

**Министерство науки и высшего образования Российской Федерации**  
Федеральное государственное автономное образовательное учреждение  
высшего образования



**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

---

Инженерная школа информационных технологий и робототехники  
Отделение информационных технологий  
Направление подготовки 09.04.04 Программная инженерия

**Отчёт по лабораторной работе №4**

**Веб-кроулер**

по дисциплине Представление знаний в системах искусственного интеллекта

Выполнил студент гр. 8ПМ4Л

\_\_\_\_\_  
Подпись

\_\_\_\_\_  
Дата

Сокуров Р.Е.  
Фамилия И.О.

Проверил лаборант ОИТ

\_\_\_\_\_  
Подпись

\_\_\_\_\_  
Дата

Сапегин А.А.  
Фамилия И.О.

Томск 2024 г.

## **Задание**

1. Установить Scrapy

2. Извлечь из веб-ресурса Steam (<https://store.steampowered.com/>).

Используя CrawlSpider из библиотеки Scrapy, извлечь информацию: название игры, дата выхода, разработчик, издатель, популярные метки для этого продукта для первых 1000 продуктов из списка «Лидеры продаж». Обратите внимание на продукты, не являющиеся играми, и комплекты (бандлы). Они не должны входить в данный перечень из собранных записей.

3. Сгруппировать игры по годам. В каждом году определить 3 самых популярных тега и 3 самых непопулярных. Вывести их.

4. Вывести игры, соответствующие самым популярным и непопулярным тегам из п.4 (в соответствии с годом, т.е. если тег принадлежит к одному году, а игра с данным тегом – к другому, то не выводить такую информацию). Примечание: например, если тег «Решения с последствиями» попало в самые непопулярные теги 2020 года, но не попало в 2019, то из игры с непопулярными тегами такие игры должны быть выведены только для 2020 года.

## Ход работы

Для установки Scrapy использовался менеджер пакетов Python pip:


```
1. pip install scrapy
```

Затем был создан проект Scrapy

```
1. scrapy startproject steam_scraper
```

Дальше, оказавшись в директории «Spiders» проекта был создан новый файл «steam\_spider.py». Его код с комментариями представлен в приложении А.

Далее был выполнен запуск паука с сохранением результата работы в файл «steam\_data.csv»:



```
steam_scraper — scrapy crawl steam -o steam_data.csv — 80x24
2025-01-07 12:07:03 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/473100/?snr=1_5_9__205> (referer: https://store.steampower
ed.com/app/473100/Shmups_Skill_Test_Original_Soundtrack/?snr=1_5_9__405)
2025-01-07 12:07:04 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/379520/> (referer: https://store.steampowered.com/app/4064
90/DELTAZEAL_Original_Soundtrack/?snr=1_5_9__405)
2025-01-07 12:07:04 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/406490/?snr=1_5_9__205> (referer: https://store.steampower
ed.com/app/406490/DELTAZEAL_Original_Soundtrack/?snr=1_5_9__405)
2025-01-07 12:07:05 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/435970/> (referer: https://store.steampowered.com/app/4850
10/RefRain__prism_memories__Chronicle_Visual_Book/?snr=1_5_9__405)
2025-01-07 12:07:05 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/485010/?snr=1_5_9__205> (referer: https://store.steampower
ed.com/app/485010/RefRain__prism_memories__Chronicle_Visual_Book/?snr=1_5_9__405
)
2025-01-07 12:07:06 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/1832490/?snr=1_5_9__205> (referer: https://store.steampowe
red.com/app/1832490/Crimzon_Clover_World_EXplosion__Complete_Soundtrack/?snr=1_5
_9__405)
2025-01-07 12:07:07 [scrapy.core.engine] DEBUG: Crawled (200) <GET https://store
.steampowered.com/app/1166290/?snr=1_5_9__205> (referer: https://store.steampowe
red.com/app/1166290/Death_and_Taxes/)
```

Рисунок 1 – Запуск паука

Получили файл с 1091 записью:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC
1																													
2																													
3																													
4																													
5																													
6																													
7																													
8																													
9																													
10																													
11																													
12																													
13																													
14																													
15																													
16																													
17																													
18																													
19																													
20																													

Рисунок 2 – Содержимое файла «steam\_data.csv»

Далее был проведён анализ полученной информации используя язык Python. Содержимое файла «data\_analysis.ipynb» приведён в приложении Б.

## Вывод

В ходе лабораторной работы была реализован веб-кроулер для считывания самых популярных игр из веб-ресурса Steam. Данные были успешно собраны и проанализированы используя библиотеку Pandas языка Python.

## **Приложение А**

Содержимое файла «steam\_spider.py»

```

import scrapy
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor

class SteamSpider(CrawlSpider):
    name = "steam"
    allowed_domains = ["store.steampowered.com"]
    start_urls =
['https://store.steampowered.com/search/?category1=998&supportedlang=english&filter=globaltopsell&ndl=1']

    # Уникальные игры
    unique_games = set()

    # Счётчик добавленных записей
    records_count = 0

    custom_settings = {
        'CONCURRENT_REQUESTS': 64,
        'DOWNLOAD_DELAY': 0.5,
        'DEFAULT_REQUEST_HEADERS': {
            'Accept-Language': 'english'
        },
    }

    rules = (
        Rule(LinkExtractor(allow=r'/app/\d+/', deny=(r'\?l=')),
callback='parse_game', follow=True),
    )

    def parse_game(self, response):
        title = response.xpath('//*[@id="appHubAppName"]/text()').get()
        release_date =
response.xpath('//*[@id="game_highlights"]/div[1]/div/div[3]/div[2]/div[2]/text()').get()
        developer = response.xpath('//*[@id="developers_list"]/a/text()').get()
        publisher =
response.xpath('//*[@id="game_highlights"]/div[1]/div/div[3]/div[4]/div[2]/a/text()').get()
        tags =
response.xpath('//*[@id="glanceCtnResponsiveRight"]/div[2]/div[2]/a/text()').getall()
        bundle =
response.xpath('//*[@id="game_area_purchase_top"]/div/h1/span/text()').get()
        music =
response.xpath('//*[@id="game_area_purchase"]/div[1]/div/h1/text()').get()

        # Проверка на уникальность
        if title and title in self.unique_games:

```

```

        return
    self.unique_games.add(title)

    # Очистка данных
    game_data = {
        'title': title.strip() if title else None,
        'release_date': release_date.strip() if release_date else None,
        'developer': developer.strip() if developer else None,
        'publisher': publisher.strip() if publisher else None,
        'tags': [tag.strip() for tag in tags],
        'bundle': bundle.strip() if bundle else None,
        'music': music.strip() if music else None,
    }

    # Фильтрация по "Bundle"
    if game_data['bundle'] is not None and game_data['bundle'] == 'Bundle':
        return

    # Фильтрация по музыкальным тегам
    music_tags = ["downloadbare soundtrack", "downloadable content",
                  "Downloadable Soundtrack", "Downloadable Content"]
    for tag in music_tags:
        if game_data['music'] is not None and tag == game_data['music']:
            return

    # Увеличиваем счётчик только при добавлении в документ
    if game_data['title'] and game_data['developer']:
        self.records_count += 1
        yield game_data

    # Остановка паука при достижении лимита
    if self.records_count >= 1000:
        self.crawler.engine.close_spider(self, reason='Reached limit of
1000 titles')

```

## **Приложение Б**

Содержимое файла «data\_analysis.ipynb»