DOI: 10.32517/2221-1993-2025-24-6-38-44

CC BY

Р. В. Коротенко
Белгородский государственный национальный исследовательский университет, г. Белгород, Россия

Э. В. Провоторов
Белгородский государственный национальный исследовательский университет, г. Белгород, Россия; Дубовская средняя общеобразовательная школа с углубленным изучением отдельных предметов, пос. Дубовое, Белгородский район, Белгородская область, Россия

И. Н. Беляева
Белгородский государственный национальный исследовательский университет, г. Белгород, Россия

ПАРСИНГ ВЕБ-СТРАНИЦ КАК ИНСТРУМЕНТ СОЗДАНИЯ МЕТОДИЧЕСКИХ МАТЕРИАЛОВ В РАБОТЕ УЧИТЕЛЯ И ПРЕПОДАВАТЕЛЯ: ПРИМЕР ИСПОЛЬЗОВАНИЯ ТЕХНОЛОГИИ

Аннотация

В статье рассматривается использование технологии парсинга веб-страниц для создания методических материалов в образовательной сфере. Парсинг позволяет автоматизировать сбор и обработку информации с веб-ресурсов, что значительно упрощает процесс создания учебных материалов для учителей и преподавателей. В статье описан разработанный инструмент парсинга, предназначенный для сбора и структурирования заданий для подготовки к единому государственному экзамену (ЕГЭ). Этот инструмент позволяет эффективно собирать задания с различных веб-страниц, обрабатывать их и сохранять в базе данных, что обеспечивает удобный доступ к материалам и их дальнейшее использование в образовательном процессе. Результаты исследования показывают, что интеграция парсинга в образовательную практику способствует повышению эффективности управления учебным процессом, экономии времени и ресурсов, а также улучшению качества образования.

Ключевые слова: автоматизация сбора данных, образовательные ресурсы, синтаксический анализ, эффективность преподавания, подготовка к экзаменам, обработка информации, технологии в образовании, парсинг.

Информация об авторах

Коротенко Роман Вячеславович, магистрант факультета математики и естественнонаучного образования, Педагогический институт, Белгородский государственный национальный исследовательский университет, г. Белгород, Россия; *e-mail:* k.roman.vyacheslavovich@gmail.com

Провоторов Эдуард Витальевич, магистрант факультета математики и естественнонаучного образования, Педагогический институт, Белгородский государственный национальный исследовательский университет, г. Белгород, Россия; учитель математики и алгоритмики, Дубовская средняя общеобразовательная школа с углубленным изучением отдельных предметов, пос. Дубовое, Белгородский район, Белгородская область, Россия; *e-mail:* edprovotorov@gmail.com

Беляева Ирина Николаевна, канд. физ.-мат. наук, доцент, зав. кафедрой математики, факультет математики и естественнонаучного образования, Педагогический институт, Белгородский государственный национальный исследовательский университет, г. Белгород, Россия; *e-mail:* ibelyaeva@bsuedu.ru

R. V. Korotenko
Belgorod State National Research University, Belgorod, Russia
E. V. Provotorov
Belgorod State National Research University, Belgorod, Russia;
Dubovskaya School, Dubovoye, Belgorod District, Belgorod Region, Russia
I. N. Belyaeva
Belgorod State National Research University, Belgorod, Russia

WEB PAGE PARSING AS A TOOL FOR CREATING TEACHING MATERIALS IN THE WORK OF TEACHERS AND EDUCATORS: A CASE STUDY OF TECHNOLOGY APPLICATION

Abstract

The article examines the use of web page parsing technology for creating educational materials in the educational sector. Parsing allows for the automation of data collection and processing from web resources, significantly simplifying the process of creating teaching materials for teachers and educators. The article describes a developed parsing tool designed for collecting and structuring tasks for preparation for the Unified State Exam (USE). This tool efficiently collects tasks from various web pages, processes them, and stores them in a database, ensuring easy access to the materials and their further use in the educational process. The research results show

© Коротенко Р. В., Провоторов Э. В., Беляева И. Н., 2025

ISSN 2221-1993 • **Информатика в школе** • 2025 • Том 24 № 6

that the integration of parsing into educational practice contributes to improving the efficiency of educational process management, saving time and resources, and enhancing the quality of education.

Keywords: data collection automation, educational resources, syntactic analysis, teaching efficiency, exam preparation, information processing, educational technologies, parsing.

