оценка и номер подъезда. **Непрерывная переменная** — переменная, которая может принимать любые значения в некотором интервале. **Дискретная переменная** — переменная, которая может принимать ограниченный диапазон значений.

Важно не только то, что именно мы фиксируем (признак), но и то, каким образом мы это делаем (переменная и шкала, в которой она выражена). Одни и те же признаки могут быть выражены разными способами, которые определяют наши дальнейшие действия. Позднее мы поговорим о том, что мы можем сделать с собранными данными.

4.2 Сбор и очистка данных

Вокруг нас много уже существующих данных. Это могут быть результаты уже проведенных опросов, цифровые следы интернет-пользователей или сведения об эффективности лекарств. Когда мы проводим собственное исследование, полезно обратиться к уже имеющимся данным. Их часто называют вторичными.

4.2.1 Международная и национальная статистика

Обычно перед началом исследования нам необходимо оценить число всех возможных наблюдений. Это поможет нам построить корректную выборку и понять, какие стратегии сбора данных могут быть уместны. Для этого можно использовать данные национальных статистических ведомств. Главное из них в России — Федеральная служба государственной статистики (Росстат), но не только. Национальные данные могут собираться разными способами:

- это опросы разных уровней (от региональных подразделений организаций до масштабных исследований Росстата);
- статистические показатели ЗАГСов (например, их данные по рождаемости и смертности — наиболее надёжные из всех, что у нас есть);
- отчёты по разнообразным показателям муниципальных и федеральных ведомств (Роструд, Центробанк, Минэкономразвития и так далее).

Предположим, мы решили исследовать взгляды жителей нескольких городов с населением больше миллиона на проблему гендерного неравенства. Чтобы грамотно составить выборку и представлять себе генеральную совокупность, нам могут пригодиться демографические данные об изучаемых городах – ожидаемая продолжительность жизни у мужчин и женщин, среднее количество детей в семье и др. Такие данные мы можем получить из национальных статистик и переписей населения. Впрочем, национальные данные (и в России, и в других странах), к сожалению, часто бывают низкого качества: с искажениями, неполной информацией, дубликатами наблюдений и так далее. Кроме того, эти данные обычно находятся в закрытом режиме доступа.

Но зачастую для решения наших задач требуется оценить проблему в общем и сравнить её с другими примерами. Допустим, мы изучаем связь между средним уровнем образования в стране и мнением о проблеме гендерного неравенства. В этом случае кажется логичным начать разбираться в проблеме с того, чтобы обратиться к мировому опыту – нам пригодится международная статистика. Такие данные собирают и публикуют международные организации, такие как ООН и Всемирный банк.

При этом важно помнить, что данные для такой статистики предоставляют национальные министерства, НКО и другие организации. Поэтому, например, Россия в статистике ООН будет представлена российскими же данными.

Поскольку мы рассуждаем о связи между средним уровнем образования в стране и взглядами на проблему гендерного неравенства, то мы можем обратиться к информации о том, какое место изучаемая нами страна занимает в стороннем международном рейтинге гендерного равенства. Таких рейтингов на разных уровнях довольно много, их составляют по самым разным вопросам.

Например, все мы встречали рейтинги университетов. Они бывают общемировые, так и по конкретным регионам, предметам, сложности вступительных испытаний. В основе таких рейтингов часто экспертная оценка группы или организации, которая ранжирует участников по собственным параметрам.

Стоит быть внимательным к выбору данных и их источников. За данными может стоять определённая методология (зачастую подробно описанная), но она может иметь недостатки, которые могут помешать использовать их в исследовании. Например, в неё могут быть включены субъективные мнения экспертов о том, как именно должны быть собраны данные. Обычно это приводит к появлению в методологии допущений или упрощений, который могут значительно повлиять на смысл собранных данных. Вопросы могут быть не только к методологии, но и к качеству сбора. Поэтому нужно держать в уме все потенциальные сложности, чтобы оценить степени их влияния на итоговый результат.

4.2.2 Парсинг открытых данных

Интернет-пользователи оставляют большое количество цифровых следов, анализ которых может быть полезным для социального исследования. Источниками таких цифровых следов могут быть:

- социальные сети и блоги;
- новостные сайты (например, раздел комментариев);
- государственные и негосударственные площадки для обсуждений (например, онлайн-петиции). Цифровые следы можно преобразовать в данные. Для этого можно использовать API (application programming interface) или сохранить себе нужные интернет-страницы и найти в них данные (эту процедуру часто называют парсинг). Если есть выбор между двумя способами, то лучше выбрать первый. API обычно есть у многих государственных онлайн-ресурсов, библиотек и социальных сетей.

API – это набор правил и программных инструкций, которые позволяют двум программам (вашему скрипту для сбора и серверу с данными) взаимодействовать друг с другом.

