МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Комп'ютерна система виявлення плагіату

Освітній ступінь "бакалавр"
Спеціальність 123 – "комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	доц. Щербаков Є.В. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Обінський С.М. (ініціали, прізвище)
Група:		КІ-15з

Сєвєродонецьк 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітній ступінь бакалавр
Напрям підготовки _____
(шифр і назва)
Спеціальність 123 – комп'ютерна інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри КНІ
_____ І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Обінському Сергію Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Комп'ютерна система виявлення плагіату

керівник проекту (роботи) Щербаков Євген Васильович, к.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "13" 05 2019 р. № 84/15.15
2. Термін подання студентом роботи 16.06.2019
3. Вихідні дані до роботи Теоретичні відомості про плагіат та интиплагіат, методи розробки веб-застосунків на мові Python, відомості про принципи функціонування веб-краулерів та парсерів, принципи оцінки тексту на унікальність

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз предметної області «плагіат», програмна реалізація, комп'ютерна модель web-додатку, охорона праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст. викл. Критська Я.О.		

7. Дата видачі завдання 30.04.2019

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз завдання та робота з літературою	05.05.2019 - 13.05.2019	
2	Аналіз технічних засобів	14.05.2019 - 22.05.2019	
3	Розробка алгоритму	22.05.2019 - 02.06.2019	
4	Програмна реалізація	02.06 .2019- 11.06.2019	
5	Оформлення пояснювальної записки та	11.06.2019 - 16.06.2019	
	електронних плакатів		

Здобувач вищої освіти _____ Обінський С.М.
(підпис) (прізвище та ініціали)

Керівник _____ Рязанцев О.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка дипломної роботи: 71 с., 9 рис., 27 джерел.

Робота присвячена вирішенню проблеми плагіату. Розроблено методи пошуку в Інтернеті, які дозволяють знаходити веб-сторінки з відповідним до пошукового запиту змістом та методи текстового пошуку і аналізу, що дозволяють визначити ступінь унікальності тексту чи його частини.

Розроблені алгоритми реалізації запропонованих методів використано для створення спеціалізованих програмних засобів пошуку і аналізу тесту.

Ключові слова: АНТИПЛАГІАТ, ПОШУК В ІНТЕРНЕТІ, ПАРСИНГ, УНІКАЛЬНІСТЬ, WEB-СТОРІНКА.

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А,. м. Сєвєродонецьк, 93400с.

ЗМІСТ

ВСТУП

Останнім часом доступність інформації стає все більш важливою характеристикою сучасного суспільства. Проблема вдосконалення доступу до інформаційних ресурсів є усвідомленою на всіх рівнях їх формування та управління ними як у нас в країні, так і за кордоном. Дійсно, наука і освіта як ніякі інші сфери людської діяльності вимагають оперативної, своєчасної та достовірної інформації. Тому досить актуальна стала унікальність тієї чи іншої інформації.

Розвиток електронних інформаційних ресурсів сприяє підвищенню якості досліджень вчених і аналітиків за рахунок більшої доступності, прискорення пошуку необхідної інформації. Природно в цих умовах необхідно створювати нову інформацію, а не видавати чужу за свою. Якщо використаний чужий текст, то завжди потрібно давати посилання на джерело.

Плагіат - це оприлюднення (опублікування), повністю або частково, чужого твору під іменем особи, яка не є автором цього твору

Оприлюднення означає створення можливості ознайомлення з твором інших осіб шляхом опублікування, публічного виконання, публічного показу, публічної демонстрації, публічного сповіщення тощо.

Загалом за використання плагіату передбачено (залежно від кожного конкретного правопорушення) цивільну, адміністративну та кримінальну відповідальність.

Якщо зробити рерайтинг статті, курсової роботи її так, щоб зміст не змінилося, а змінилася лише структура, то спеціальні сервіси перевірки унікальності такі, як антиплагіат онлайн покажуть, що стаття унікальна. Для цього вони, в першу чергу і, призначені, щоб перевіряти матеріали для публікації.

Основне завдання анти плагіату – виявлення збіг деяких частин тексту для визначення ступеню їх унікальності на основі системи розпізнавання текстів та методів пошуку в Інтернеті.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ «ПЛАГІАТ»

1.1 Поняття плагіату

Для того, аби визначитись з тим, що таке плагіат, слід у першу чергу звернутися до чинного законодавства. Згідно ст. 50 Закону «Про авторське право та суміжні права», плагіат - це оприлюднення (опублікування), повністю або частково, чужого твору під іменем особи, яка не є автором цього твору.

Оприлюднення означає створення можливості ознайомлення з твором інших осіб шляхом опублікування, публічного виконання, публічного показу, публічної демонстрації, публічного сповіщення тощо.

На практиці виділяють такі види плагіату, як:

1) Копіювання чужої роботи (як без, так і з відома) та оприлюднення її під своїм іменем;

2) Представлення суміші власних та запозичених в інших аргументів без належного цитування джерел;

3) Перефразування чужої роботи без належно оформленого посилання на оригінального автора або видавця.

Також слід виділити такі види, як фальсифікація (вигадування тих чи інших, наприклад, статистичних показників з подальшим вказуванням їх у якості власної роботи), реплікація (це процес копіювання даних з одного джерела на багато інших і навпаки, тобто своєрідне «тиражування» інформації без дозволу автора), а також републікація (повторне або багаторазове обнародування в іншому джерелі чужої інформації за справжнім підписом автора й посиланням на джерело), рерайт (додавання до чужого матеріалу без дозволу автора додаткової інформацію, з переробкою раніше обнародуваного матеріалу і заміною слів та виразів) та компіляція (укладання з кількох чужих матеріалів свого, та редагування без дозволу – смислова, стилістична, граматична правка й скорочення чужого матеріалу).

Водночас, у ст. 10 Закону «Про авторське право та суміжні права» вказується, що не може підпадати під поняття плагіату самовільне оприлюднення таких об’єктів (оскільки на них не поширюється авторське право):

1) Повідомлень про новини дня або поточні події, що мають характер звичайної прес-інформації;

2) Творів народної творчості (фольклору);

3) Виданих органами державної влади у межах їх повноважень офіційних документів політичного, законодавчого, адміністративного характеру (законів, указів, постанов, судових рішень, державних стандартів тощо) та їх офіційних перекладів;

4) Державних символів України, державних нагород, символів та знаків органів державної влади, Збройних Сил України та інших військових формувань,символіки територіальних громад, символів та знаків підприємств, установ та організацій (після їх офіційного затвердження);

5) Грошових знаків;

6) Розкладів руху транспортних засобів, розкладів телерадіопередач, телефонних довідників та інших аналогічних баз даних, що не відповідають критеріям оригінальності і на які поширюється право sui-generis (своєрідне право, право особливого роду).

Також не є плагіатом опублікування анонімного твору під власним іменем, так як у цьому випадку на анонімний твір авторське право не поширюється.

1.2 Законодавча основа і відповідальність

Існуючий на сьогодні в Україні правовий механізм притягнення до відповідальності за плагіат є досить неоднозначним. Справа в тому, що з точки зору законодавства прописано достатньо відповідних норм, які дають визначення поняття плагіату гарантують захист прав авторів, а також

встановлюють перелік правопорушень з використанням плагіату та заходи відповідальності за них. Так статті 41 Конституції України, кожен має право володіти, користуватися і розпоряджатися своєю результатами своєї інтелектуальної, творчої діяльності. Стаття 54 Основного Закону встановлює заборону використовувати або поширювати об’єкти інтелектуальної власності без згоди їх автора, за винятками, встановленими законом.

У Цивільному Кодексі України захисту прав автора від плагіату приділено статті 433-456.

Загалом за використання плагіату передбачено (залежно від кожного конкретного правопорушення) цивільну, адміністративну та кримінальну відповідальність.

Адміністративна відповідальність у вигляді штрафу від десяти до двохсот неоподаткованих мінімумів доходів громадян (тобто від 170 до 2400 грн.) передбачена статтею 51-2 кодексу України про адміністративні правопорушення за незаконне використання об'єкта права інтелектуальної власності (літературного чи художнього твору, їх виконання, фонограми, передачі організації мовлення, комп'ютерної програми, бази даних, наукового відкриття, винаходу, корисної моделі, промислового зразка, знака для товарів і послуг, топографії інтегральної мікросхеми, раціоналізаторської пропозиції, сорту рослин тощо), привласнення авторства на такий об'єкт або інше умисне порушення прав на об'єкт права інтелектуальної власності, що охороняється законом.

Кримінальну відповідальність за подібного роду дії передбачено статтею 176 Кримінального кодексу України, а саме, штраф від двохсот до тисячі неоподаткованих мінімумів доходів громадян (від 2400 до 17000 грн.) або виправними роботами на строк до двох років, з конфіскацією всіх примірників творів, матеріальних носіїв комп'ютерних програм, баз даних, виконань, фонограм, програм мовлення та обладнання і матеріалів, призначених для їх виготовлення і відтворення. Такі заходи впливу застосовуються до особи, винної у незаконному відтворенні, розповсюдженні

творів науки, літератури, мистецтва, комп'ютерних програм і баз даних, а так само незаконному відтворенні, розповсюдженні виконань, фонограм і програм мовлення, їх незаконному тиражуванні та розповсюдженні на аудіо- та відеокасетах, дискетах, інших носіях інформації, а також інше використання чужих творів, комп'ютерних програм і баз даних, об'єктів суміжних прав без дозволу осіб, які мають авторське право або суміжні права, якщо ці дії завдали матеріальної шкоди у великому розмірі.

Примітка до цієї статті кодексу дає визначення поняття «матеріальна шкода завдана в значному розмірі»: її розмір у двадцять разів перевищує неоподаткований мінімум доходів громадян (тобто становить більше 1400 грн.). Також статтею передбачено більші розміри заходів відповідальності за вчинення таких дій повторно, або за попередньою змовою групою осіб, за завдання внаслідок таких дій шкоди у великому розмірі (яка у двісті і більше разів перевищує неоподаткований мінімум доходів громадян, тобто становить більше 3400 грн.), а також при вчиненні їх службовою особою з використанням нею службового становища. Або у випадку завдання шкоди в особливо великому розмірі (при перевищенні відповідного неоподаткованого мінімуму в тисячу і більше разів, тобто якщо шкода більше або дорівнює 17000 грн.). Найбільш суворим покаранням, передбаченим за вчинення даного злочину є позбавлення волі на строк від трьох до шести років з позбавленням на строк до трьох років права займати певні посади або займатися певною діяльністю та з конфіскацією і знищенням об’єктів і знарядь відповідного злочину.

Найчастіше у випадку використання плагіату винна особа притягується до цивільної відповідальності. Так, протягом періоду починаючи з 2006 року судами України було винесено 41 рішення про притягнення до цивільної відповідальності, тоді як про притягнення до кримінальної відповідальності за подібні дії було винесено лише 2 вироки, а про застосування адміністративної відповідальності – одна постанова.

В цивільному процесі у випадку доведення факту використання плагіату суд може винести рішення про притягнення винної особи до таких заходів відповідальності, як:

– відшкодування збитків та (або) моральної шкоди, завданих порушенням авторського права;

– стягнення доходу, отриманого в результаті відповідного порушення;

– виплату компенсації, що визначається судом у розмірі від 10 до 50000 мінімальних заробітних плат замість відшкодування збитків та стягнення доходу;

– конфіскацію примірників творів, виданих з порушенням авторського права.

Відповідні повноваження суду прописані у ст. 52 Закону «Про авторське право та суміжні права».

Також згідно ст. 432 ЦКУ, суд може винести рішення про:

– застосування негайних заходів щодо запобігання порушенню права інтелектуальної власності та збереження відповідних доказів;

– зупинення пропуску через митний кордон України товарів, імпорт чи експорт яких здійснюється з порушенням права інтелектуальної власності;

– вилучення з цивільного обороту товарів, виготовлених або введених у цивільний оборот з порушенням права інтелектуальної власності;

– вилучення з цивільного обороту матеріалів та знарядь, які використовувалися переважно для виготовлення товарів з порушенням права інтелектуальної власності;

– застосування разового грошового стягнення замість відшкодування збитків за неправомірне використання об'єкта права інтелектуальної власності. Розмір стягнення визначається відповідно до закону з урахуванням вини особи та інших обставин, що мають істотне значення;

– опублікування в засобах масової інформації відомостей про порушення права інтелектуальної власності та зміст судового рішення щодо такого порушення.

Водночас, незважаючи на досить непогану урегульованість питання відповідальності за плагіат на законодавчому рівні, на практиці коли постає необхідність захисту порушених прав автора, існують деякі проблеми. Насамперед пов’язані з пред’явленням доказів. Так, якщо авторське право було порушене при публікації плагіату на веб-сторінці, позивач може подати до суду як додаток до позову лише роздруківку такої сторінки (яка є копією сторінки, а отже має бути засвідчена нотаріально), окрім того автор не у всіх випадках може відслідкувати час опублікування плагіату в мережі Інтернет, його конкретних авторів (а не лише домен, на якому така сторінка розміщена).

І наостанок, хотілось би навести інформацію з інтернет-ресурсу «Вікіпедія» щодо того, як не допускати плагіату в процесі створення творів будь-якого жанру, статей тощо.

Цитування має використовуватися у всіх випадках, коли в роботі використовуються дані, взяті зі сторонніх джерел, а не отримані або створені безпосередньо автором. Порушення вказаних нижче правил і їх недотримання має розцінюватися як плагіат:

– якщо думка автора наводиться дослівно, то її слід взяти в лапки;

– якщо цитується великий уривок тексту, то він може не братися в лапки, натомість — виділяється або відбивається від решти тексту певним способом (набирається іншим кеглем, шрифтом, накресленням, відбивається від основного тексту більшими абзацними відступами тощо);

– допускається скорочення цитати, яке не веде до викривлення думки автора. Місце скорочення має бути відзначене в цитаті квадратними дужками з трикрапкою всередині;

– допускається перефразування цитати, зміна словоформ чи відмінків певних слів. В такому разі, цитата в лапки не береться, але в квадратних дужках обов'язково ставиться посилання на джерело (його порядковий номер зі списку використаної літератури, який додається до роботи);

– в списку використаної літератури завжди слід вказувати навіть ті джерела, які використовувалися під час підготовки роботи і вивчення теми, навіть якщо прямих посилань чи цитувань цих джерел в роботі немає.