Information about the authors

Roman V. Korotenko, a master's student at the Faculty of Mathematics and Science, Institute of Pedagogy, Belgorod State National Research University, Belgorod, Russia; *e-mail:* k.roman.vyacheslavovich@gmail.com

Eduard V. Provotorov, a master's student at the Faculty of Mathematics and Science, Institute of Pedagogy, Belgorod State National Research University, Belgorod, Russia; a mathematics and algorithmics teacher, Dubovskaya School, Belgorod District, Belgorod Region, Russia; *e-mail:* edprovotorov@gmail.com

Irina N. Belyaeva, Candidate of Sciences (Physics and Mathematics), Docent, Head of the Department of Mathematics, Faculty of Mathematics and Science, Institute of Pedagogy, Belgorod State National Research University, Belgorod, Russia; *e-mail:* ibelyaeva@bsuedu.ru

1. Введение

В современном образовательном процессе учителя все чаще обращаются к цифровым технологиям для улучшения качества обучения и повышения эффективности своей работы. Одним из таких инструментов является **парсинг веб-страниц**. Парсинг позволяет автоматизировать сбор и обработку информации из интернета, что открывает новые возможности для разработки методических материалов. Введение парсинга в педагогическую практику помогает учителям быстро находить актуальные материалы, адаптировать их под нужды своих учеников и создавать уникальные образовательные ресурсы. При этом автоматизация процесса парсинга значительно сокращает временны́е затраты, которые при ручном сборе данных были бы неразумно высокими. Это делает данный инструмент особенно ценным в условиях ограниченных времени и ресурсов.

О. Н. Ие пишет, что ***парсинг*** *(или синтаксический анализ) представляет собой процесс извлечения и структурирования данных с веб-ресурсов с помощью программ, написанных на языках программирования, таких как PHP, Python и др.* [5]. По ее словам, «“парсить” означает собирать и систематизировать информацию, размещенную на определенных сайтах, с помощью специальных программ, автоматизирующих процесс» [там же, с. 2].

А. В. Ермоленко и соавторы уточняют: «В настоящее время остается актуальной задача получения данных из различных, часто неструктурированных, источников. Для автоматизации решения этой задачи востребован синтаксический анализ, называемый парсингом» [4, с. 56]. Авторы отмечают, что подобные программы преобразуют исходную информацию в более удобный формат, адаптированный под конкретные цели.

П. М. Акимов с соавторами в своей статье приводят более точное определение понятию синтаксического анализа: «Синтаксический анализ — это процесс, который определяет, принадлежит ли некоторая последовательность токенов (слов) какому-либо языку (например, естественному или формальному), описанному правилами некоторой грамматики. Задачей этого этапа является сконструировать эффективный анализатор, который определяет, является ли корректной входная строка. Синтаксический анализ выполняет парсер, он же синтаксический анализатор — часть программы, преобразующая входные данные в структурированный формат» [2, с. 112].

Также авторы статьи [2] отмечают, что парсинг данных является важным инструментом, так как он позволяет разработчикам решить одну из основных проблем — отсутствие структурированности информации. Парсинг помогает справиться с этой проблемой, структурируя данные и делая их более удобными для анализа и использования. В условиях растущей потребности в структурированных данных и автоматизации процессов, значимость парсинга будет только увеличиваться [2, 9].

Если рассмотреть законность автоматизированного сбора информации, можно сделать вывод, что российское законодательство запрещает или ограничивает следующие действия:

1. Нарушение авторских прав.
2. Неправомерный доступ к защищенной законом компьютерной информации.
3. Незаконный сбор сведений, составляющих коммерческую тайну.
4. Злоупотребление правом.
5. Использование гражданских прав для ограничения конкуренции.

Тем не менее, согласно А. А. Караваевой и М. С. Овсовой, **автоматический сбор данных из открытых источников в интернете может быть законным при соблюдении ряда условий** [6, 7]:

1. Информация должна быть доступна публично и не защищена авторским правом.
2. Парсинг должен проводиться законными методами.
3. Процесс парсинга не должен перегружать сайт и мешать его нормальной работе.
4. Автоматический сбор данных не должен ограничивать конкуренцию.
5. Условия использования сайта должны соблюдаться.
6. Личные данные пользователей не должны извлекаться.
7. Контент должен соответствовать стандартам правомерного использования.

