метадані

Заголовок
2023_ФЕП_41с_Крамар_Н_Ю_текст.pdf

Автор
Назар Крамар

Науковий керівник / Експерт
Олег Кушнір

підрозділ
Факультет електроніки та комп'ютерних технологій

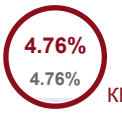
Перелік можливих спроб маніпуляцій з текстом

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

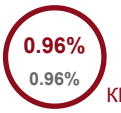
Заміна букв	ß	1
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки	ß	0
Парафрази (SmartMarks)	a	21

Обсяг знайдених подібностей

Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 2



КЦ

10
Довжина фрази для коефіцієнта подібності 2

6282
Кількість слів

49634
Кількість символів

Подібності за списком джерел

Прокручуйте список та аналізуйте, особливо, фрагменти, які перевищують КП 2 (позначено жирним шрифтом). Скористайтеся посиланням "Позначити фрагмент" та перегляньте, чи є вони короткими фразами, розкиданими в документі (випадкові схожості), численними короткими фразами поруч з іншими (мозаїчний плагіат) або великими фрагментами без зазначення джерела (прямий плагіат).

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	http://dwl.kiev.ua/art/oiad/oiad.pdf	84	1.34 %
2	http://samzan.net/212872	30	0.48 %
3	http://samzan.net/212872	30	0.48 %
4	http://inmad.vntu.edu.ua/portal/static/D0498CB2-9985-4F82-908A-54B72B2F3B42.pdf	27	0.43 %
5	2023_Фел-41_Ковальчук_БО_робота.pdf [redacted] 6/9/2023 The Ivan Franko National University (Факультет електроніки та комп'ютерних технологій)	18	0.29 %
6	http://dwl.kiev.ua/art/oiad/oiad.pdf	15	0.24 %

7	http://inmad.vntu.edu.ua/portal/static/D0498CB2-9985-4F82-908A-54B72B2F3B42.pdf	15	0.24 %
8	2021_Фel-41_Калашніков_В.О._текст.pdf [redacted] 6/7/2021 The Ivan Franko National University (Факультет електроніки та комп'ютерних технологій)	14	0.22 %
9	http://ukrkniga.org.ua/ukrkniga-text/152/5/	13	0.21 %
10	2023_Фel-41_Ковальчук_БО_робота.pdf [redacted] 6/9/2023 The Ivan Franko National University (Факультет електроніки та комп'ютерних технологій)	11	0.18 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з домашньої бази даних (0.86 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	2023_Фel-41_Ковальчук_БО_робота.pdf [redacted] 6/9/2023 **The Ivan Franko National University (Факультет електроніки та комп'ютерних технологій)**	**29 (2)**	**0.46 %**
2	2021_Фel-41_Калашніков_В.О._текст.pdf [redacted] 6/7/2021 **The Ivan Franko National University (Факультет електроніки та комп'ютерних технологій)**	**25 (3)**	**0.40 %**

з програми обміну базами даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з Інтернету (4.15 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	**http://dwl.kiev.ua/art/oiad/oiad.pdf**	**99 (2)**	**1.58 %**
2	**http://samzan.net/212872**	**71 (3)**	**1.13 %**
3	**http://inmad.vntu.edu.ua/portal/static/D0498CB2-9985-4F82-908A-54B72B2F3B42.pdf**	**63 (4)**	**1.00 %**
4	**http://ukrkniga.org.ua/ukrkniga-text/152/5/**	**13 (1)**	**0.21 %**
5	**https://er.knutd.edu.ua/bitstream/123456789/19447/1/Dyplom122_Glembotskiy_Astistova.pdf**	**10 (1)**	**0.16 %**
6	http://195.22.112.37/bitstream/ntb/657/1/07.pdf	5 (1)	0.08 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій
Кафедра оптоелектроніки та інформаційних **технологій**

Допустити до захисту
Завідувач кафедри

_____ _____.
(підпис) (ПІБ)

≪____≫ _____20__ **р.**
Кваліфікаційна робота
Бакалавр
(освітній ступень)

На тему " Програма вивчення лінгвістичної статистики "

Виконав:
Студент групи ФЕП-41.
Спеціальності:
121 Інженерія Програмного Забезпечення
_____ ___Крамар Н. Ю.___
(підпис) (ПІБ)

Науковий керівник:
_____ __проф. Кушнір О. С._
(підпис) (ПІБ)

≪_____≫ _____2023 р.
Рецензент:
_____ _проф. Стадник В. Й._
(підпис) (ПІБ)

Львів 2023

АНОТАЦІЯ

Дипломна робота присвячена розробці програмного забезпечення для вивчення лінгвістичної статистики. В роботі досліджено методи аналізу текстів та використання статистичних методів у лінгвістиці.