1.3 Поняття анти плагіату

Антиплагіат - комп'ютерна програма, яка виявляє збіг деяких частин тексту. Самі по собі тексти можуть не бути плагіатом. Це можуть бути посилання, цитати і викладення і т. д. При написанні статті, автори можуть використовувати власний текст і різні джерела інформації. Багато авторів пишуть тексти на замовлення, і вони відповідальні за виконувану роботу. Неважливо, буде це стаття про модний бренд або стаття про будівництво, будь-який виконавець буде зацікавлений в унікальності своєї статті. Жоден клієнт не захоче мати на своєму сайті тексти, на які можуть пред'явити права конкуренти.

При купівлі або продажу своєї статті жодна людина не в змозі знайти будь-які копії в інтернеті з-за великої кількості інформації. Та й часу на це у нього не вистачить. Ось тут і знадобиться програма-антиплагіат, яка зможе чітко проаналізувати інформацію, перевірити статтю на унікальність. Така перевірка допоможе у статті уникнути копіювання фраз або мовних зворотів або просто повторень.

Метод антиплагіат визначає ступінь унікальності документів. База даних програми - це відкриті джерела інтернету, включаючи базу рефератів. Перевірка тексту починається з завантаження статті. Програма сприймає досліджуваний текст, як набір пропозицій. Кожне речення складається з набору слів. Під час перевірки програма порівнює кожне речення в тексті з пропозиціями, які знаходяться в базі даних системи. Якщо слова або їх порядок в реченні співпадуть зі словами в реченнях з бази даних, то це буде вважатися плагіатом.

Якщо програма антиплагіат не знайшла відповідників в цих пошукових системах, то текст, що перевіряється, є унікальним.

Якщо текст повністю або по шматках скачаний з інтернету, то програма вкаже посилання на всі першоджерела і відсоток унікальності тексту.

1.4 Огляд оснвних методів і засобів пошуку плагіату

Перевірка на плагіату тій чи іншій формі існувала завжди, але не була достатньо ефективною. З появою спеціалізованого програмного забезпечення із виявлення плагіату ця справа значно полегшилась, але автоматизована перевірка, значною мірою, є допоміжною для прийняття кінцевого рішення фахівцями-експертами. Одним із сучасних напрямів боротьби з академічним плагіатом є його виявлення і констатація за допомогою комп'ютерних програм. На сьогоднішній день існує небагато програмних засобів, які допоможуть встановити відсоток унікальності тексту. Кожне програмне забезпечення має свої особливості, переваги та недоліки, про що мова йтиме далі.

Сервіси для перевірки текстів на унікальність працюють, переважно, за однаковим алгоритмом. Документ транслюється текстовим редактором в текст у форматі .txt і перевіряється. Перевірка та пошук співпадінь виконується по шинглах (методом шингл – розбивки тексту).

Шингл – структурно-логічний фрагмент тексту, що складається з послідовності декількох слів. Пошук в Інтернеті здійснюється декількома пошуковими системами. В результаті візуалізується відсоток оригінальності тексту та список сайтів з відсотком збігу у відповідному кольорі в залежності від застосованих пошукових серверів. При виборі програм необхідно звертати увагу на їхню здатність повноцінно підтримували українську, російську та англійську мови, зручність та доступність для використання

1.5 Постановка задачі

Таким чином, визначення унікальності тексту та його частин є актуальним завданням. Тому ставиться завдання розробки методу пошуку веб-сторінок з відповідним змістом та знаходження однакових слів на цих сторінках.

Для цього необхідно вирішити такі завдання:

– провести аналіз існуючих методів пошуку в Інтренеті;

– розробити метод аналізу тексту;

– реалізувати метод визначення оцінки унікальності тексту на знайдених веб-сторінках ;

– реалізувати веб-застосунок для користування розробленими методами.

2 ПРОГРАМНА РЕАЛІЗАЦІЯ

2.1 Обґрунтування вибору програмного середовища

Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня з строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Серед основних її переваг можна назвати такі:

– чистий синтаксис (для виділення блоків слід використовувати відступи);

– переносність програм (що властиве більшості інтерпретованих мов);

– стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);

– можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);

– стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане на мові Python;

– зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);

– відкритий код (можливість редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Інтерпретатор мови Python і багата Стандартна бібліотека (як вихідні тексти, так і бінарні дистрибутиви для всіх основних операційних систем) можуть бути отримані з сайту Python www.python.org, і можуть вільно розповсюджуватися. Цей самий сайт має дистрибутиви та посилання на численні модулі, програми, утиліти та додаткову документацію.

Інтерпретатор мови Python може бути розширений функціями та типами даних, розробленими на C чи C++ (або на іншій мові, яку можна викликати із C). Python також зручна як мова розширення для прикладних програм, що потребують подальшого налагодження.

Python портований і працює майже на всіх відомих платформах — від КПК до мейнфреймів. Існують порти під Microsoft Windows, всі варіанти UNIX (включаючи FreeBSD та GNU/Linux), Plan 9, Mac OS та Mac OS X, iPhone OS 2.0 і вище, Palm OS, OS/2, Amiga, AS/400 та навіть OS/390, Symbian та Android.

У міру старіння платформи її підтримка в основній гілці мови припиняється. Наприклад, з версії 2.6 припинена підтримка Windows 95, Windows 98 та Windows ME.Однак на цих платформах можна використовувати попередні версії Python — спільнота активно підтримує версії Python починаючи від 2.3 (для них виходять виправлення).

При цьому, на відміну від багатьох портованих систем, для всіх основних платформ Python має підтримку характерних для даної платформи технологій (наприклад, Microsoft COM/DCOM). Навіть більше, існує спеціальна версія Python для віртуальної машини Java — Jython, що дозволяє інтерпретатору виконуватися на будь-якій системі, яка підтримує Java, при цьому класи Java можуть безпосередньо використовуватися з Python й навіть

бути написаними на ньому. Також кілька проектів забезпечують інтеграцію з платформою Microsoft.NET, основні з яких — IronPython та Python.Net.

Python підтримує динамічну типізацію, тобто, тип змінної визначається лише під час виконання. З базових типів слід зазначити підтримку цілих чисел довільної довжини і комплексних чисел. Python має багату бібліотеку для роботи з рядками, зокрема, кодованими в юнікоді.

З колекцій Python підтримує кортежі (tuples), списки (масиви), словники (асоціативні масиви) і від версії 2.4, множини.

Система класів підтримує множинне успадкування і метапрограмування. Будь-який тип, включаючи базові, входить до системи класів, й за необхідності можливе успадкування навіть від базових типів.

Подібно Ліспу та Прологу в режимі відлагодження, інтерпретатор Python має інтерактивний режим роботи, при якому введені з клавіатури вирази відразу ж виконуються, а результат виводиться на екран. Цей режим цікавий не тільки новачкам, але й досвідченим програмістам, які можуть протестувати в інтерактивному режимі будь-який фрагмент коду, перш ніж використовувати його в основній програмі, або просто використовувати як калькулятор з великим набором функцій.

В інтерактивному режимі доступний дебагер pdb та система довідки. Система допомоги працює для модулів, класів і функцій, тільки якщо ті були забезпечені рядками документації.

Крім вбудованої, існує й покращена інтерактивна оболонка IPython.

Дизайн мови Python побудований навколо об'єктно-орієнтованої моделі програмування. Реалізація ООП в Python є елегантною, потужною та добре продуманою, але разом з тим, достатньо специфічною в порівнянні з іншими об'єктно-орієнтованими мовами.

Можливості та особливості:

1) Класи є одночасно об'єктами з усіма нижче наведеними можливостями.

2) Успадкування, в тому числі множинне.

3) Поліморфізм (всі функції віртуальні).

4) Інкапсуляція (два рівні — загальнодоступні та приховані методи і поля). Особливість — приховані члени доступні для використання та помічені як приховані лише особливими іменами.

5) Спеціальні методи, що керують життєвим циклом об'єкта: конструктори, деструктори, розподільники пам'яті.

6) Перевантаження операторів (усіх, крім `is`, `'.'`, `'='` і символьних логічних).

7) Властивості (імітація поля за допомогою функцій).

8) Управління доступу до полів (емуляція полів і методів, частковий доступ тощо).

9) Методи для управління найпоширенішими операціями (істинносне значення, `len()`, глибоке копіювання, серіалізація, ітерація по об'єкту)

10) Метапрограмування (управління створенням класів, тригери на створення класів, та ін)

11) Повна інтроспекція.

12) Класові та статичні методи, класові поля.

13) Класи, вкладені у функції та інші класи.

Python підтримує парадигму функціонального програмування, зокрема:

– функція є об'єктом;

– функції вищих порядків;

– рекурсія;

– розвинена обробка списків (спискові вирази, операції над послідовностями, ітератори);

– аналог замикань (closures);

– часткове застосування функції;

– можливість реалізації інших засобів на самій мові (наприклад, каррінг).

Python поставляється «з батарейками в комплекті».

Багата стандартна бібліотека є однією з привабливих сторін Python. Тут є засоби для роботи з багатьма мережевими протоколами та форматами Інтернету, наприклад, модулі для написання HTTP-серверів та клієнтів, для розбору та створення поштових повідомлень, для роботи з XML, тощо. Набір модулів для роботи з операційною системою дозволяє писати крос-платформні застосунки. Існують модулі для роботи з регулярними виразами, текстовими кодуваннями, мультимедійними форматами, криптографічними протоколами, архівами, серіалізацією даних, юніт-тестуванням та ін.

Крім стандартної бібліотеки існує багато інших, що надають інтерфейс до всіх системних викликів на різних платформах; зокрема, на платформі Win32 підтримуються всі виклики Win32 API, а також COM в обсязі не меншому, ніж у Visual Basic або Delphi. Існує велика кількість прикладних бібліотек для Python у різноманітних галузях: веб-розробка, бази даних, обробка зображень, обробка тексту, чисельні методи, програми операційної системи тощо.

Для Python прийнята специфікація програмного інтерфейсу до баз даних DB-API 2 та розроблено відповідні цій специфікації пакети для доступу до різних СУБД: PostgreSQL, Oracle, Sybase, Firebird (Interbase), Informix, Microsoft SQL Server, MySQL та sqlite. На платформі Microsoft Windows доступ до БД можливий через ADO(ADOdb). Комерційний пакет mxODBC для доступу до СУБД через ODBC для платформ Windows і UNIX розроблений eGenix. Для Python написано багато ORM: (SQLObject, SQLAlchemy, Dejavu, Django), виконані програмні каркаси для розробки веб-застосунків (Django, Pylons).

Бібліотека NumPy для роботи з багатовимірними масивами дозволяє досягти продуктивності наукових розрахунків, порівнянної зі спеціалізованими пакетами. SciPувикористовує NumPy і надає доступ до великого спектру математичних методів (матрична алгебра — BLAS, level 1-3 і LAPACK; ШПФ).

Бібліотека WSGI - інтерфейс шлюзу з веб-сервером (Python Web Server Gateway Interface).

Python надає простий і зручний програмний інтерфейс C API для написання власних модулів на мовах C та C++. Інструмент SWIG дозволяє майже автоматично отримувати прив'язки для використання C/C++ бібліотек у коді на Python. Можливості цього та інших інструментів варіюються від автоматичної генерації (C/C++/Fortran)-Python інтерфейсів за спеціальними файлами (SWIG, pyste, SIP, pyfort) до надання зручніших API (boost::python, CXX та ін.) Інструмент стандартної бібліотеки ctypes дозволяє програмам Python безпосередньо викликати функції з динамічних бібліотек/DLL, написаних на C. Існують модулі, що дозволяють вбудовувати код на C/C++ прямо у вихідні файли Python, створюючи розширення «на льоту» (pyinline, weave). Для підключення математичних функцій, особливо із застосуванням NumPy, наразі офіційно рекомендованим є Cython

Інший підхід полягає у вбудовуванні інтерпретатора Python у застосунки. Python легко вбудовується в програми на Java, C/C++, Ocaml. Взаємодія Python-застосунків з іншими системами можлива також за допомогою CORBA, XML-RPC, SOAP, COM.

За допомогою Pyrex можлива компіляція Python-подібної мови (додано можливість типізації) в еквівалентний Сі-код і зв'язування із зовнішніми модулями [8].

Експериментальний проект shed skin передбачає створення компілятора для трансформації неявно типізованих Python програм в оптимізований C++ код. Починаючи з версії 0.22 shed skin дозволяє компілювати окремі функції в модулі розширень. Повна компіляція (станом на 1 липня 2007) далека від завершення.

Python та переважна більшість бібліотек до нього безкоштовні й поставляються у вихідних кодах. Навіть більше, на відміну від багатьох відкритих систем, ліцензія ніяк не обмежує використання Python у

комерційних розробках та не накладає ніяких зобов'язань, крім зазначення авторських прав.

З Python поставляється бібліотека tkinter на основі Tcl/Tk для створення крос-платформних програм з графічним інтерфейсом.

Для науково-технічної мети найбільшого поширення набуло використання matplotlib — бібліотеки з інтерфейсом, аналогічним MATLAB Plot Tool.

Існують розширення, що дозволяють використовувати всі основні GUI бібліотеки — wxPython, засноване на бібліотеці wxWidgets, PyGTK для GTK+, PyQt та PySide для Qt та інші. Деякі з них також надають широкі можливості для роботи з базами даних, графікою та мережами, використовуючи всі можливості бібліотеки, на якій базуються.

Для створення ігор та програм, що вимагають нестандартного інтерфейсу, можна використовувати бібліотеку Pygame. Вона також надає великі засоби роботи з мультимедіа: з її допомогою можна керувати звуком і зображеннями, відтворювати відео. Надаване pygame апаратне прискорення графіки OpenGL має більш високорівневий інтерфейс в порівнянні з PyOpenGL, що копіює семантику C-бібліотеки для OpenGL. Є також PyOgr, що забезпечує прив'язку до OGRE — високорівневої об'єктно-орієнтованої бібліотеки 3D-графіки. Крім того, існує бібліотека pythonOCC , що забезпечує прив'язку до середовища 3D-моделювання та симуляції OpenCascade.

Для роботи з растровою графікою використовується бібліотека Python Imaging Library [9].

2.2 Аналіз фреймворків

Flask - мікрофреймворк для веб-додатків, створений з використанням Python. Його основу складає інструментарій Werkzeug та рушій шаблонів Jinja2. Поширюється відповідно до умов ліцензії BSD.

Станом на грудень 2016 року стабільна версія Flask має номер 0.12. Flask використовується для розробки таких проектів як Pinterest, LinkedIn, а також сторінка спільноти Flask.