А. В. Ермоленко и соавторы выделяют **основные задачи, при которых может потребоваться синтаксический анализ (парсинг)** [4, с. 56–57]:

1. Обновление информации для поддержки актуальности веб-ресурса, например, отслеживание изменения курса валют или прогноза погоды при помощи веб-сервисов.
2. Сбор и быстрое копирование информации с веб-ресурсов для размещения на собственном сайте,

это актуально для новостных проектов, интернет-витрин.

3. Сбор больших данных для дальнейшего анализа, проверки статистических гипотез, машинного обучения при помощи веб-сервисов или веб-скрейпинга; так, парсинг широко используется для сбора данных в биоинформатике, например, у открытой базы данных генетических и белковых взаимодействий модельных организмов и людей BioGRID существует REST API.

Д. З. Цхошвили и Н. А. Иванова замечают, что **технология парсинга активно интегрируется и в образовательную сферу,** что позволяет значительно повысить эффективность различных процессов [10]. Среди примеров такой интеграции авторы выделяют автоматизированные системы оценки труда преподавателей. Эти системы используют парсинг для сбора данных из научных библиотек о публикационной активности, что позволяет объективно оценивать научную работу каждого преподавателя. Кроме того, парсинг применяется для создания мобильных приложений, отображающих расписание занятий университета, что авторы находят удобным для студентов, которым предоставляется доступ к актуальной информации.

Существуют **два способа парсинга, актуальных для решения прикладных задач** [4]: ***использование готовых ресурсов*** и ***создание программ***.

Среди готовых ресурсов выделяют несколько основных видов парсеров [1]:

1. ***Облачные парсеры:*** эти инструменты работают в облаке, не требуют установки и позволяют автоматизировать сбор данных через веб-интерфейс или API. *Примеры:* Import.io, Mozenda, Octoparse.
2. ***Десктопные парсеры:*** программы, устанавливаемые на компьютер, часто с мощным функционалом и гибкими настройками. *Примеры:* Screaming Frog, Netpeak Spider.
3. ***Браузерные расширения:*** инструменты, интегрируемые в браузер, для сбора данных с веб-страниц. *Примеры:* Data Scraper, Kimono.
4. ***Надстройки для Excel и Google Таблиц:*** используют макросы и функции для парсинга данных прямо в таблицах. *Примеры:* ParserOK, функции IMPORTXML и IMPORTHTML.

Относительно ресурсов, которые можно использовать для парсинга, А. А. Караваева пишет: «Для парсинга данных можно использовать уже существующую специально для этого программу либо программу, написанную самостоятельно. Парсеры могут быть написаны на любом языке программирования (Delphi, PHP, C++ и т. д.), который поддерживает регулярные выражения — наборы метасимволов, применяющиеся в процессе поиска данных. Сохранение результатов автоматического сбора информации обычно происходит в файлы формата *.txt, *.sql, *.xml» [6, с. 102].

Хотя готовые сервисы для парсинга данных с сайтов могут быть удобны, но они нередко требуют дополнительных денежных затрат, особенно при частых запросах. В. В. Прокопенко пишет, что в таких случаях имеет смысл разработать **собственное программное решение**. Программа анализирует исходный код страницы, находит нужные данные и сохраняет их в удобном формате, таком как SQL, XML, TXT или таблицы [8].

Кроме этого В. В. Прокопенко отмечает необходимость формирования базы данных при работе парсера: «Для этих целей актуально создать базу данных, которая с помощью интеллектуальных элементов способна выдать требуемую пользователю информацию в кратчайшие сроки» [там же, с. 71].

Т. Э. Вильданов и Н. С. Иванов отмечают, что язык программирования Python является наиболее подходящим инструментом для создания программ-парсеров в связи с его популярностью, легкостью и мощностью [3]. Авторы утверждают, что «это простой, но в то же время очень мощный язык программирования, имеющий множество библиотек для решения всех типов задач» [там же, с. 24].

Именно поэтому, а также из-за отсутствия специализированных API у нужных нам сайтов, их специфической верстки и наших особых требований к сбору данных мы решили использовать второй способ парсинга, применяя для этого язык программирования Python. Это позволит нам реализовать **цель:** *создать собственную полную базу заданий ЕГЭ по информатике, демонстрируя практическое применение данной технологии для методических разработок в образовательном процессе*.