Для розробки програмного забезпечення використано мову програмування C# у середовищі Visual Studio з допомогою Microsoft .Net Framework на базі Об'єктно-орієнтованого програмування. Було розроблено програмний продукт, що включає набір інструментів для аналізу текстів та обробки лінгвістичних даних.

У роботі описано основні функції програми, підрахунок частоти вживання слів, аналіз структури тексту, визначення стилістики та інші. Було проведено тестування програми на різноманітних даних, що дозволило оцінити її ефективність та точність.

Результати дослідження та розробки можуть бути використані в лінгвістичних дослідженнях та при вивченні мови, а також у сферах, пов'язаних з аналізом текстів та лінгвістичною обробкою даних.

ABSTRACT

The **thesis is devoted to the development of software for the** study of linguistic statistics. The work examines the methods of text analysis and the use of statistical methods in linguistics.

To develop the software, the C# programming language was used in the Visual Studio environment with the help of the Microsoft .Net Framework based on Object-oriented programming. A software product was developed, including a set of tools for analyzing texts and processing linguistic data.

The work describes the main functions of the program, the calculation of the frequency of

word usage, the analysis of the text structure, the definition of stylistics, and others. The program was tested on a variety of data, which made it possible to assess its effectiveness and accuracy.

The results of research and development can be used in linguistic research and language learning, as well as in areas related to text analysis and linguistic data processing.

ЗМІСТ

ВСТУП

Особливо актуальним в наш час є міждисциплінарні дослідження мови, а статистична лінгвістика є одним з напрямків, які відкривають нові шляхи для дослідження літератури та мови. Мова як складна система дискретних одиниць має не тільки якісні, але й кількісні характеристики. Статистичні методики з комп'ютерною підтримкою мають величезний потенціал для вирішення багатьох теоретичних та практичних завдань лінгвістики та обробки текстових даних. Результати, отримані статистичними методами, застосовуються в багатьох сферах сучасної науки: стилеметрії, лінгводидактиці, глоттохронології, дешифруванні історичних писемностей, стенографії, судовій та кримінальній лінгвістиці, комп'ютерних технологіях тощо.

Оскільки мова - це ймовірнісна, а не жорстко детермінована система, то для її пізнання квантитативні методи, пов'язані з дослідженням частотних, ймовірнісних, градуальних та інших характеристик, **не тільки бажані, але й необхідні.** **Подібність між членами одного мовного колективу полягає не тільки в тому, які мовні одиниці (фонеми, лексеми, граматичні форми і синтаксичні конструкції) вони** використовують, але й **у тому, як часто вони їх уживають.** На цьому факті побудовані **частотні словники, у яких слова розташовані не за алфавітом, а за** способами використання у мовленні. Такі словники допомагають вивчати мову як ймовірнісну систему і досліджувати залежності між різними мовними одиницями. Дослідження частотного складу мови дозволяє розуміти, які слова і висловлювання є найбільш типовими для даної мови, а також допомагає вивчати особливості мовлення різних груп населення (наприклад, за віком, статтю, регіоном тощо). Крім того, частотний аналіз дозволяє виявити мовні тенденції, зміни у мовленні протягом часу і визначити найбільш часто вживані слова, що корисно при вивченні мови та її перекладі. Отже, використання квантитативних методів дослідження мови є необхідним для повного розуміння мовної системи та розвитку мовознавства як науки.

РОЗДІЛ 1. ЛІТЕРАТУРНИЙ ОГЛЯД

1.1. Лінгвістика

Лінгвістика (від лат. Lingua - мова) — наука, що вивчає мову як систему знаків,

засіб комунікації і взаємодії між людьми. Це широке поле, яке включає в себе вивчення мовних структур, звуків мови, граматики, семантики, психолінгвістики (вивчення мовленнєвої діяльності), соціолінгвістики (вивчення соціальної ролі мови), прикладної лінгвістики (застосування знань лінгвістики в різних практичних сферах, таких як переклад, викладання мови, розробка програмного забезпечення для машинного перекладу тощо).
Лінгвістика є міждисциплінарною наукою, що взаємодіє з іншими галузями знань, такими як когнітивна психологія, антропологія, філософія, інформатика тощо. Вона допомагає нам розуміти, як люди взаємодіють один з одним через мову, як мова впливає на наш спосіб мислення і поведінку, і як ми можемо ефективно використовувати мову для досягнення різних цілей.