Flask називається мікрофреймворком, оскільки він не вимагає спеціальних засобів чи бібліотек. У ньому відсутній рівень абстракції для роботи з базою даних, перевірки форм або інші компоненти, які надають широковживані функції за допомогою сторонніх бібліотек. Однак, Flask має підтримку розширень, які забезпечують додаткові властивості таким чином, наче вони були доступні у Flask із самого початку. Існують розширення для встановлення об'єктно-реляційних зв'язків, перевірки форм, контролю процесу завантаження, підтримки різноманітних відкритих технологій аутентифікації та декількох поширених засобів для фреймворку. Розширення оновлюються частіше аніж базовий код.

Flask створено Арміном Ронакером у 2010 році в рамках діяльності проекту Pocoo.

"Все починалось як першоквітневий жарт, проте заживши великої слави, проект став по праву серйозним додатком."

"It came out of an April Fool's joke but proved popular enough to make into a serious application in its own right."

Flask базується на засобі Werkzeug WSGI а також рушієві шаблонів Jinja2, що їх було створено як проекти Pocoo у 2007 та 2008 роках відповідно, коли Ронакер та Ґеорґ Брандл створювали систему дошки оголошень на Python.

Незважаючи на відсутність головного релізу, Flask став надзвичайно популярним серед шанувальників Python. Станом на середину 2016 року, він був найбільш популярним веб-фреймворком Python на GitHub.

1) Містить сервер для розробки та відлагоджувач;
2) Вбудована підтримка юніт-тестів;
3) Управління запитами RESTful;
4) Використовує шаблони Jinja2;

5) Має підтримку безпечних куків (сесії на стороні клієнта);
6) 100% відповідність WSGI 1.0;
7) Підтримка Unicode;
8) Докладна документація;
9) Сумісність з Google App Engine;
10) Наявність розширень для забезпечення бажаної поведінки.

SQLite - полегшена реляційна система керування базами даних. Втілена у вигляді бібліотеки, де реалізовано багато зі стандарту SQL-92.. Сирцевий код SQLite поширюється як суспільне надбання, тобто може використовуватися без обмежень та безоплатно з будь-якою метою. Фінансову підтримку розробників SQLite здійснює спеціально створений консорціум, до якого входять такі компанії, як Adobe, Oracle, Mozilla, Nokia, Bentley і Bloomberg.

У 2005 році проект отримав нагороду Google-O'Reilly Open Source Awards.

Особливістю SQLite є те, що вона не використовує парадигму клієнт-сервер, тобто рушій SQLite не є окремим процесом, з яким взаємодіє застосунок, а надає бібліотеку, з якою програма компілюється і рушій стає складовою частиною програми. Таким чином, як протокол обміну використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому комп'ютері, на якому виконується застосунок. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції весь файл, що зберігає базу даних, блокується; ACID-функції досягаються зокрема за рахунок створення файлу-журналу.

Кілька процесів або потоків можуть одночасно без жодних проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, коли жодних інших запитів у цей час не обслуговується; інакше спроба запису закінчується невдачею, і в програму повертається код

помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

У комплекті постачання йде також функціональна клієнтська частина у вигляді виконуваного файлу sqlite3, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина працює з командного рядка, і дозволяє звертатися до файлу БД на основі типових функцій ОС.

Завдяки архітектурі рушія можливо використовувати SQlite як на вбудовуваних (embedded) системах, так і на виділених машинах з гігабайтними масивами даних.

Особливості:

1) Транзакції атомарні, послідовні, ізольовані, і міцні (ACID) навіть після збоїв системи і збоїв живлення.

2) Встановлення без конфігурації - не потребує ані установки, ані адміністрування.

3) Реалізує значну частину стандарту SQL92.

4) База даних зберігається в одному крос-платформовому файлі на диску.

5) Підтримка терабайтних розмірів баз даних і гігабайтного розміру рядків і BLOBів.

6) Малий розмір коду: менше ніж 350KB повністю налаштований, і менш 200KB з опущеними додатковими функціями.

7) Швидший за популярні рушії клієнт-серверних баз даних для найпоширеніших операцій.

8) Простий, легкий у використанні API.

9) Написана в ANSI C, включена прив'язка до TCL; доступні також прив'язки для десятків інших мов.

10) Добре прокоментований сирцевий код зі 100% тестовий покриттям гілок.

11) Доступний як єдиний файл сирцевого коду на ANSI C, який можна легко вставити в інший проект.

12) Автономність: немає зовнішніх залежностей.

13) Крос-платформовість: з коробки підтримується Unix (Linux і Mac OS X), OS/2, Windows (Win32 і WinCE). Легко переноситься на інші системи.

14) Сирці перебувають в суспільному надбанні.

15) Поставляється з автономним клієнтом інтерфейсу командного рядка, який може бути використаний для управління базами даних SQLite.

Створення та обслуговування БД можуть здійснюватись через текстову консоль SQL-командами або через спеціальні інструменти, у тому числі — з графічним інтерфейсом користувача.

Сама бібліотека SQLite написана мовою C; існує велика кількість прив'язок до інших мов програмування, зокрема до C++, Java, Python, Perl, PHP, Tcl (засоби для роботи з Tcl включені в комплект постачання SQLite), Ruby, Haskell, Scheme, Smalltalk і Lua, а також до багатьох інших. Повний список наявних засобів можна знайти на сторінці проекту.

У ряді інструментаріїв присутня можливість використання SQLite як бази даних, наприклад:

– Django;

– Java;

– PHP;

– Ruby on Rails;

– Trac;

– Symfony;

– Parser ;

– 1C.

Багато програм підтримують SQLite як формат зберігання даних, зокрема:

– Amarok (може використовувати бази даних SQLite як сховище музичної колекції);

– Gajim — SQLite використовується для зберігання історії контактів;

– Songbird (як застосунок, заснований на XULRunner 1.9);

– Banshee;

– F-Spot;

– платформа XUL на рушії Gecko 1.9, XULRunner 1.9 і, потенційно, всі застосунки, засновані на цій платформі, у тому числі і Firefox від версії 3.0;

– Google Chrome;

– Google Gears;

– Mendeley — менеджер паперів-pdf, академічний засіб для дослідження (реалізується desktop & web);

– Zotero — менеджер інформації, бібліографічний менеджер, программа Firefox.

SQLAlchemy - інструментарій SQL та об'єктно-реляційне відображення для мови програмування Python випущене під ліцензією MIT.

SQLAlchemy надає "повний набір добре відомих шаблонів корпоративного рівня стабільності, сконструйованих для високопродуктивного доступу до бази даних, написаних простою мовою Python". Філософія SQLAlchemy стверджує що бази даних SQL поводяться тим менш подібно на колекції об'єктів чим більше починають важити розмір та продуктивність, і навпаки, колекції об'єктів починають поводитись тим менш подібно на таблиці і записи чим більш починає важити рівень абстракції. Тому, було впроваджено шаблон Data Mapper (подібний на Hibernate для Java) замість шаблону active Record який використовується в багатьох інших об'єктно-реляційних відображеннях. Проте, додаткові плагіни, такі як Elixir та declarativeдозволяють користувачам розробку з декларативним синтаксисом.

SQLAlchemy була вперше випущена в лютому 2006-го, і швидко стала однією з найширше використовуваних в спільноті Python бібліотек для об'єктно-реляційного відображення.

2.3 Організація краулінгу і парсингу

Краулінг - це процес сканування сайту пошуковою системою. Сканування – початковий етап, дані збираються для подальшої внутрішньої обробки(побудова індексів) та не відображаються в результатах пошуку. Просканована сторінка не обов’язково є проіндексованою. У пошукової системи ресурси обмежені, а методи краулінгу допоможуть оптимізувати процес: щоб для кожного сайту виділялася необхідна кількість «потужності», щоб успішно його індексувати.

– швидка індексація;

– швидка переіндексація (якщо відбулися зміни з документом);

– якісні індекси (щоб в індекси потрапляли лише якісні документи, не потрапляв малоінформативний контент) [7].

Краулінговий бюджет потрібен лише для сайтів з великим об’ємом інформації (від 100 сторінок). Адже маленькі сайти пошукова система проіндексує за відносно малу кількість часу (неділя, місяць). Також краулінг використовується для сайтів, що часто змінюються.

Пошуковий робот (crawler, краулер, павук, бот) – програма для збору контенту в інтернеті. Пошуковий робот складається з безлічі комп’ютерів, що обирають сторінки швидше, ніж користувач за допомогою свого веб-браузеру. Фактично він може обробляти тисячі різних сторінок одночасно.

Принцип роботи краулера:

1) Максимальне охоплення мережі;

2) Економія серверних ресурсів;

3) Не сканувати те, що закрито;

4) Оцінка корисності документу ще до його відкриття(авторитетність сторінки, рівень на URL-сторінці(кількість слешів), і т.і.);

5) Оцінка корисності сайту після перших сканувань(уникнення сайтів с дублюванням, якісний контент).

Що впливає на краулінговий бюджет:

1) Швидкість віддачі, розмір документу;

2) Об’єм сайту;

3) Якість контенту(недопустима наявність малоінформативних сторінок);

4) Коди статусів (якщо не 200/304);

5) Відвідуваність сайту;

6) Виділення IP-адреси;

7) Популярність посилань(кількість, авторитетність посилань необхідні для пришвидшення індексації).

Для того, щоб подивитися на сайт очима краулера, потрібно вимкнути обробку JavaScript

Є декілька способів

1) Вимкнути через консоль розробника(F12);

2) Використання інструменту «Подивитися як Googlebot»;

3) Спеціальне програмне забезпечення (http://pr-cy.ru/simulator/, https://netpeaksoftware.com/ru/spider та інші).

Crawljax: Crawling Ajax-based Web Applications - це Java-інструмент з відкритим кодом, що дозволить протестувати ваш web – застосунок фактично імітуючи користувача по браузингу сайту. Crawljax може досліджувати сайт, що використовує технологію ajax, при цьому автоматично створюючи динамічний граф станів DOM.

В основу Crawljax покладено дослідження 2007 р. Алі Мешбаха та Арі Ван Дрьосена. Основна ідея була закладена в їх спільній праці «Exposing the Hidden-WebInduced by Ajax», в якій вони показали як динамічний сайт, що використовує технологію ajax, може бути представлений у вигляді графа статичних станів DOM та переходів між ними.

Пізніше ця робота використовувалася для створення методів для пошукових систем, що давали б змогу їм краулити та аналізувати зміст динамічних web-додатків.

В кінці роботи Crawljax формує html-репорт, що містить граф станів та переходів по сайту, статистику щодо своєї роботи, список відвідуваних url-ів та детальну інформацію щодо кожного стану, в який може переходити DOM.

Також використовують платформу Node.js.

Пошуковий робот – це найважливіший елемент пошукової системи, в завдання якого входить збір нових даних про сайти та їх оновлення. Пошуковий робот являє собою програму, яка діє приблизно так, як і браузерна програма – зчитує інформацію з веб-сторінок. Пошуковий робот, бот, краулер, пошуковий павук, web crawler, ant, automatic indexer, bot, web spider, web robots, web scutter – це все назви одного й того ж явища, які можуть зустрічатися в англомовному і україномовному інтернеті. Пошукова система може мати не один, а кілька пошукових роботів. Кожен бот являє собою автоматичний скрипт, що має свій метод роботи, своє конкретне завдання для певного сайту. Бот як корабель-дослідник

Щоб уявити собі механізм роботи робота, скористаємося художнім образом. Уявіть океан, в якому існують архіпелаги островів. Частина цих островів відкрита, вивчена, нанесена на карту. Частина ще не відкрита. Частина щойно з’явилася, наприклад, в результаті вулканічної діяльності. Корабель-дослідник (або декілька кораблів) заходять на острови, а потім інформація заноситься на карту. Ось на карті з’явився новий острів. Ось старий острів, на на ньому збудований місто. А цей острів зник, пішов під воду. Так само як корабель-дослідник, бот методично досліджує інтернет у пошуках нових сайтів, нових сторінок, нових файлів, зчитує, заносить їх до реєстру пошукової машини, тобто індексує. Для чого це потрібно пошуковій системі? Для того, щоб вона могла видати на запит найточнішу відповідь, що відповідає картині даних на самий останній момент. Для чого це потрібно сайту? Для того, щоб потрапити у видачу, тобто для того, щоб на пошуковий запит, пов’язаний з ним, система у своїй відповіді зазначила б саме цей сайт. Для чого це потрібно користувачеві? Для отримання правильної адекватної відповіді на своє питання.

Отже, в морі інтернету з’явився новий острів – новий сайт. Як довго він буде залишатися в безвісті, навіть якщо містить необхідну інформацію? Як багато часу потрібно веб-павуку, щоб дістатися до нього і занести в свій список? Як часто краулер буде помічати зміни, які відбуваються на сайті? Чи вся інформація доступна павуку? Що робити, щоб в пошук не потрапляли певні сторінки та файли? Павукова діяльність так влаштована, що рано чи пізно сайт буде помічений і проіндексований. Однак, це може зайняти кілька місяців. Щоб пошуковий робот швидше помітив його, потрібно внести сайт в спеціальні списки-каталоги, що існують при пошукових системах. Мова в першу чергу йде про такі пошукові Гіганти, як Google і Яндекс. Раз проіндексувавши сайт, бот буде регулярно туди заходити. Однак частота його відвідувань безпосередньо пов’язана з частотою оновлення сайтів.

Помітивши, що сайт оновлюється приблизно раз на тиждень, бот заходить туди приблизно стільки ж, відповідно, нова веб-сторінка сайту може залишатися непоміченою кілька днів. І навпаки: існують рухливі блоги, які додають записи по кілька разів на день. Відповідно, робот контролює їх дуже часто й нові сторінки індексуються вже через кілька хвилин. Діяльність робота визначається заданим пошуковим методом, система методів гнучка і змінюється. Завдання та обмеження Як вже було сказано, система володіє великою кількістю різних роботів, які виконують різні завдання: одні шукають нові сторінки, інші відповідають за знаходження “мертвих” сайтів і чистку пошукових даних, треті індексують картинки, четверті – знаходять відео. Є робот, який відповідає за перевірку коректності посилань і робот, який читає виключно коментарі. Для робота одне з найважливіших значень має файл robots.txt, розташований на підконтрольному сервері. Зайшовши на будь-який сайт, робот звертається в першу чергу до нього. Цей файл – інструкція для робота. По-перше, robots.txt може взагалі не допустити бота на сайт і сайт залишиться не проіндексованим. По-друге, robots.txt може закрити боту доступ до певних сторінок і файлів.

Кожна пошукова система має свого власного бота, при цьому пошуковий робот Google може значно відрізнятися за механізмом роботи від аналогічної програми "Яндекса" або інших систем.