Предположим, что пользователю необходимо получить перечень русскоязычных академических публикаций в сфере молекулярной биологии за последние полгода. Если база данных научных публикаций имеет встроенный API, исследователь может сформулировать свой запрос, следуя инструкции владельца данных и получить результат в одном из форматов, подходящих для дальнейшего анализа (JSON, XML и др.). Важно понимать, что API предоставляет доступ не ко всем типам информации: у каждого ресурса есть свои внутренние правила, которые обычно описаны в соответствующем разделе сайта.

Поскольку большая часть интернет-ресурсов не имеет доступа к API, для сбора данных применяют парсинг. Это может быть актуально, например, для мониторинга меняющейся информации: цен на товар, погоды и др. Такие данные можно собирать вручную (путём простого копирования/скачивания необходимой информации) или же автоматизировать процесс с помощью парсера – специально написанной для вашей задачи программы.

Суть работы парсера вкратце можно описать таким образом. Любая информация, находящаяся в интернете, размечена с помощью HTML. У каждой веб-страницы есть код, в котором описаны типы данных (изображение, текст и др.), их характеристики (размер и цвет шрифта), другие детали (заголовки, основной текст). Программа-парсер понимает этот код и распознаёт его элементы. Нам достаточно сформулировать, какие данные мы хотим получить.

Предположим, нам нужно проанализировать комментарии, которые оставляют пользователи под новостями в разделе «Здоровье и медицина». В первую очередь необходимо определить интересующие нас HTML-элементы. Парсер будет искать аналогичные элементы и выполнять ту работу, которая заняла бы очень много времени вручную: открывать десятки и сотни страниц новостей в нужно разделе, находить комментарии, выгружать их в общий документ.

Извлекаемая информация может содержать персональные данные или объекты авторского права – работа с такими данными может быть рискованной со стороны законодательства.

4.3 Парсинг данных и автоматизация браузера, регулярные выражения

Веб-страницы и её элементы разечены с помощью языка HTML. Коротко о том, как он устроен.

4.3.1 Совсем немного про HTML

С помощью HTML мы подсказываем браузеру (Google Chrome, Safari, Opera), из каких семантических единиц (заголовок, абзац, ссылка и так далее) состоит контент на странице. Совсем простой пример выглядит так:

```
<h1>Это заголовок</h1>
<p>Это текст.</p>
```

HTML — это язык разметки. Ключевое слово — язык разметки. Это не язык программирования, не путайте. В разницу между этими понятиями мы углубляться не будем, просто запомните это.

Вот что происходит, когда вы вводите в браузер адрес веб-сайта:

- ваш компьютер отправляет запрос на сервер, на котором хранится HTML-документ;
- сервер отвечает на запрос и присылает HTML-документ для запрошенной страницы;
- браузер читает этот код и вычисляет, как отобразить элементы (заголовки, картинки, ссылки) и где они должны располагаться на странице. HTML состоит из тегов — в нашем примере это <h1></h1> и . Именно благодаря тегам браузер понимает, из каких элементов состоит страница: для него это что-то вроде инструкции по сборке элементов.

У тега могут быть атрибуты. Они нужны для передачи дополнительной информации браузеру. Выглядят атрибуты так: это пара ключ="значение". В примере ниже тэг — это элемент «гиперссылка». У него есть атрибут href, значение которого — адрес сайта, куда будет отправлен пользователь при клике.

```
<a href="https://www.w3schools.com/html/html_attributes.asp">Ссылка на статью об HTML-аттрибутах</a>
```

Ссылка на статью об HTML-аттрибутах (https://www.w3schools.com/html/html_attributes.asp)

4.3.2 Сверим часы

Нужно понимать, что современные технологии не стоят на месте и что защита данных от несанкционированного парсинга тоже шагнула далеко вперёд в своём развитии. Если несколько лет назад было достаточно обратиться к данным веб-страницы на сервер, то сейчас большая часть контента генерируется на стороне клиента благодаря средствам javascript. Поэтому мы будем имитировать действия обычного пользователя, чтобы заходить на определённые ресурсы и брать необходимые данные. Для этого может потребоваться библиотека, используемая в области тестирования веб-интерфейсов, которая называется Selenium.

Но до сих пор существуют сайты, которые согласны отдавать по API данные. К примеру, небезызвестный Sunlight может предоставить данные о своих товарах, если попросить его об этом.