Перед непосредственным написанием программы необходимо определиться с библиотекой для Python, используемой для осуществления сбора данных с интернет-ресурсов.

А. Н. Чирков называет BeautifulSoup4 удобной библиотекой для парсинга статичных данных. В частности, он утверждает: «Библиотека BeautifulSoup4 является отличным решением для использования в работе со статичным контентом, а также она очень проста в использовании. С ее помощью процесс написания кода для парсинга происходит гораздо быстрее, что является хорошим показателем для экономии времени» [11, с. 77]. Автор считает, что она помогает быстро и эффективно собирать информацию с веб-страниц, экономя время и ресурсы по сравнению с другими похожими инструментами. Именно поэтому мы выбрали этот модуль Python для работы.

2. Объекты и методы исследования

В данной статье основными объектами исследования являются веб-страницы, содержащие учебные материалы, и другие образовательные ресурсы. Также объектами исследования выступают программные инструменты и библиотеки, используемые для парсинга данных с этих веб-страниц, такие как BeautifulSoup4 и др.

Для достижения целей исследования были применены методы синтаксического анализа (парсинга), которые позволяют автоматически собирать, обрабатывать и структурировать информацию с сайтов.

3. Результаты и их обсуждение

В разработанном программном коде процесс обработки веб-страниц и сохранения данных в базу данных можно представить как последовательность шагов, выполняемых различными функциями, каждая из которых вносит свой вклад в обработку и структурирование дан-

ISSN 2221-1993 • **Информатика в школе** • 2025 • Том 24 № 6

```
def extract_section_from_filename(filename):
    match = re.search(r'test\d+\s*(.*?)\.html', filename)
    if match:
        return match.group(1).strip()
    return None
```

Листинг 1. Код функции extract_section_from_filename()

ных. Далее мы рассмотрим основные функции программы, необходимые для целостного понимания комплексного процесса парсинга, опуская элементы программы, реализованные для решения частных задач, актуальных только для конкретного обрабатываемого ресурса.

Начнем с анализа последовательности выполнения функций.

Программа начинается с извлечения раздела из имени файла, что необходимо для корректной идентификации источника данных. Функция *extract_section_from_filename*() принимает путь к файлу в качестве аргумента и на основе регулярного выражения выделяет название раздела (листинг 1). Этот этап важен для дальнейшего использования данных из разных разделов в одном проекте, позволяя программе классифицировать и сохранять задачи с учетом раздела.

Следующим логическим этапом является создание базы данных, которая выступает в роли хранилища для собранной информации. Функция *create_database*() принимает имя базы данных и проверяет, существует ли уже структура, соответствующая необходимой таблице (листинг 2). Если таблицы еще нет, функция создает ее с полями, необходимыми для хранения всех деталей задачи, включая текст задачи, изображения, решение и ответ. Создание базы данных на данном этапе упрощает процесс дальнейшего добавления записей.

После настройки хранилища следует этап определения темы задачи. Функция *find_theme*() исследует HTML-структуру страницы для извлечения соответствующего названия темы, которая может находиться в определенном CSS-классе или другом атрибуте (листинг 3). В данном случае — в теге <h3>. Это позволяет сгруппировать задачи по темам, что важно для последующего анализа и категориального хранения данных.

Следующий шаг — обработка текста задачи. Функция *extract_and_process_text*() извлекает текстовые данные и ссылки на изображения из заданного HTML-элемента (листинг 4). Она использует BeautifulSoup для парсинга содержимого и выделения текстовых фрагментов и ссылок на изображения.

```
def create_database(db_name):
    conn = sqlite3.connect(db_name)
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS tasks (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            section_number INTEGER,
            section TEXT,
            theme TEXT,
            number INTEGER,
            text TEXT,
            solution TEXT,
            answer TEXT,
            text_image TEXT,
            solution_image1 TEXT,
            solution_image2 TEXT,
            solution_image3 TEXT,
            text_image_svg TEXT,
            solution_image1_svg TEXT,
            solution_image2_svg TEXT,
            solution_image3_svg TEXT,
            is_converted BOOLEAN DEFAULT FALSE
        )
    ''')
    conn.commit()
    conn.close()
```