У загальних рисах принцип роботи робота полягає в наступному: програма «приходить» на сайт по зовнішніх посиланнях і, починаючи з головної сторінки, «читає» веб-ресурс (в тому числі переглядаючи ті службові дані, які не бачить користувач). Бот може переміщатися між сторінками одного сайту, так і переходити на інші. Як програма вибирає, який індексувати сайт? Найчастіше «подорож» павука починається з новинних сайтів або великих ресурсів, каталогів і агрегаторів з великою посилальної масою. Пошуковий робот безперервно сканує сторінки одну за одною, на швидкість і послідовність індексації впливають наступні фактори:

– внутрішні: перелиновка (внутрішні посилання між сторінками одного і того ж ресурсу), розмір сайту, правильність коду, зручність для користувачів і так далі;

– зовнішні: загальний обсяг посилальної маси, яка веде на сайт.

Першим ділом пошуковий робот шукає на будь-якому сайті файл robots.txt. Подальша індексація ресурсу проводиться, грунтуючись на інформації, одержаної саме від цього документа. Файл містить точні інструкції для "павуків", що дозволяє підвищити шанси відвідування сторінки пошуковими роботами, а отже, і домогтися якнайшвидшого попадання сайту в видачі "Яндекса" і Google;

Часто поняття «пошуковий робот» плутають з інтелектуальними, користувацькими або автономними агентами, "мурахами" або "хробаками". Значні відмінності є тільки порівняно з агентами, інші визначення позначають схожі види роботів.

Так, агенти можуть бути:

– інтелектуальними : програми, які переміщуються від сайту до сайту, самостійно вирішуючи, як діяти далі; вони мало поширені в інтернеті;

– автономними: такі агенти допомагають користувачеві у виборі продукту, пошуку або заповненні форм, це так звані фільтри, які мало відносяться до мережевих програм;

– користувацькими: програми сприяють взаємодії користувача з Всесвітньою павутиною, це браузери (наприклад, Opera, IE, Google Chrome, Firefox), месенджери (Viber, Telegram) або поштові програми (MS Outlook або Qualcomm)."Мурашки" і "черв'яки" більше схожі з пошуковими "павуками". Перші утворюють між собою мережу і злагоджено взаємодіють подібно до справжньої мурашиної колонії, черв'яки здатні самовідтворюватися, в іншому діють так само, як і стандартний пошуковий робот.

Розрізняють безліч різновидів пошукових роботів. Залежно від призначення вони бувають:

1) «Дзеркальними» - переглядають дублікати сайтів.

2) Мобільними – націлені на мобільні версії інтернет-сторінок.

3) Швидкодіючими – фіксують нову інформацію оперативно, переглядаючи останні оновлення.

4) Ссилочними – індексують посилання, підраховують їх кількість.

5) Индексаторами різних типів вмісту окремих програм для тексту, аудіо та відео, зображень.

6) «Шпигунськими» - шукають сторінки, які ще не відображаються в пошуковій системі.

7) «Дятлами» - періодично відвідують сайти, щоб перевірити їх актуальність і працездатність.

8) Національними – переглядають веб-ресурси, розташовані на доменах однієї країни (наприклад, .ru, .kz або .ua).

9) Глобальними – індексують всі національні сайти.

Існують також окремі роботи пошукових систем. В теорії їх функціональність може значно різнитися, але на практиці програми

практично ідентичні. Основні відмінності індексації інтернет-сторінок роботами двох основних пошукових систем полягають у наступному:

– строгість перевірки. Вважається, що механізм пошукового робота "Яндекса" дещо суворіше оцінює сайт на відповідність стандартам Всесвітньої павутини;

– збереження цілісності сайту. Пошуковий робот Google індексує сайт цілком (в тому числі медіаконтент), "Яндекс" може переглядати сторінки вибірково;

– швидкість перевірки нових сторінок. Google додає новий ресурс в пошукову видачу протягом декількох днів, у випадку з "Яндексом" процес може розтягнутися на два тижні і більше;

– частота переіндексації. Пошуковий робот "Яндекса" перевіряє наявність оновлень пару раз в тиждень, а Google – один раз в 14 днів.

Інтернет, звичайно ж, не обмежується двома пошуковими системами. Інші пошукачі мають своїх роботів, які слідують власним параметрами індексації. Крім того, існує кілька "павуків", які розроблені не великими пошуковими ресурсами, а окремими командами або веб-майстрами.

Всупереч поширеній думці, "павуки" не обробляють отриману інформацію. Програма сканує тільки і зберігає веб-сторінки, а подальшою обробкою займаються зовсім інші роботи. Також багато користувачів вважають, що пошукові роботи чинять негативний вплив і «шкідливі» інтернету. Дійсно, окремі версії "павуків" можуть значно перевантажувати сервера. Має місце і людський фактор – веб-майстер, який створював програму, може допускати помилки в налаштуваннях робота. Все ж більшість діючих програм добре спроектовані і професійно управляються, а будь-які виникаючі проблеми оперативно усуваються.

Пошукові роботи є автоматичними програмами, але процес індексації може частково контролюватися веб-майстром. У цьому значно допомагає зовнішня і внутрішня оптимізація ресурсу. Крім того, можна вручну додати

новий сайт в пошукову систему: великі ресурси мають спеціальні форми реєстрації веб-сторінок.

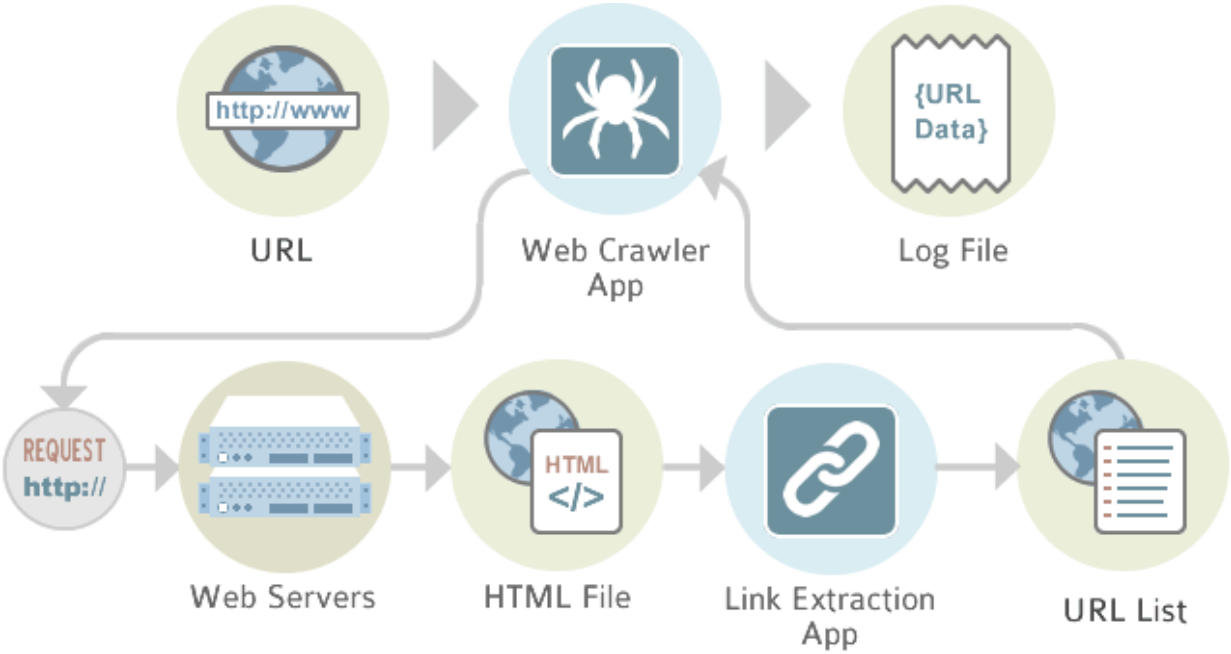


Рисунок 2.1 – Організація пошукового роботу

Синтаксичний аналіз - в інформатиці це процес аналізу вхідної послідовності символів, з метою розбору граматичної структури згідно із заданою формальною граматикою. Синтаксичний аналізатор - це програма або частина програми, яка виконує синтаксичний аналіз.

Під час синтаксичного аналізу текст оформлюється у структуру даних, зазвичай - в дерево, яке відповідає синтаксичній структурі вхідної послідовності, і добре підходить для подальшої обробки. Зазвичай синтаксичні аналізатори працюють в два етапи: на першому ідентифікуються осмислені токени (виконується лексичний аналіз), на другому створюється дерево розбору.

Під час синтаксичного аналізу текст оформлюється у структуру даних, зазвичай - в дерево, яке відповідає синтаксичній структурі вхідної послідовності, і добре підходить для подальшої обробки. Зазвичай синтаксичні аналізатори працюють в два етапи: на першому ідентифікуються

осмислені токени (виконується лексичний аналіз), на другому створюється дерево розбору.

Синтаксичний аналізатор - це програмний компонент, який приймає вхідні дані (часто текст) і створює структуру даних - часто дерево розбору, абстрактне дерево синтаксису або іншу ієрархічну структуру - забезпечує структурне представлення вводу, перевіряє правильність синтаксису в процесі. Для аналізу можуть передувати або слідувати інші кроки, або їх можна об'єднати в один крок. Аналізатору часто передує окремий лексичний аналізатор, який створює токени з послідовності введених символів; Крім того, їх можна об'єднати у парсінг без сканування. Аналізатори можуть бути запрограмовані вручну або автоматично або напів автоматично генератором парсерів. Розбір допомагає шаблону, який виробляє відформатований вихід. Вони можуть використовуватись у різних ділянках, але часто з'являються разом, наприклад, пара scanf/printf, або як вхідний (аналіз вхідних даних) та вихідний етапи (створення кінцевого коду) компілятора.

Вхідними даними для синтаксичного аналізатора часто є текст на деякій комп'ютерній мові, але також може бути текстом природною мовою або менш структурованими текстовими даними, в цьому випадку, як правило, витягуються лише окремі частини тексту, а не дерево розбору. Параметри відрізняються від дуже простих функцій, таких як scanf, до складних програм, таких як інтерфейс компілятора C++ або HTML-аналізатор веб-браузера. Важливий клас простий синтаксичний аналіз виконується за допомогою регулярних виразів, в яких група регулярних виразів визначає регулярну мову та двигун регулярного виразу, автоматично генеруючи аналізатор для цієї мови, що дозволяє узгодити шаблон та вилучення тексту. В інших контекстах регулярні вирази замість цього використовуються перед розбором, як етап лексизації, вихід якого потім використовується аналізатором.

Використання аналізаторів залежить від вхідних даних. У випадку з мовами даних часто використовується синтаксичний аналізатор як функція

читання файлів у програмі, наприклад, читання в HTML або XML-тексті; ці приклади є мовами розмітки даних. У випадку мов програмування є компонентом компілятора або інтерпретатора, який аналізує початковий код мови комп'ютерного програмування для створення певної форми внутрішнього представлення; аналізатор є ключовим кроком в інтерфейсі компілятора. Мови програмування, як правило, вказуються в термінах детерміністичної контекстно-вільної граматики, оскільки для них можуть бути написані швидкі та ефективні аналізатори. Для компіляторів сам аналіз може бути виконаний за один прохід або кілька проходів.

Майбутні недоліки компілятора з одним прохідним процесом у значній мірі можуть бути вирішені шляхом додавання виправлень, коли передбачається виправлення впродовж прямого переходу, а виправлення застосовуються в зворотньому напрямку, коли поточний сегмент програми є таким, що має бути завершений. Приклад, коли такий механізм виправлення може бути корисним, буде формальним твердженням GOTO, де ціль GOTO невідома, доки не буде пройдено сегмент програми. У такому випадку застосування виправлення буде відкладено, доки не буде визначено куди вказує GOTO. Очевидно, що відсталий GOTO не вимагає виправлення.

Контекстні граматики обмежені в тій мірі, в якій вони можуть виразити всі вимоги до мови. Неформально, причиною є те, що пам'ять в такій мові обмежена. Граматика не запам'ятовує наявність конструкції над довільним введенням; це необхідно для мови, в якій, наприклад, ім'я повинно бути оголошено, перш ніж може бути посилання на нього. Однак більш потужні граматики, які можуть обійти це обмеження, не можуть бути ефективно розібрані. Таким чином, загальною стратегією є створення аналізатора для контекстно-вільної граматики, який приймає потрібні конструкції мови (тобто він приймає деякі недійсні конструкції); пізніше, небажані конструкції можуть бути відфільтровані на етапі семантичного аналізу (контекстного аналізу).

Замість того, щоб аналізувати на етапі парсинга, це відбувається шляхом перевірки значень в дереві синтаксису, отже, як частина семантичного аналізу: контекстно-залежний синтаксис на практиці часто більш легко аналізується ніж семантика.

Наступний приклад демонструє загальний випадок розбору комп'ютерної мови з двома рівнями граматики: лексичної та синтаксичної.

Перший етап - генерація токенів або лексичний аналіз, за допомогою яких потік вхідних символів поділяється на значущі символи, визначені граматикою регулярні вирази. Наприклад, програма калькулятора буде отримувати на вхід, такий рядок "12 * (3 + 4)^2" і розділити його на токени 12, *, (, 3, +, 4,), ^, 2, кожен з яких є значущим символом в контексті арифметичного виразу. Лексери міститимуть правила, які вказують на те, що символи *, +, ^, (і) позначають початок нового токену, так що незначні токени типу"12*" або "(3" не будуть створені).

Наступним етапом є парсинг чи синтаксичний аналіз, який перевіряє, чи токени утворюють допустимий вираз. Це, як правило, робиться з посиланням на контекстну граматику, яка рекурсивно визначає компоненти, які можуть скласти вираз та порядок їх появи. Проте не всі правила, що визначають мови програмування, можуть бути виражені контекстно-вільними граматиками, наприклад, дійсність типу та правильне декларування ідентифікаторів. Ці правила можуть бути формально виражені атрибутивними граматики.

Заключна фаза - це семантичний аналіз або парсинг, який розробляє наслідки тільки що підтвердженого вираження та прийняття відповідних заходів. У випадку калькулятора чи інтерпретатора, дія полягає в оцінці виразу або програми, а компілятор, з іншого боку, генерує якийсь код. Атриматичні граматики також можуть бути використані для визначення цих дій [12].