In [1]:

```
#!pip install beautifulsoup4
```

In [2]:

```
from bs4 import BeautifulSoup as bs
import requests
import pandas as pd

html = requests.get('https://sunlight.net/catalog').text  # получаем html код сайта
soup = bs(html)  # создаём экземпляр класса BeautifulSoup

data_dict = {
    'url' : [],
    'name' : [],
    'price' : []
}

links = soup.find_all('a', class_='cl-item-link js-cl-item-link js-cl-item-root-link')  # получаем список ссылок и наи
for i, link in enumerate(links):
    url = link.get("href")  # получаем ссылку товара
    name = link.get_text()  # извлекаем наименование из блока со ссылкой
    price = soup.find_all("div", class_="cl-item-info-price-discount")[i].get_text()  # извлекаем цену
    print(f"Запись №{i}\nURL: {url}\nName: {name}\nPrice: {price}\n\n")
#     print(i)
#     print(f"Url - {url}")
#     print(f"Name - {name}")
#     print(f"Price - {price}\n")
    data_dict['url'].append(url)
    data_dict['name'].append(name)
    data_dict['price'].append(price)
```

```
/home/agat.local/s.bulganin/anaconda3/lib/python3.11/site-packages/pandas/core/arrays/masked.py:60: UserWa
rning: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (

Запись №0
URL: /catalog/earring_251657.html
Name:
Серьги, вставка: фианитом бесцветным; Розовое золото 585 пробы.

Price:
                            11 990 ₽



Запись №1
URL: /catalog/ring_348425.html
Name:
Кольцо, вставка: фианит бесцветный; Розовое золото 585 пробы.

Price:
```

In [3]:

```
data_dict
```

Out[3]:


```
{'url': ['/catalog/earring_251657.html',
  '/catalog/ring_348425.html',
  '/catalog/ring_69933.html',
  '/catalog/pendants_351578.html',
  '/catalog/earring_286758.html',
  '/catalog/ring_165495.html',
  '/catalog/earring_57987.html',
  '/catalog/ring_58020.html',
  '/catalog/ring_80829.html',
  '/catalog/earring_80960.html',
  '/catalog/ring_56676.html',
  '/catalog/earring_56677.html',
  '/catalog/ring_100464.html',
  '/catalog/ring_83348.html',
  '/catalog/earring_108798.html',
  '/catalog/ring_108796.html',
  '/catalog/ring_69894.html',
  '/catalog/ring_51515.html',
```


In [4]:

```
# превращаю словарь в датафрейм
data = pd.DataFrame(data_dict)
data.head(5)
```

Out[4]:

	url	name	price
0	/catalog/earring_251657.html	\nСерьги, вставка: фианитом бесцветным; Розово...	\n 11 990 ₽\n ...
1	/catalog/ring_348425.html	\nКольцо, вставка: фианит бесцветный; Розовое ...	\n 10 990 ₽\n ...
2	/catalog/ring_69933.html	\nКольцо с 1 гидротермальным изумрудом, 0.26 к...	\n 13 990 ₽\n ...
3	/catalog/pendants_351578.html	\nПодвеска с 1 гидротермальным изумрудом, 0.3 ...	\n 5 990 ₽\n ...
4	/catalog/earring_286758.html	\nСерьги с 20 бриллиантами, 0.03 карат, огранк...	\n 14 990 ₽\n ...

Задание

1. Имеется список компаний:

`companies = ['Проект по использованию технологий компьютерного зрения на базе искусственного интеллекта (ИИ) для анализа медицинских изображений','Skillbox ', 'MMA.Metaratings','Метарейтинг','СберМаркет','Balance Platform','Московская биржа','Samsung Electronics','Нетология','Дневник МЭШ','Цифровое образование']` На основе имеющихся данных из аналитических статей, связанных с информационными технологиями, бизнесом и интернетом с сайта (https://habr.com/ru/search/ (https://habr.com/ru/search/)) необходимо построить исходный набор данных. Набор данных должен включать названия, описание, рейтинг и сферу деятельности компаний, дату публикации, а также текст статей из Интернет-ресурсов. Разработанный парсер должен извлекать гиперссылки из начальной страницы с последующим обходом всех страниц по полученным ссылкам и извлечением их содержимого. В помощь документация к библиотеке beautifulsoup4:
https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)
P.S. Для создания запроса к API можно использовать код ниже:

In [ ]:

```
name_company1 = '%20'.join(name_company.split())
url = 'https://habr.com/ru/search/?q='+ name_company1 + "&target_type=companies&order=relevance"
```

In [ ]:

```

```

2. A. Изучить статью (https://www.scrapingbee.com/blog/selenium-python/);
   B. Имеется некоторый номер телефона `NUMBER`, к которому привязан аккаунт в Telegram. Также известно, что чат с человеком по этому аккаунту можно открыть в веб-версии, используя ссылку вида `https://t.me/{NUMBER}`. Используя средства для веб-скраппинга, написать скрипт, который отправит мне сообщение `Это сообщение отправлено автоматически {Имя} {Фамилия}. Отвечать не нужно.` Номер телефона я дам. При написании решения поможет статья (https://habr.com/ru/articles/735394/)

In [ ]:

```

```