Листинг 2. Код функции create_database()

```
def find_theme(task):
    current_element = task
    while current_element:
        previous_element = current_element.find_previous("h3")
        if previous_element:
            return previous_element.text.strip()
        current_element = current_element.find_previous()
    return None
```

Листинг 3. Код функции find_theme()

```
def extract_and_process_text(element):
    text_parts = []
    images = []
    for child in element.descendants:
        if child.name == "img":
            src = child.get("src")
            if src and '/get_file?id=' in src:
                images.append(src)
            alt_text = child.get("alt")
            if alt_text:
                formula_text = convert_formula_to_text(alt_text)
                text_parts.append(formula_text)
        elif child.name is None:
            if child.string and not isinstance(child, BeautifulSoup):
                text_parts.append(child.string)
    combined_text = ''.join(text_parts)
    return remove_hyphens_and_shy(combined_text.strip()), images
```

Листинг 4. Код функции extract_and_process_text()

Чтобы работать с изображениями, в коде предусмотрена функция *process_image*(). Она берет ссылку, загружает картинку, сохраняет ее в формате SVG с кодировкой Base64, а затем конвертирует в PNG (листинг 5). Получившийся PNG-файл загружается на Telegraph, откуда мы получаем прямую ссылку на изображение, размещенное на удаленном сервере.

Функция *save_svg_to_png*() нужна для преобразования SVG в PNG (листинг 6). Сначала она берет содержимое SVG и сохраняет его во временный файл *temp.svg*. После чего запускается Inkscape, который конвертирует файл в PNG с нужной плотностью пикселей.

Код для загрузки в Telegraph использует API, предоставленный данным сервисом (функция

```
def process_image(url):
    image_content = fetch_image(url)
    if image_content:
        if url.endswith('.svg') or b'<svg' in image_content:
            png_content, modified_svg_content = save_svg_to_png(image_content)
            if png_content:
                png_url = upload_image_to_telegraph(png_content)
                svg_base64 = encode_image_to_base64(modified_svg_content, 'svg+xml')
                return svg_base64, png_url
        else:
            png_url = upload_image_to_telegraph(image_content)
            return None, png_url
    return None, None
```

Листинг 5. Код функции process_image()

```
def save_svg_to_png(svg_content):
    with open("temp.svg", "wb") as f:
        f.write(svg_content)

    subprocess.run(
        ["/Applications/Inkscape.app/Contents/MacOS/inkscape",
         "temp.svg",
         "--export-png=temp.png",
         "--export-dpi=288"]
    )

    if os.path.exists("temp.png"):
        with open("temp.png", "rb") as f:
            png_content = f.read()

        with open("temp.svg", "rb") as f:
            modified_svg_content = f.read()

        os.remove("temp.svg")
        os.remove("temp.png")

        return png_content, modified_svg_content

    return None, None
```

Листинг 6. Код функции save_svg_to_png()

ISSN 2221-1993 • **Информатика в школе** • 2025 • Том 24 № 6

```
def encode_image_to_base64(image_content, image_format):
    return f"" + base64.b64encode(image_content).decode('utf-8')

def upload_image_to_telegraph(image_content):
        url = 'https://telegra.ph/upload'

    while True:
        start_time = time.time()
        try:
            response = requests.post(url, files={'file': ('file', BytesIO(image_content), 'image/png')}, timeout=5)
            response.raise_for_status()  # Вызывает ошибку, если статус-код не в диапазоне 200-299
            json_data = response.json()
            if 'src' in json_data[0]:
                return 'https://telegra.ph' + json_data[0]['src']
            else:
                print("Error uploading image: no 'src' field in the response")
        except (requests.exceptions.RequestException, KeyError, IndexError) as e:
            print(f"Error uploading image: {e}")

        elapsed_time = time.time() - start_time
        if elapsed_time >= 5:
            print("Timeout exceeded. Restarting.")
```

Листинг 7. Код функций encode_image_to_base64() и upload_image_to_telegraph()

```
def add_task_to_database(db_name, section_number, section_name, task_theme,
task_number, task_text, task_solution, task_answer, text_image_png, text_image_svg,
*solution_images_data):
    conn = sqlite3.connect(db_name)
    cursor = conn.cursor()
    cursor.execute('''
        INSERT INTO tasks (section_number, section, theme, number, text, solution,
answer, text_image, solution_image1, solution_image2, solution_image3,
text_image_svg, solution_image1_svg, solution_image2_svg, solution_image3_svg)
        VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
    ''', (section_number, section_name, task_theme, task_number, task_text,
task_solution, task_answer, text_image_png, *solution_images_data[:3],
text_image_svg, *solution_images_data[3:]))
    conn.commit()
    conn.close()
```