Завдання аналізатора по суті полягає в тому, щоб визначити, як вхід можна отримати з початкового символу граматики. Це можна зробити по суті двома способами:

– низхідний синтаксичний аналіз - аналіз зверху вниз можна розглядати як спробу знайти найбільший початок z слова x, здатний бути початком виводжуваного з K слова, шляхом пошуку в синтаксичному дереві за допомогою розгорнення зверху вниз заданих формальних правил граматики. Токени читаються зліва направо. Інклюзивний вибір використовується для задоволення багатозначності шляхом розширення всіх альтернативних правих правил граматики;

– синтаксичний аналіз знизу - Синтаксичний аналізатор може починатися з вводу та спробувати переписати його на символ початку. Інтуїтивно, синтаксичний аналізатор намагається знайти найбільш основні елементи, потім елементи, що містять їх, і так далі. Аналізатор LR є прикладами аналізаторів знизу вгору. Іншим терміном, що використовують для цього типу аналізатора, є Shift-Reduce, синтаксичний аналіз.

LL-аналізатор та рекурсивний спуск є прикладами аналізаторами згори-вниз, які не можуть враховувати ліву рекурсію. Хоча вважалося, що прості реалізації синтаксичного аналізу зверху вниз не можуть враховувати пряму та непряму ліву рекурсію і можуть вимагати експоненціального часу та просторово складності під час аналізу неоднозначних контекстно-вільних граматик, складніші методи для розбору зверху вниз створюються Frost, Hafiz, і Callaghan, які враховують багатозначність і ліва рекурсія в поліноміальному часі і які генерують поліноміальне представлення потенційно експоненціального числа дерев розбору. Їх метод здатний виготовити як ліва, так і правомірне похідні вхідних даних щодо даної контекстно-вільної граматики.

Парсинг - це зіставлення рядки природної мови або мови програмування з формальними правилами, це інструмент роботи із строковими даними. Наведу приклад. Уявіть себе радистом на війні. Ви

отримуєте зашифроване повідомлення. У вас є правила дешифрування. Ви починаєте розгадувати послання за цим методом.

Ви дивіться спочатку на символ з отриманого повідомлення. Потім на свою таблицю з його значенням. Наприклад, цифрі "1" відповідає буква "Я". Ви зіставляєте все символи і отримуєте те повідомлення, яке можна прочитати.

Парсинг працює точно так само. Є деякий шаблон повідомлення, написаний на формальній мові. З ним порівнюється якась рядок. Парсинг застосовується в програмуванні, в аналітиці. Може бути корисний в будь-якій області, де є можливість роботи із строковими даними.

У загальному випадку, парсинг будує шаблон послідовності символів. Наприклад, може використовуватися деревоподібна структура. Вона показує, в якій послідовності в рядку зустрічаються символи. Може вказувати на пріоритет, якщо мова йде про математичному вираженні [11].

Такі структури потрібні для аналізу даних.

Парсити можна і інтернет-ресурси. Це роблять, коли потрібно зрозуміти, який контент міститься на сторінці.

Знайти на сторінках сайту тільки ту інформацію, яка потрібна вам для аналізу - це завдання парсинга.

Скрипт парсинга працює з текстовою інформацією. Він витягає потрібні дані, представляє їх у зручному вигляді.

Наприклад, ви - власник інтернет-магазину. І ви хочете швидко зібрати дані про інших магазинах - ваших конкурентів. Вас цікавить інформація з карток товарів. Ви хочете зрозуміти, як їх заповнюють конкуренти, що вони роблять краще вас. Ви визначаєте, інформація з яких сайтів вам потрібна. Вибираєте програму або скрипт, якими будете парсити текст. Запускаєте. Програма в одному файлі може зібрати інформацію.

Наприклад, назва, ціну на товар, категорію і опис. Далі ви вже зможете проаналізувати це. Наприклад, вирішити, яку ціну встановити для свого асортименту.

А може, вам потрібно попрацювати з відгуками клієнтів? Це теж завдання для парсинга сайту - збираєте потрібну інформацію в одному місці і читаєте, що про вашого конкурента пишуть клієнти.

Етапи парсинга даних:

1) Збір контенту. Зазвичай в програму для парсинга завантажується код сторінки сайту. І з ним уже працює спеціальний скрипт - розбиває весь код на лексеми, аналізує, яка інформація потрібна користувачеві.

2) Витяг інформації. Користувачеві не потрібна вся інформація зі сторінки. Повернемося до прикладу вище. Нас цікавлять тільки відгуки клієнтів під конкретними товарами - наприклад, кормом для кішок. Парсер знаходитиме в коді сторінки то місце, де вказана категорія товару: "Корм для кішок". Далі він визначить те місце на сторінці, де розміщені коментарі. І витягне в кінцевий файл тільки тексти коментарів.

3) Збереження результатів. Коли вся потрібна інформація витягнута з сайтів, потрібно її зберегти. Зазвичай такі дані оформляють у вигляді таблиць, щоб було наочне уявлення. Можна вносити записи в базу даних. Як буде зручніше аналітику [10].

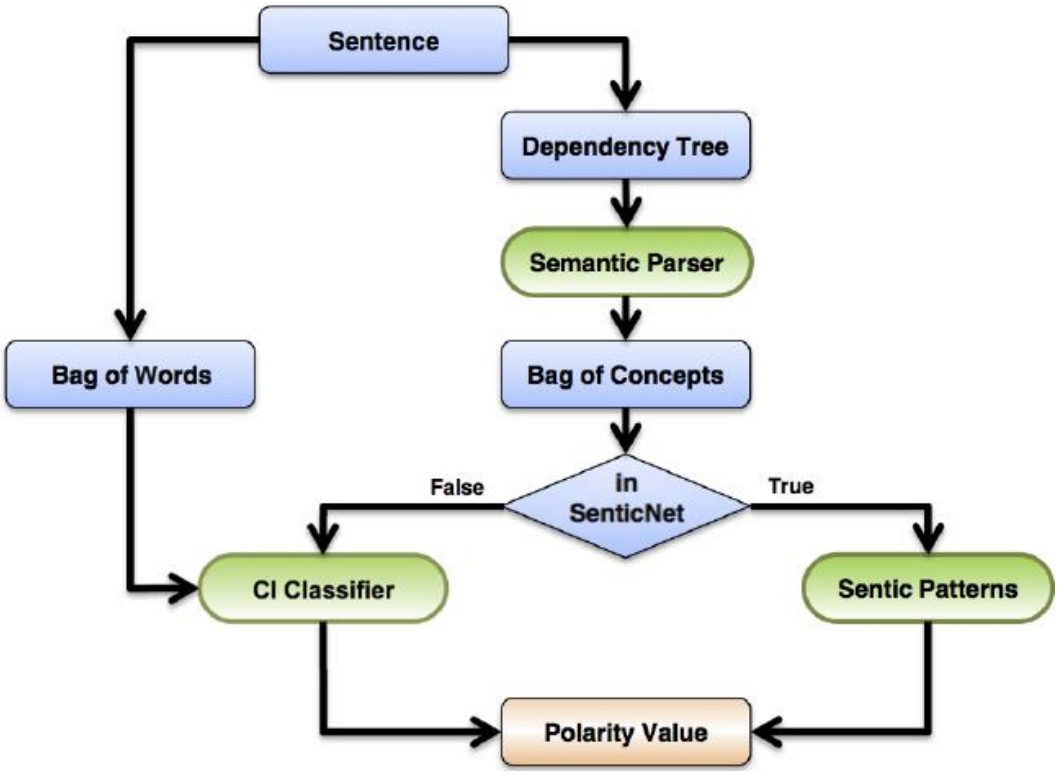


Рисунок 2.2 – Алгоритм парсингу

Будь-який власник сайту хоче захистити свій контент. Крадіжка будь-якої інформації - погано. Ваш контент може з'явитися на іншому ресурсі, ваша стаття може перестати вважатися унікальною.

Існує декілька методів, якими можна запобігти крадіжці контенту з вашого ресурсу:

– розмежування прав доступу. Це найпростіший метод. Ви можете приховати інформацію про структуру сайту. Зробити так, щоб вона була доступна тільки адміністраторам;

– вибір тривалості затримки між запитами. Цей метод добре працює, коли на сервер направляються хаотичні інтенсивні запити. Вони йдуть від однієї машини з різними проміжками. Ви можете встановити тимчасову затримку між запитами, які надходять від однієї машини;

– створення чорного і білого списку. Це списки користувачів. У білому знаходяться добропорядні користувачі. Чорний список для тих людей, які порушили правила поведінки сайту, намагалися вкрасти контент і т. Д.

– встановити інтервал оновлення сторінок. Щоб знизити ефективність парсинга, встановіть час оновлення сторінок у файлі sitemap.xml. Ви можете обмежити частоту запитів, обсяг даних при завантаженні;

– використання методів захисту від роботів. Так само як капча, підтвердження реєстрації на ресурсі. Те, що зможе виконати людина, але не зможе виконати машина [3].

Парсинг може використовуватися як на благо, так і на шкоду. Цей метод допомагає проаналізувати великі обсяги текстової інформації. Але в той же час, проаналізувати можуть вас, вкрасти контент, витягнути конфіденційну інформацію, яка не повинна потрапити в чужі руки.

Суть парсинга:

1) Користувач задає URL сайту, який необхідно парсити (це може бути як головна, так і внутрішня сторінка сайту). Можна також задавати відразу декілька URLов. Таким чином, вони все відразу будуть поміщені в чергу на обробку.

2) Починається перший прохід, парсер завантажує перший заданий URL і збирає з нього всі внутрішні посилання (в загальному випадку), поміщаючи їх в кінець черги на обробку.

3) Далі парсер знаходить в тілі сторінки блок контенту, згідно обмеженням, заданих в полях «Введіть регулярний вираз для пошуку початкової позиції обрізки» і «Введіть рядки, за якими буде знизу обрізатися стаття». Пошук блоку контенту і подальше додавання поста в блог проводиться тільки в разі, якщо сторінка проходить по заданих користувачем обмеженням. В іншому випадку, сторінка просто додається в базу надрукованих сторінок і відбувається перехід до кроку 8.

4) Далі парсер знаходить в тілі сторінки заголовок майбутнього поста.

5) Далі блок контенту проходить обробку (про неї я розповім пізніше).

6) Далі вирізаний і оброблений блок контенту, а також заголовок, надсилаються в сервіс Google Translate для перекладу з мови, обраного в списку «Вибір мови з якого переводити», на мову, обраний в списку «Вибір мови на який переводити». (якщо обраний один і той же мова в обох списках, крок 6 пропускається, і ви отримуєте на виході текст оригіналу).

7) Далі блок контенту з раніше визначеним заголовком додається у вигляді поста в блог користувача, а поточна сторінка додається в базу надрукованих сторінок.

8) Якщо чергу сторінок для обробки не порожня, то всі кроки з кроку 2 до кроку 8 повторюються до тих пір, поки не буде оброблено кількість сторінок, вказане в налаштуванні «Скільки максимум постів за цей прохід» або поки чергу сторінок для обробки не буде порожня.

9) Коли оброблено максимальну кількість сторінок за прохід, але чергу сторінок для обробки не порожня, користувачеві задається питання «Продовжити обробку необроблених сторінок?».

10) Якщо користувач натискає ОК, то починається наступний прохід (з кроку 2), і так до тих пір, поки черга сторінок для обробки не буде порожня або користувач не встановить галочку «Зупинити парсинг якомога швидше» [4].

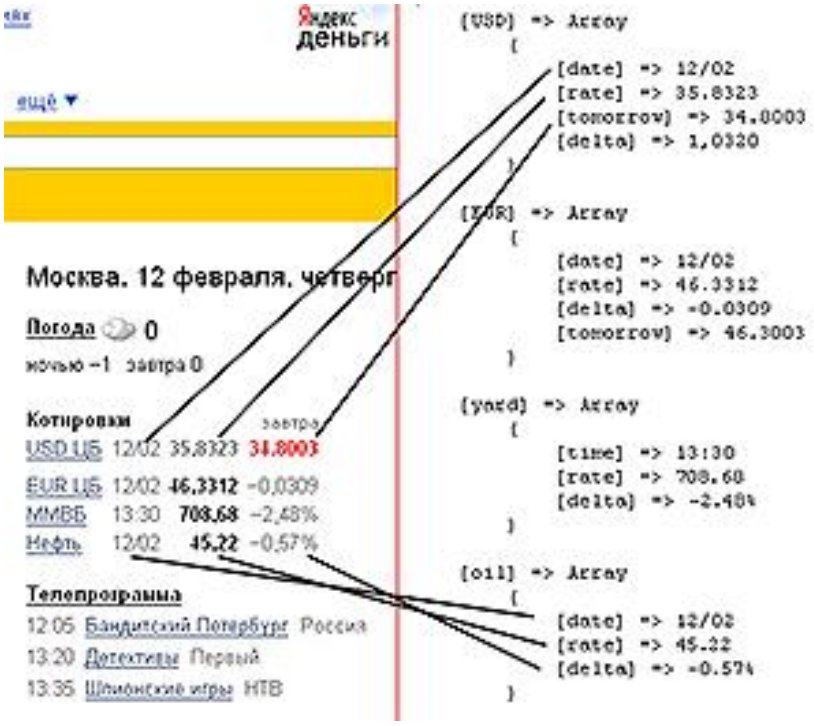


Рисунок 2.3 – Приклад виконання парсингу сайта

3 КОМП'ЮТЕРНА МОДЕЛЬ WEB-ДОДАТКУ

3.1 Огляд основних методів

Структура проекту:

\ spiders

\ Spiders \ __ init__.py

\ Spiders \ links.py

\ Spiders \ SpecSpider.py

__init__.py

items.py

pipelines.py

settings.py

Тут описані клас павука для отримання списку посилань на знайдені ресурси і клас павука для збору списку посилань. Для визначення полів використаний Xpath. У файлі pipelines.py описані дії по збереженню даних. Для створення бази даних sqlite із заданою структурою таблиць я використовував sqlalchemy.

По-перше створюємо екземпляр класу declarative_base (), від якого будемо наслідувати класи, для опису таблиць бази даних, в яких будемо зберігати знайдену інформацію. Це класи SpecTable, щоб зберегти список заголовків, і DataTable, для збереження даних авторів.

У кожному класі задаємо атрибут __tablename__. Це ім'я таблиці в базі даних. потім задаємо поля. Перше поле - первинний ключ. Решта поля, наприклад номер заловок.

У методі __init __ () заповнюємо поля таблиці.

У класі GtudataPipeline описаний процес роботи з базою даних. при ініціалізації перевіряємо наявність файлу бази даних в папці проекту. Якщо файл відсутній, то створюємо базу даних із заданою структурою.

У методі process_item описуємо власне збереження в базу даних. Перевіряємо, екземпляром якого класу є item. Залежно від цього заповнюємо

одну з двох таблиць. Для цього створюємо екземпляри класів DataTable і SpecTabl. Додаємо новий запис. При відкритті павука створюємо сесію. При закритті павука завершуємо зміни.

У методі ouptup ми виводимо отримані посилання. При цьому метод rating виводить процентне співвідношення тексту, який збігся.

3.2 Інструкція користувача

Для роботи з веб-застосунком користувач повинен виконувати наступні дії:

1) Запустити веб-застосунок
2) Ввести текст чи його частину в поле для введення
3) Натиснути кнопку «Отправить»
4) В разі необхідності очистити полу для введення натиснути кнопку «Очистить»
5) Дочекатися обробки запиту. В формі «Результат» можна буде побачити знайдені адреси веб-сторінок, де міститься шуканий текст, або його частина. Нижче в процентах виведений ступінь унікальності шуканого тексту.

3.3 Тестування прорами

Тестування - це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою оцінки.

Для данного веб-застосунка ми проведемо тестування «чорної скриньки» щоб перевірити наступне:

1) Як виконуються функції програми.

2) Як приймаються вихідні дані.

3) Які результати отримуємо.

3.4.1 Запуск веб-додатку

Заходимо на стартову сторінку. Як бачимо, сторінка запустилась і готова для роботи. Поле для вводу тексту активне, що свідчить про можливість вводу тексту та працездатність программи.

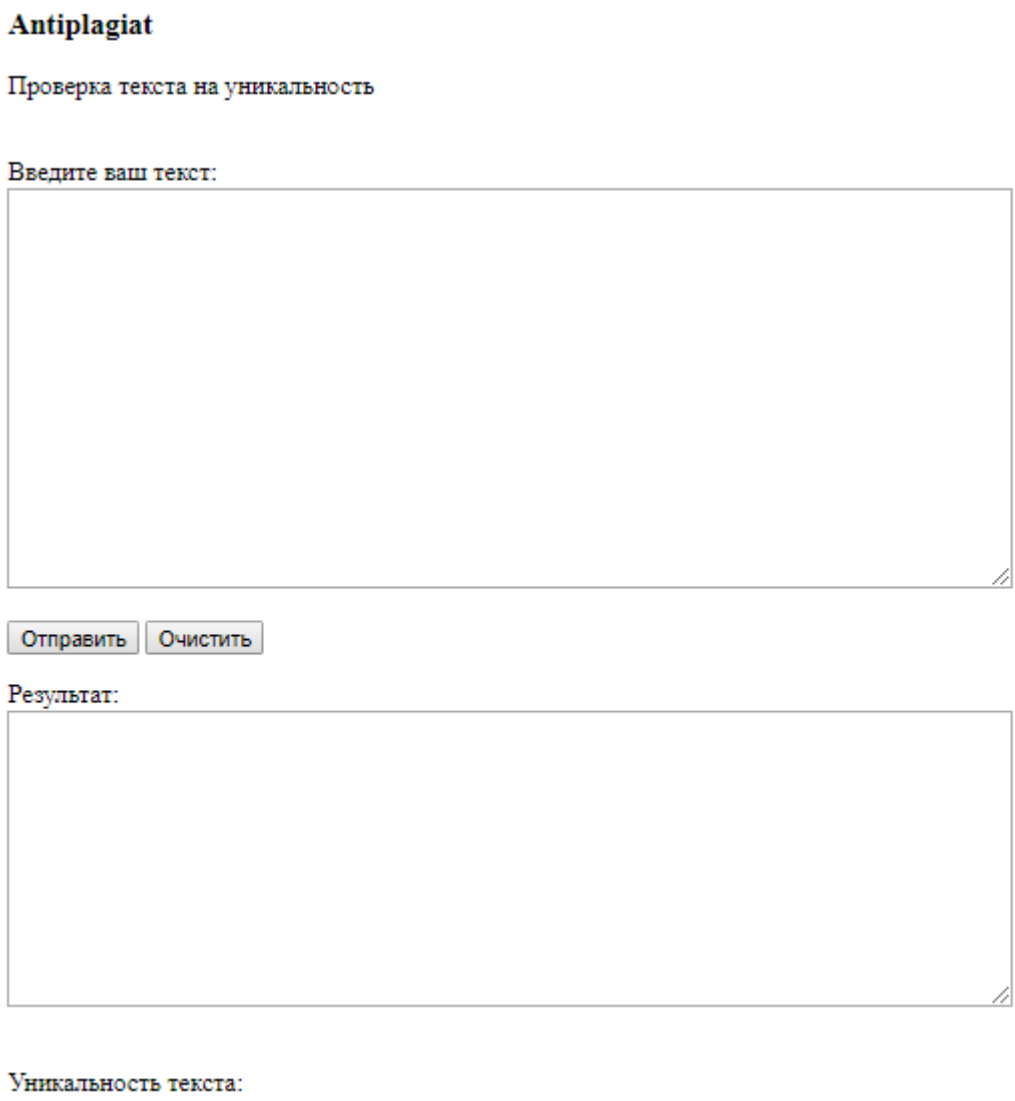


Рисунок 3.1- Стартова сторінка

3.4.2 Перевірка працездатності

Для перевірки функцій і результатів роботи застосунку введемо текст в поле для введення. Натиснемо кнопку «Очистити». Введений текст зник з поля введення, що свідчить про те, що метод очистки поля введення працює справно. Для подальшого тестування введемо текст в поле введення і натиснемо кнопку «Відправити».

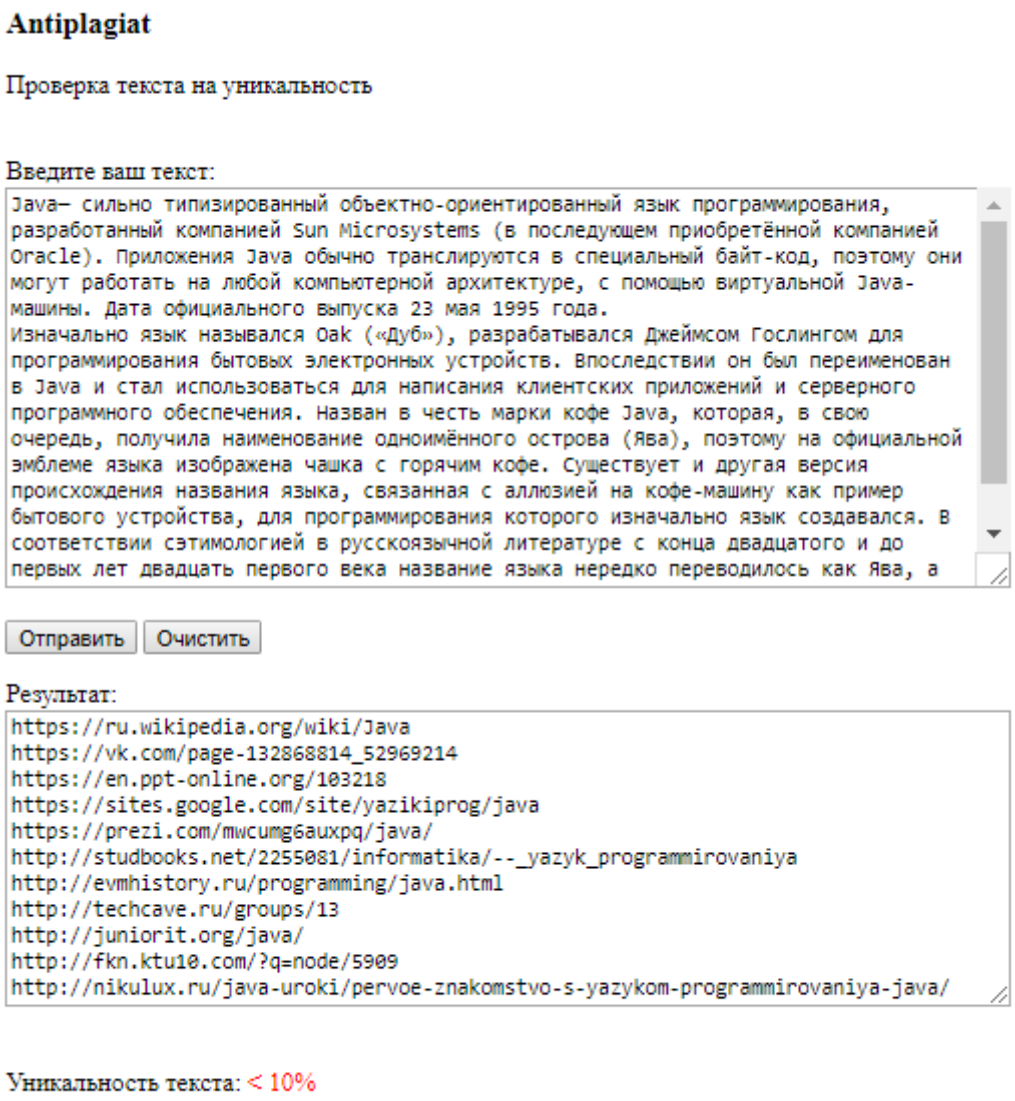


Рисунок 3.2 - Результат роботи програми

Після натискання кнопки «Відправити» в полі «Резальтат» зявилися посилання на сторінки, де був виявлений введений раніше текст.

Це свідчить про правильність роботи програми. Нижче поля «Результат» можемо бачити запис: «Унікальність тексту : < 10%». Це означає, що програма провела оцінку введеного тексту на унікальність. В нашому випадку це значення менше 10%. Тепер можна зробити висновок, що даний текст не є унікальним і розміщений на багатьох веб-ресурсах, що в свою чергу означає наявність плагіату.

Отже, в ході тестування програми порушень або неправильної роботи не було виявлено. Програма відповідає технічним вимогам і повністю працездатна. Інтерфейс зручний та інтуїтивно зрозумілий, придатний для використання. Всі кнопки працюють і виконують відповідні їм функції, програма правильно праймає і обробляє вхідний тескт, виводить достовірні результати та правильно оцінює текст на унікальність.

4 ОХОРОНА ПРАЦІ

4.1 Аналіз потенційних небезпечних і шкідливих виробничих чинників проектованого об’єкту, що мають вплив на персонал

У даному дипломному проекті розробляється програмне забезпечення.

Розроблене програмне забезпечення орієнтоване на роботу з персональним комп'ютером. Експлуатовані для вирішення внутрішньовиробничих завдань ПЕОМ типу IBM PC мають наступні характеристики:

споживана потужність	220 Вт;
робоча напруга	220 В;
напруга джерел живлення	+12 В; - 12 В; +5 В;
робоча частота	50 Гц.

Відповідно до [14] до легкої фізичної роботи відносяться всі види діяльності, виконувані сидячи і ті, що не потребують фізичної напруги. Робота користувача ПК відноситься до категорії 1а.

При роботі на ПЕОМ користувач піддається ряду потенційних небезпек. Унаслідок недотримання правил техніки безпеки при роботі з машиною (невиконання огляду відкритих частин ПЕОМ, що знаходяться під напругою або знятих для ремонту вузлів) для користувача існує небезпека поразки електричним струмом.

Джерелами підвищеної небезпеки можуть служити наступні елементи:

- розподільний щит;
- джерела живлення;
- блоки ПЕОМ і друку, що знаходяться в ремонті.

Ще одна проблема полягає у тому, що спектр випромінювання комп'ютерного монітора включає рентгенівську, ультрафіолетову і інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівського проміння мала, оскільки цей вид випромінювання

поглинається речовиною екрану. Проте велику увагу слід приділяти біологічним ефектам низькочастотних електромагнітних полів (аж до порушення ДНК).

Відповідно до [15], при обслуговуванні ПЕОМ мають місце фізичні і психофізичні небезпечні, а також шкідливі виробничі чинники:

- підвищене значення напруги в електричному ланцюзі, замикання
- якої може відбутися через тіло людини;
- підвищений рівень статичної електрики;
- підвищений рівень електромагнітних випромінювань;
- підвищена або знижена температура повітря робочої зони;
- підвищений або знижений рух повітря;
- підвищена або знижена вологість повітря;
- відсутність або недостатність природного світла;
- підвищена пульсація світлового потоку;
- недостатня освітленість робочого місця;
- підвищений рівень шуму на робочому місці;
- розумове перенапруження;
- емоційні навантаження;
- монотонність праці.

4.2 Заходи щодо техніки безпеки

Основним небезпечним чинником при роботі з ЕОМ є небезпека поразки людини електричним струмом, яка посилюється тим, що органи чуття людини не можуть на відстані знайти наявності електричної напруги на устаткуванні.

Проходячи через тіло людини, електричний струм чинить на нього складну дію, що є сукупністю термічної (нагрів тканин і біологічних середовищ), електролітичної (розкладання крові і плазми) і біологічної

(роздратування і збудження нервових волокон і інших органів тканин організму) дій.

Тяжкість поразки людини електричним струмом залежить від цілого ряду чинників:

– значення сили струму;

– електричного опору тіла людини і тривалості протікання через нього струму;

– роду і частоти струму;

– індивідуальних властивостей людини і навколишнього середовища.

Розроблений дипломний проект передбачає наступні технічні способи і засоби, що застерігають людину від ураження електричним струмом [16]:

– заземлення електроустановок;

– занулення;

– захисне відключення;

– електричне розділення ятерів;

– використовування малої напруги;

– ізоляція частин, що проводять струм;

– огорожа електроустановок.

Занулення зменшує напругу дотику і обмежує година, протягом якого людина, ткнувшись до корпусу, може потрапити під дію напруги.

Струм однофазного короткого замикання визначається по наближеній формулі:

$$I_K = \frac{\text{Uф}}{\text{Zп} + \frac{\text{Zт}}{3}}\ , \qquad (4.1)$$

де Uф - номінальна фазна напруга мережі, В;

Zп - повний опір петлі, створене фазними і нульовими дротами, Ом;

Z_T - повний опір струму короткого замикання на корпус, Ом.

Згідно таблиці 4 [17]: Zт /3 = 0,1 Ом.

Для провідників і жил кабелю для розрахунку повного опору петлі використовуємо формулу(4.2.) :

$$Zп = \sqrt{Rп^2 + Xп^2}\ , \qquad (5.2)$$

де Rп = Rф + Ro - сумарний активний опір фазного Rф і нульового Ro дротів, Ом;

Хп - індуктивний опір паяння дротів, Ом.