Листинг 8. Код функции add_task_to_database()

	123 id	123 section_number	A-Z section	A-Z theme	123 number	A-Z text	A-Z solution	A-Z answer
1	1	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	1	На рисунке справа схема дорог Н-ского районa	Сопоставим населённые пункты гр	35
2	2	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	2	На рисунке слева изображена схема дорог Н-с	Сопоставим населённые пункты гр	67
3	3	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	3	На рисунке схема дорог изображена в виде гра	Заметим, что E — единственная ве	26
4	4	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	4	На рисунке слева изображена схема дорог Н-с	Заметим, что D — единственная ве	26
5	5	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	5	На рисунке слева изображена схема дорог N-с	Заметим, что A — единственная ве	45|П4ПБ
6	6	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	6	На рисунке слева изображена схема дорог N-с	Заметим, что B — единственная ве	46
7	7	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	7	На рисунке слева изображена схема дорог Н-с	Заметим, что вершины А и Д — еди	26
8	8	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	8	На рисунке слева изображена схема дорог Н-с	Заметим, что пункты К и Ж имеют	14
9	9	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	9	На рисунке схема дорог изображена в виде гра	Вершина Б — единственная верши	37
10	10	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	10	На рисунке схема дорог изображена в виде гра	Заметим, что Д — единственная ве	19
11	11	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	11	На рисунке схема дорог N-ского района изобр	Заметим, что A — единственная ве	22
12	12	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	12	На рисунке схема дорог N-ского района изобр	Заметим, что K — единственная ве	20
13	13	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	13	На рисунке схема дорог изображена в виде гра	Заметим, что вершина единствен	8
14	14	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	14	На рисунке схема дорог Н-ского района изобр	Вершина F — единственная верши	92
15	15	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	15	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется о	19
16	16	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	16	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется о	48
17	17	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	17	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется ч	19
18	18	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	18	На рисунке схема дорог изображена в виде гра	Заметим, что E — единственный п	ДАЕВГБЖИ
19	19	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	19	На рисунке справа схема дорог Н-ского районa	Заметим, что в таблице имеется 3	7
20	20	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	20	На рисунке схема дорог изображена в виде гра	Посчитаем количество дорог, вых	23
21	21	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	21	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется д	ИГВБЕДАЖ
22	22	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	22	На рисунке схема дорог N-ского района изобр	Заметим, что в таблице имеется о	25
23	23	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	23	Так как таблицу и схему рисовали независимо	Заметим, что вершины К и Ж — эт	17
24	24	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	24	На рисунке схема дорог Н-ского района изобр	Заметим, что в таблице имеется о	36
25	25	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	25	На рисунке схема дорог Н-ского района изобр	Заметим, что в таблице имеется о	34
26	26	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	26	На рисунке схема дорог Н-ского района изобр	Пункт С является единственным п	35
27	27	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	27	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется о	БЖКИДАЕГВ
28	28	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	28	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется т	14
29	29	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	29	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеется т	ДГВЖАБЕКИ
30	30	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	30	На рисунке схема дорог изображена в виде гра	Заметим, что в таблице имеются ч	78
31	31	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	31	На рисунке слева изображена схема дорог N-с	На схеме имеется один населенны	26
32	32	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	32	На рисунке схема дорог изображена в виде гра	На графе два населённых пункта с	ЕВЖИГДБАК
33	33	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	33	На рисунке схема дорог изображена в виде гра	Так как населенный пункт Д единс	16
34	34	1	Анализ информационных моделей	Неоднозначное соотнесение таблицы и графа	34	На рисунке схема дорог изображена в виде гра	На схеме только один населенный	22

Рис. Выборка первых строк созданной в результате работы парсера таблицы базы данных для хранения заданий ЕГЭ по информатике

upload_image_to_telegraph()). Хранение изображений осуществляется на бесплатной основе, что удобно для хранения большого количества изображений в сети. Кодировка SVG-изображения в Base64, которая осуществляется функцией *encode_image_to_base64*(), необходима для удобного хранения изображения в базе данных, а также может быть встроена в HTML для будущего веб-ресурса. Код этих функций представлен в листинге 7.