Перетин 1 км мідного дроту S = 2.5 мм, тоді згідно таблицям 5 і 6 [17], має такий опір:

Хп = 0,11 Ом;

Rф = 7,55 Ом;

Ro = 7,55 Ом.

Отже, Rп = 7,55 + 7,55 = 15,1 Ом.

Тоді по формулі(4.2) знаходимо повний опір петлі :

$$Zп = \sqrt{15{,}1^2 + 0{,}11^2} \approx 15{,}1(\text{Ом}).$$

Струм однофазного короткого замикання рівний:

$$Iк = \frac{220}{15{,}1 + 0{,}1} = 14{,}47\ \ (\text{А}).$$

Дія плавкої вставки на ПЕОМ забезпечується, якщо виконується співвідношення:

$$Iк \geq k * Iн\ , \qquad (4.3)$$

де Ін - номінальний струм спрацьовування плавкої вставки, А;

k - коефіцієнт кратності нелінійного струму Ін, А.

Коефіцієнт кратності нелінійного струму Ін розраховується по формулі(4.4.) :

$$Ін = P / U, \qquad (4.4)$$

де P = 220 Вт - споживана потужність;

U = 220 В - робоча напруга;

k = 3 А - для плавких вставок.

Отже, Ін = 220 / 220 = 1 А.

Підставивши значення у вираз(4.3), одержимо:

$$14{,}47 > 3*1.$$

Таким чином, доведено, що апарат забезпечить спрацьовування (і захист) при підвищенні номінального струму.

4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці

Вимоги до виробничих приміщень встановлюються [18], СНіП, відповідними ГОСТами і ОСТамі з урахуванням небезпечних і шкідливих чинників, що утворюються в процесі експлуатації електроустаткування.

Підвищення працездатності людини і збереження її здоров'я забезпечується стабільними метеорологічними умовами. Мікроклімат виробничих приміщень [14] визначається діючими на організм людини поєднаннями температури, вологості і швидкості руху повітря, а також температури навколишніх поверхонь. Значне коливання параметрів мікроклімату приводити до порушення систем кровообігу, нервової і потовидільної, що може викликати підвищення або пониження температури тіла, слабкість, запаморочення і навіть непритомність.

Відповідно до [14] встановлюють оптимальну і допустиму температуру, відносну вологість і швидкість руху повітря в робочій зоні . За відсутності надмірного тепла, вологи, шкідливих речовин в приміщенні досить природної вентиляції.

У приміщенні для виконання робіт операторського типу (категорія 1а), пов'язаних з нервово-емоційною напругою, проектом передбачається дотримання наступних нормованих величин параметрів мікроклімату (табл. 4.1).

Таблиця 4.1 - Санітарні норми мікроклімату робочої зони приміщень для робіт категорії 1а.

Пора року	Температура, С	Відносна вологість, %	Швидкість руху повітря, м/с
Холодна	22…24	40…60	0,1
Тепло	23…25	40…60	0,1

У приміщенні, де знаходиться ПЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (з пристроєм вентиляційних каналів в перекриттях будівлі і вертикальних шахт) й установленого промислового кондиціонера фірми Mitsubishi, який дозволяє вирішити переважну більшість завдань по створенню та підтримці необхідних параметрів повітряного середовища. Цей метод забезпечує приток потрібної кількості свіжого повітря, визначеного в СНіП (30 $м^3$ в годину на одного працівника).

Шум на виробництві має шкідливу дію на організм людини. Стомлення операторів через шум збільшує число помилок при роботі, призводить до виникнення травм. Для оператора ПЕОМ джерелом шуму є робота принтера. Щоб усунути це джерело шуму, використовують наступні методи. При покупці принтера слід вибирати найбільш шумозахисні матричні принтери або з великою швидкістю роботи (струменеві, лазерні). Рекомендується

принтер поміщати в найбільш віддалене місце від персоналу, або застосувати звукоізоляцію та звукопоглинання (під принтер підкладають демпфуючі підкладки з пористих звукопоглинальних матеріалів з листів тонкої повсті, поролону, пеноплену).

При роботі на ПЕОМ, проектом передбачені наступні методи захисту від електромагнітного випромінювання: обмеження часом, відстанню, властивостями екрану.

Обмеження годині роботи на ПЕОМ складає 3,5-4,5 години. Захист відстанню передбачає розміщення монітора на відстані 0,4-0,5 м від оператора. Передбачений монітор 20" TFT, Samsung 2043BW відповідає вимогам стандарту [19].

Стандарт [19] пред’являє жорсткі вимоги в таких областях: ергономіка (фізична, візуальна і зручність користування), енергія, випромінювання(електричних і магнітних полів), навколишнє середовище і екологія, а також пожежна та електрична безпека, які відповідають всім вимогам [15].

Для зниження стомлюваності та підвищення продуктивності праці обслуговуючого персоналу в колірній композиції інтер'єру приміщень для ПЕОМ дипломним проектом пропонується використовувати спокійні колірні поєднання і покриття, що не дають відблисків.

У проекті передбачається використання сумісного освітлення. У світлий час доби приміщення освітлюватиметься через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Як штучне освітлення необхідно використовувати штучне робоче загальне освітлення. Для загального освітлення необхідно використовувати люмінесцентні лампи. Вони володіють наступними перевагами: високою світловою віддачею, тривалим терміном служби, хоча мають і недоліки: високу пульсацію світлового потоку.

При експлуатації ПЕОМ виробляється зорова робота. Відповідно до [20] ця робота відноситься до розряду 5а. При цьому нормоване освітлення на робочому місці(Ен) при загальному освітленні дорівнює 200 лк.

Приміщення завдовжки 12 м, шириною 10 м, заввишки 4 м обладнується світильниками типу ЛП02П, оснащеними лампами типу ЛБ зі світловим потоком 3120 лм кожна.

Виконаємо розрахунок кількості світильників в робочому приміщенні завдовжки a=12 м, шириною b=10 м, заввишки з=4 м, використовуючи формулу (4.5) розрахунку штучного освітлення при горизонтальній робочій поверхні методом світлового потоку:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \quad (4.5)$$

де F - світловий потік = 3120 лм;

E - максимально допустима освітленість робочих поверхонь = 200 лк;

S - площа підлоги = 120 м2;

Z - поправочний коефіцієнт світильника = 1,2;

k - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації світильників = 1,5;

n - кількість світильників;

U - коефіцієнт використовування освітлювальної установки = 0,6;

M - кількість ламп у світильнику =2.

З формули(4.5) виразимо n(4.6) і визначимо кількість світильників для даного приміщення:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \quad (4.6)$$

Отже, $n = (200 \cdot 120 \cdot 1{,}2 \cdot 1{,}5)/(3120 \cdot 0{,}6 \cdot 2) = 12$.

Виходячи з цього, рекомендується використовувати 12 світильників. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. 4.1.

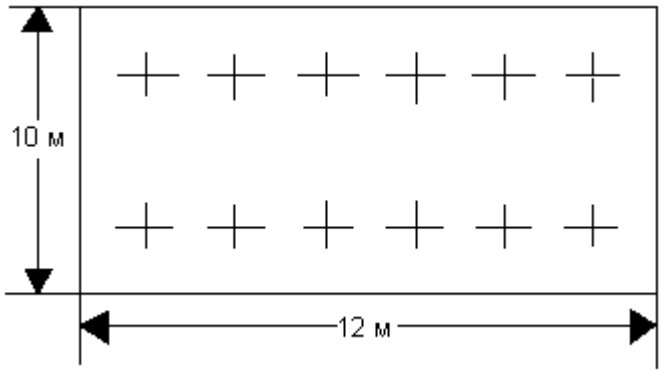


Рисунок 4.1 - Схема розташування світильників

4.4 Рекомендації по пожежній безпеці

Пожежі в приміщеннях, де встановлена обчислювальна техніка, представляють небезпеку для життя людини. Пожежі також пов’язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки, що у свою чергу спричиняє за собою порушення ходу технологічного процесу.

Пожежа може виникнути при наявності горючої речовини та внесення джерела запалювання в горюче середовище. Пальними матеріалами в приміщеннях, де розташовані ПЕОМ, є:

– поліамід - матеріал корпусу мікросхеми, горюча речовина, температура самозаймання аерогелю 420 З ;

– полівінілхлорид - ізоляційний матеріал, горюча речовина, температуразапалювання 335 З, температура самозаймання 530 З, кількість енергії, що виділяється при згоранні - 18000 - 20700 кДж/кг;

– стеклотекстоліт ДЦ - матеріал друкарських плат, важкозаймистий матеріал, показник горючості 1.74, не схильний до температурного самозаймання;

– пластика кабельний №489 - матеріал ізоляції кабелю, горючий матеріал, показник горючості більш 2.1;

– деревина - будівельний і обробний матеріал, матеріал з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, теплота згорання 18731 - 20853 кДж/кг, температура запалювання 399 З, схильна до самозаймання [17].

Згідно [21] приміщення відносяться до категорії В (пожежовибухонебезпечним) і згідно правилам побудови електроустановок простір усередині приміщення відноситься до вогненебезпечної зони класу П - IIа (зони, розташовані в приміщеннях, в яких зберігаються тверді горючі речовини).

Потенційними джерелами запалення при роботі ПЕОМ є:

– іскри при замиканні і розмиканні ланцюгів;

– іскри і дуги коротких замикань;

– перегріви від тривалого перевантаження і наявності перехідного опору.

Продуктами згорання, що виділяються при пожежі, є: оксид вуглецю, сірчистий газ, оксид азоту, синильна кислота, акропеін, фосген, хлор та ін. При горінні пластмас, окрім звичайних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідрідні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол та ін., що шкідливо впливають на організм людини.

Для захисту персоналу від дії небезпечних і шкідливих чинників пожежі проектом передбачається застосування промислового протигаза з коробкою марки В (жовта).

Пожежна безпека об’єктів народного господарства регламентується [9] і забезпечується системами запобігання пожежам і протипожежному захисту.

Для успішного гасіння пожеж вирішальне значення має швидке виявлення пожежі і своєчасний виклик пожежних підрозділів до місця пожежі.

Зменшити горюче навантаження не представляється можливим, тому проектом передбачається застосувати наступні способи і їх комбінації для запобігання утворенню(внесення) джерел запалення :

– застосування устаткування, що задовольняє вимогам електростатичної безпеки;

– застосування в конструкції швидкодіючих засобів захисного відключення можливих джерел запалення;

– виключення можливості появи іскрового заряду статичної електрики в горючому середовищі з енергією, рівної і вище мінімальної енергії запалення;

– підтримка температури нагріву поверхні машин, механізмів, устаткування, пристроїв, речовин і матеріалів, які можуть увійти до контакту з пальним середовищем, нижче гранично допустимої, становить 80% якнайменшої температури самозаймання пального.

– заміна небезпечних технологічних операцій більш безпечними;

– ізольоване розташування небезпечних технологічних установок і устаткування;

– зменшення кількості пальних і вибухонебезпечних речовин, що знаходяться у виробничих приміщеннях;

– запобігання можливості утворення пальних сумішей на лінії, вентиляційних системах і ін.;

– механізація, автоматизація та справність(потокова) виробництва;

– суворе дотримання стандартів і точне виконання встановленого технологічного режиму;

– запобігання можливості появи в небезпечних місцях джерел запалення;

– запобігання розповсюдженню пожеж і вибухів;

– використовування устаткування і пристроїв, при роботі яких не виникає джерел запалення;

– виконання вимог сумісного зберігання речовин і матеріалів;

– наявність громовідводу;

– організація автоматичного контролю параметрів, що визначають джерела запалення;

– ліквідація можливості самозаймання речовин і матеріалів.

Для запобігання пожежі в обчислювальних центрах проектом пропонується виконання наступних вимог:

– електроживлення ЕОМ повинно мати автоматичне блокування відключення електроенергії на випадок зупинки системи охолоджування і кондиціонування;

– система вентиляції обчислювальних центрів повинна бути обладнана блокуючими пристроями, що забезпечують її відключення на випадок пожежі;

– робочі місця повинні бути оснащені пожежними щитами, сигналізацією, засобами для сповіщення про пожежну небезпеку (телефонами), медичними аптечками для надання першої медичної допомоги, розробленим планом евакуації.

Для зниження пожежної небезпеки в приміщеннях використовуються первинні засоби гасіння пожеж, а також система автоматичної пожежної сигналізації, яка дозволяє знайти початкову стадію загоряння, швидко і точно оповістити службу пожежної охорони про час і місце виникнення пожежі.

Відповідно до [23] приміщення категорії В підлягають устаткуванню системами автоматичної пожежної сигналізації. Проектом передбачається застосування датчика типу ІДФ - 1(димовий фотоелектричний датчик), оскільки специфікою пожеж обчислювальної техніки і радіоапаратури є, в першу чергу, виділення диму, а потім - підвищення температури.

При виникненні пожежі в робочому приміщенні обслуговуючий персонал зобов'язаний негайно вжити заходи по ліквідації пожежі. Для

ліквідації пожежі використовують вогнегасники (хімічно-пінні, пінні для повітря ОП-5, ОП-6, ОП-9, вуглекислотні ОУ-5), пісок, пожежний інвентар(сокири, ломи, багри, шерстяну або азбестову ковдри) [24]. Як засіб індивідуального захисту проектом передбачається використовування промислового протигаза з маскою, фільтруючої коробки В.

В якості організаційно-технічних заходів рекомендується проводити навчання робочого персоналу правилам пожежної безпеки.

Висновки

Сучасний розвиток інформаційних технологій, а саме розвиток мережі Інтернет, значно спростив процес обміну інформацією. Доступність інформаційних ресурсів безперечно сприяє розвитку особистості та суспільства в цілому. У зв'язку з цим, актуальною постає задача ідентифікації даних у процесі аналізу інформаційних потоків, що є головним завданням сучасних пошукових систем, спрямованих на підбір унікального контенту Web-ресурсів. Як відомо, популярні пошукові системи, такі, наприклад, як Google, при виведенні списку сайтів за запитом користувача віддають перевагу сайтам з унікальним контентом .

Крім того, важливого значення набуває проблема визначення унікальності даних в інформаційному просторі з метою виключення явища плагіату у процесі ідентифікаційного аналізу наявних ресурсів баз даних. Такі питання є актуальними, перш за все, для вищих навчальних закладів та науково-дослідних установ. Специфіка призначення автоматизованих систем пошуку плагіату (АСПП) зумовлює потребу зміни та модифікації існуючих методів і засобів встановлення унікальності даних. Найвідомішим в Україні спеціалізованим програмним забезпеченням такого типу є програми, які використовують експерти ВАК України для перевірки дисертацій на плагіат . Але розробка нових та удосконалення відомих методів і засобів пошуку плагіату та їх реалізація в автоматизованій пошуковій системі залишається актуальною.

У рамках дипломної роботи був розроблений і реалізований веб-застосунок для пошуку і оцінки унікальності тексту чи його частин в середовищі Eclipse. Були використані технології пошуку та обробки веб-сторінок, розроблений метод перевірки та оцінки тексту на унікальність.

Під час четвертого розділу було проаналізовано умови праці, виявлені причини травматизму і захворювань, можливі небезпечні й шкідливі виробничі фактори. Також було проведено ряд розрахунків щодо виконання

вимог охорони праці в приміщенні відділу програмного забезпечення. Дотримання цих вимог є важливим для збереження працездатності та здоров’я працівників.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1) Ґвідо ван Россум Підручник мови Python [Текст] / Фред Л. Дарк Молодший перевод с англ. – К.: ДиаСофт, 1993. – 264 с.

2) Пориньте у Python 3 Марк Пілігрим [Текст] / Д.А. Форсайт, Д. Понс; перевод с англ. – К.: Вильямс, 2004. – 928 с.

3) Веб-парсинг на Ruby [Електронний ресурс] – Режим доступу до ресурсу: https://habrahabr.ru/post/252379/.

4) МЕТОДЫ ПАРСИНГА САЙТОВ [Електронний ресурс] – Режим доступу до ресурсу: http://ponka.vnukov.ru/content/metody-parsinga-saytov.

5) Grune D. Parsing Techniques - A Practical Guide [Текст] / D. Grune, C. Jacobs. – Chichester: Originally published by Ellis Horwood, 1990. – 320 с.

6) Aho A. V. The theory of parsing, translation, and compiling [Текст] / A. V. Aho, J. D. Ullman. – USA: Prentice-Hall, 1972. – 121 с.

7) AngularJS [Електронний ресурс] – Режим доступу до ресурсу: https://angularjs.org/.

8) Popular Technology [Електронний ресурс] Режим доступу: http://www.popultechnology.net/2010/08/google-scholar-illiteracy-in-pnas.html.

9) Lxml - XML and HTML with Python [Електронний ресурс] – Режим доступу до ресурсу: http://lxml.de/.

10) Beautiful Soup 4 Python [Електронний ресурс] – Режим доступу до ресурсу: http://www.pythonforbeginners.com/beautifulsoup/beautifulsoup-4-python.

11) Парсинг: Что? Зачем? Как? [Електронний ресурс] – Режим доступу до ресурсу: http://parsing.valemak.com.

12) 7 способов защитить сайт от парсинга и как их обойти [Електронний ресурс] – Режим доступу до ресурсу: https://ergonotes.ru/7-sposobov-zashhitit-sayt-ot-parsinga-i-kak-ih-oboyti.

13) Mitchell R. Web Scraping with Python [Текст] / Ryan Mitchell. – Boston: O\'Reilly Media, 2015. – 256 с.

14) Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. Постанова N 42 від 01.12.99. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/va042282-99

15) Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин ДСанПІН 3.3.2.007-98. Затверджено Постановою Головного державного санітарного лікаря України 10 грудня 1998 р. N 7. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

16) НПАОП 40.1-1.21-98 «Правила безпечної експлуатації електроустановок споживачів». *Наказ від 09.01.98 №4.* Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0093-98

17) ГОСТ 12.1.044-89 «Система стандартов безопасности труда. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения». Постанова від 12.12.1989 № 3683. Режим доступу: www. URL: http://online.budstandart.com/ru/catalog/doc-page?id_doc=51048

18) *ДСП 173-96 «Державні санітарні правила планування та забудови населених пунктів». Наказ від 19.06.1996 №173.* Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0379-96

19) TCO'07 Certified Displays. © 2007 Copyright TCO Development AB. Режим доступу: www. URL: https://tcocertified.com/files/2015/11/TCO-Certified-Displays-7.0.pdf

20) ДБН В.2.5-28:2018 «Природне і штучне освітлення». Режим доступу: www. URL: http://www.minregion.gov.ua/wp-content/uploads/2018/12/V2528-1.pdf

21) ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою». Наказ від 15.06.2016 №158. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/v0158858-16

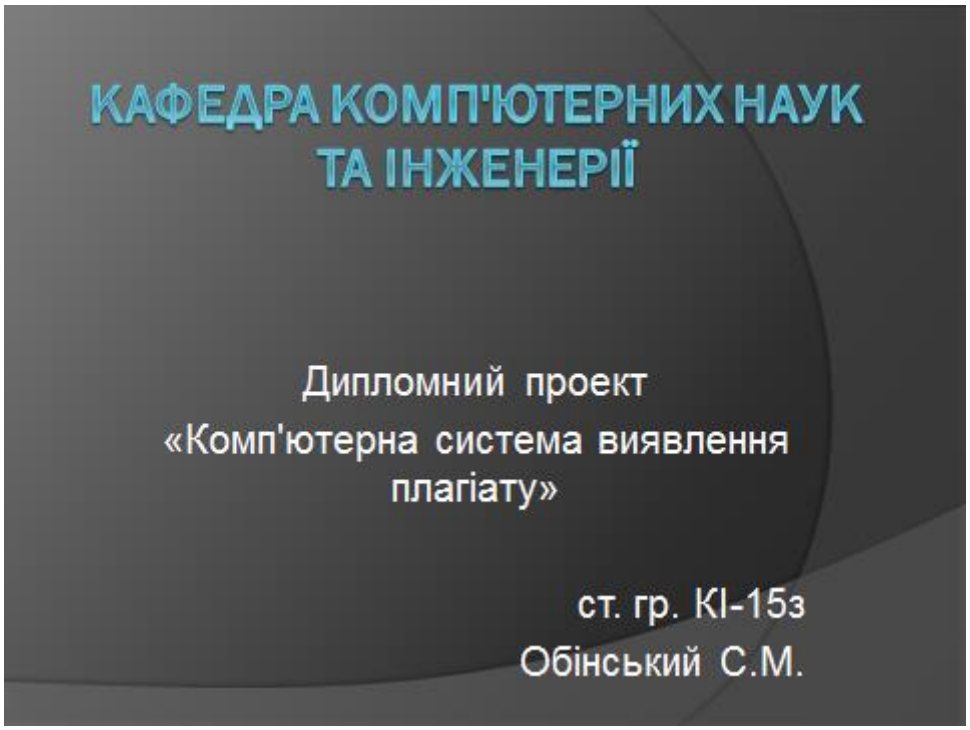
22) ГОСТ 12.1.004-91 «Система стандартов безопасности труда. Пожарная безопасность. Общие требования». Режим доступу: www. URL: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=48679

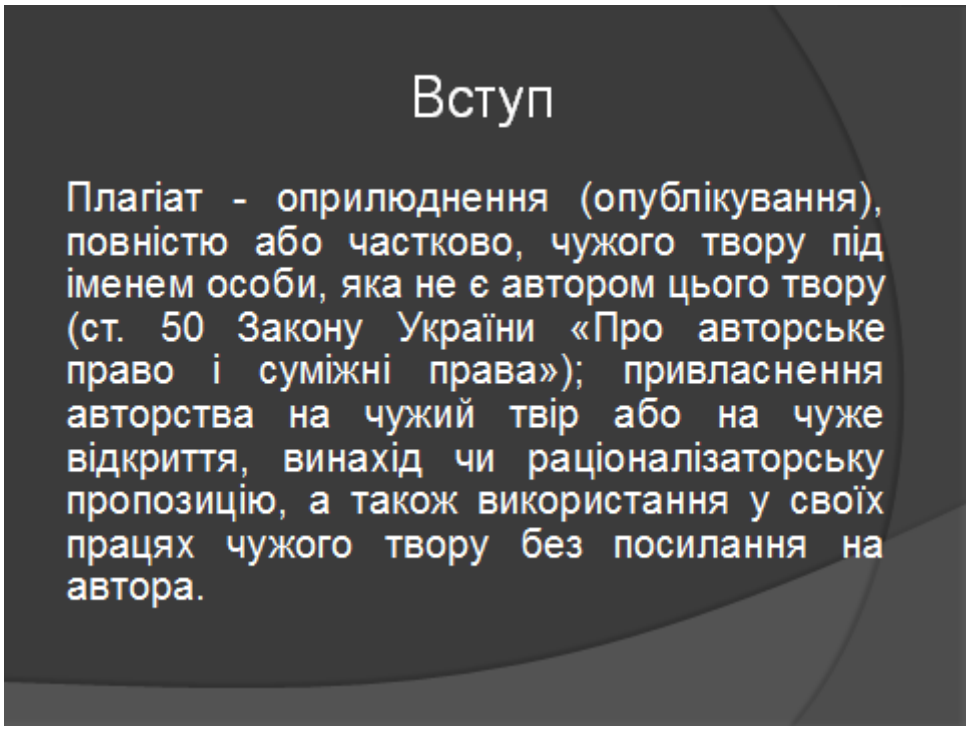
23) НАПБ А.01.001-2014 «Правила пожежної безпеки в Україні». Наказ від 30.12.2014 №1417. Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0252-15

24) НАПБ Б.01.008-2018 «Про затвердження правил експлуатації та типових норм належності вогнегасників». Наказ від 15.01.2018 №25. Режим доступу: www. URL: http://search.ligazakon.ua/ l_doc2.nsf/link1/ RE31677.html

ДОДАТОК А.

Електронні плакати





Види плагіату:

- - Копіювання інформації іншого автора та видання роботи за свою без оформлення цитування;
- - Дослівне копіювання чужої роботи у свою без належного оформлення цитування;
- - Парафраза - переказ своїми словами тексту іншого автора, суть якого полягає в заміні слів та знаків;
- - Компіляція - процес написання твору, наукової праці на підставі чужих матеріалів без самостійного дослідження та опрацювання джерел.

Актуальність проблеми

Розвиток електронних інформаційних ресурсів сприяє підвищенню якості досліджень вчених і аналітиків за рахунок більшої доступності, прискорення пошуку необхідної інформації. Ці фактори, в свою чергу, сприяють поширенню плагіату.

Отже, дане питання є актуальним і не буде втрачати своєї актуальності, оскільки з роками інформаційні ресурси будуть розширюватися, а значить, збільшиться кількість не унікальних робіт.

Постановка задачі

Ставиться завдання розробки методу пошуку веб-сторінок з відповідним змістом та знаходження однаковго слів на цих сторінках.

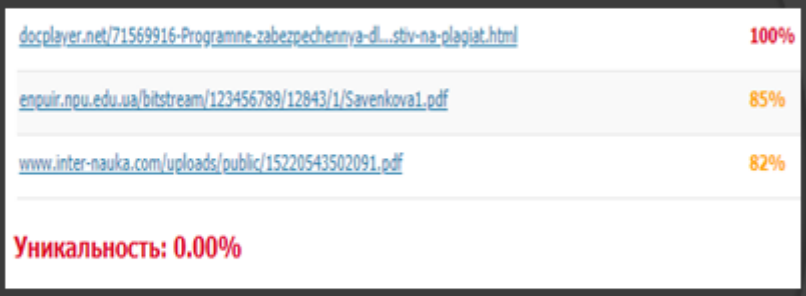
Для цього необхідно вирішити такі завдання:

- провести аналіз існуючих методів пошуку плагіату в Інтренеті;
- розробити метод аналізу тексту;
- реалізувати метод визначення оцінки унікальності тексту на знайдених веб-сторінках ;
- - реалізувати веб-застосунок для користування розробленими методами.

Аналіз основних частин

- Для знаходження веб-сторінок був розроблений веб-краулер, призначений для перебору сторінок та їх первинного аналізу.
- Для повного аналізу веб-сторінок був розроблений парсер і функції для порівняння вхідного тексту і тексту веб-сторінок.
- Також був розроблений метод оцінки і виведення результатів перевірки унікальності введеного тексту.

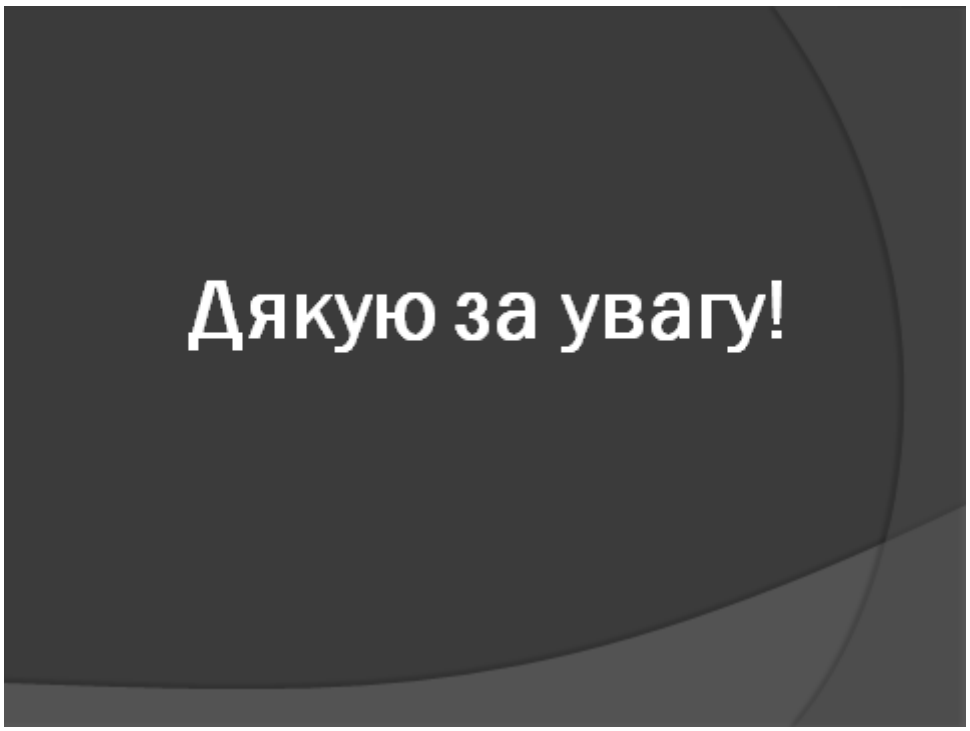
Тестовий приклад



Аналіз результатів

В результаті виконання роботи був розроблений веб-застосунок, який дозволяє виконувати пошук тексту на веб-сторінках для аналізу та зіставлення його з введеним текстом.

По завершенню роботи програми можемо спостерігати список знайдених адрес і відсоток не унікального матеріалу.


Дякую за увагу!