После сбора всех данных (текст задачи, решение, изображения) они сохраняются в базу, эту задачу выполняет функция *add_task_to_database*(), которая добавляет новую запись в таблицу (листинг 8).

Программа последовательно выполняет функции, которые читают HTML-файлы, извлекают из них задания и сохраняют полученные данные в базу. Результатом работы описанной программы является база данных, изображенная на рисунке, она содержит более 4000 заданий. Обработка такого количества заданий, включая конвертацию изображений и их загрузку на сервер, заняла у программы пять минут. Очевидно, что ручное выполнение этих действий в аналогичном объеме заняло бы несколько суток.

4. Заключение

Технология парсинга веб-страниц представляет собой мощный инструмент для создания методических материалов, который значительно упрощает и ускоряет процесс сбора и обработки информации. В условиях современного образовательного процесса, когда учителя и преподаватели сталкиваются с необходимостью быстро адаптировать учебные материалы под нужды обучающихся, парсинг становится незаменимым помощником.

Можно сделать вывод, что созданная нами программа — это полезный инструмент для учителей и преподавателей. Использование автоматизированных технологий в сборе данных значительно экономит время, что позволяет педагогам сосредоточиться на более важных аспектах своей работы — обучении и развитии учеников.

Список источников

1. 30+ парсеров для сбора данных с любого сайта // Хабр. https://habr.com/ru/company/click/blog/494020/

2. *Акимов П. М., Губанкова Е. В., Гузеев А. В., Сосновиков Г. К.* Исследование практического использования методов парсинга данных для разработки веб-служб на примере парсинга математических выражений // Телекоммуникации и информационные технологии. 2016. Т. 3. № 1. С. 111–114. EDN: XROBVN.

3. *Вильданов Т. Э., Иванов Н. С.* Анализ инструментов парсинга и веб-скрейпинга в рамках разработки арбитражной инвестиционной стратегии на рынке спортивных ставок // Мягкие измерения и вычисления. 2021. Т. 39. № 2. С. 65–77. EDN: EIGGSJ. DOI: 10.36871/2618-9976.2021.02.006.

4. *Ермоленко А. В., Котелина Н. О., Старцева Е. Н., Юркина М. Н.* О востребованности подготовки в области парсинга данных для web-разработчиков // Вестник Сыктывкарского университета. Серия 1. Математика. Механика. Информатика. 2021. № 1 (38). С. 56–69. EDN: DHZAEK. DOI: 10.34130/1992-2752_2021_1_56.

5. *Ие О. Н.* Возможности парсинга для получения данных о транспортном поведении пассажиров //Актуальные вопросы современной экономики. 2021. № 7. С. 329–339. EDN: HKLPLE. DOI: 10.34755/IROK.2021.38.45.003.

6. *Караваева А. А.* Инструменты web-scraping (парсинг) // Академическая публицистика. 2022. № 6-1. С. 99–103. EDN: IOSLDJ.

7. *Овсова М. С.* Анализ сложившейся в Российской Федерации и за рубежом судебной практики в части использования больших пользовательских данных // Молодой ученый. 2020. № 51 (341). С. 260–262. EDN: PVGHVI.

8. *Прокопенко В. В.* Парсинг как один из инструментов интеллектуальных баз данных // Аллея науки. 2020. Т. 2. № 6 (45). С. 68–75. EDN: HLNWHL.

9. *Скрыпников А. В., Денисенко В. В., Берестовой А. А.* Парсинг данных для мобильного приложения университета // Вестник Воронежского института ФСИН России. 2022. № 2. С. 126–132. EDN: OROSBW.

10. *Цхошвили Д. З., Иванова Н. А.* Примеры использования технологии парсинга // Актуальные вопросы в науке и практике. Сборник статей по материалам IV международной научно-практической конференции (Самара, 11 декабря 2017 года). В 5 ч. Ч. 1. Самара: Общество с ограниченной ответственностью Дендра, 2017. С. 135–138. EDN: YLGAZQ.

11. *Чирков А. Н.* Сравнительное исследование библиотеки BeautifulSoup4 и Selenium для парсинга веб-страниц при помощи Python // Научно-технические инновации и веб-технологии. 2022. № 1. С. 74–78. EDN: HHTJWM.