Гносеологічний аспект

Гносеологічний аспект статистичної лінгвістики фокусується на тому, як статистичні методи допомагають нам отримувати знання про мову. Використання корпусного аналізу є одним із способів цього підходу. Він полягає в дослідженні великих мовних наборів даних, таких як текстові документи або аудіозаписи, для виявлення статистичних закономірностей та розробки мовних моделей.
Гносеологічний аспект статистичної лінгвістики надає нам інформацію про структуру мови, семантику слів і фраз, граматику та вживання мовних одиниць. Наприклад, за допомогою статистичних методів можна виявити найпоширеніші слова у мовному корпусі, типові словосполучення та правила порядку слів у реченні. Цей підхід допомагає лінгвістам краще розуміти мову та її функціонування, а також розробляти нові методи для автоматичної обробки мови, машинного перекладу, розпізнавання мови та інших застосувань.

Об'єкт і суб'єкт вивчення

У статистичній лінгвістиці об'єкт вивчення охоплює мову та мовлення. Об'єктом дослідження можуть бути різні мовні явища, такі як лексика, граматика, фонетика, семантика, мовні структури та інші аспекти мови.
Суб'єкт вивчення у статистичній лінгвістиці - це самі лінгвісти, дослідники, які використовують статистичні методи для аналізу мови та отримання знань про неї. Лінгвісти застосовують статистичні моделі, алгоритми та інструменти для обробки мовних даних, виявлення статистичних закономірностей та формулювання теоретичних висновків.
Об'єкт і суб'єкт вивчення взаємодіють у статистичній лінгвістиці, де використовуються статистичні методи для аналізу мовних даних з метою зрозуміти структуру, функціонування та властивості мови. Ця взаємодія дозволяє лінгвістам отримувати нові знання про мову та розробляти прикладні рішення, такі як автоматичний переклад, розпізнавання мови, створення мовних моделей та інше.

1.2. Статистична лінгвістика

Статистична лінгвістика - це галузь лінгвістики, яка використовує математичні і статистичні методи для вивчення мови і виявлення закономірностей у мовленні. Вона є комбінацією лінгвістики та статистики, що дозволяє вивчати мову відносно об'єктивно і точно, використовуючи даний корпус текстів. Однією з головних задач статистичної лінгвістики є створення комп'ютерних систем, які здатні аналізувати і розуміти мовлення людей. Для цього вона використовує методи машинного навчання, які дозволяють програмі зрозуміти, які слова пов'язані між собою, які фрази є логічно взаємопов'язаними, і як вони можуть бути використані у реальному житті.
Статистична лінгвістика знаходить своє застосування у багатьох галузях, таких як пошукові системи, машинний переклад, розпізнавання мови, розуміння природної мови, аналіз текстів, семантичний пошук і багато іншого. Вона стає все

більш важливою в епоху швидкого розвитку інформаційних технологій та Інтернету. Наприклад, статистична лінгвістика дозволяє пошуковим системам аналізувати тисячі документів та шукати в них відповідність запиту користувача. Вона також використовується у машинному перекладі для створення більш точних перекладів з використанням статистичних моделей. Розпізнавання мови та

розуміння природної мови дозволяють комп'ютерам розуміти людську мову та відповідати на запити користувачів, що важливо у роботі голосових помічників та роботі з мовними інтерфейсами. Аналіз текстів та семантичний пошук допомагають знаходити спільні ознаки між текстами та шукати семантичні зв'язки між ними. У цілому, статистична лінгвістика забезпечує комп'ютерам можливість розуміти та обробляти природну мову, що дозволяє створювати більш ефективні та корисні інформаційні системи.

1.3. Комп’ютерна лінгвістика

Комп'ютерна лінгвістика - це галузь знань, яка поєднує мовознавство і комп'ютерні науки з метою розробки систем і програм, що працюють з мовою. Головна мета комп'ютерної лінгвістики полягає у вивченні природної мови і розумінні того, як мова працює, а також розробці програм та алгоритмів, що допомагають комп'ютерам розуміти, обробляти та генерувати мову. Комп'ютерна лінгвістика використовується в багатьох галузях, таких як автоматичний переклад, пошукові системи, синтез та розпізнавання мови, аналіз тексту та його класифікація, семантичний пошук та багато іншого. Однією з ключових галузей, де комп'ютерна лінгвістика знаходить своє застосування, є розпізнавання та синтез мови, що допомагає комп'ютерам взаємодіяти з людьми у більш природний спосіб. Крім того, комп'ютерна лінгвістика займається розробкою і вдосконаленням алгоритмів та програм для обробки мовних даних, таких як текстові корпуси, словники, граматичні правила та інші мовні ресурси. Ці дані можуть використовуватися для створення нових програм, які забезпечують швидке та точне аналіз та обробку мови.

9

Комп'ютерна лінгвістика є досить молодою галуззю знань, що активно розвивається з виникненням нових технологій та розвитком штучного інтелекту. Зараз вона є невід'ємною складовою багатьох інноваційних проектів та продуктів, які мають відношення до мови та мовного аналізу, таких як голосові помічники, системи автоматичного перекладу, розпізнавання мови, аналізу та синтезу мови, робототехніка, соціальні мережі, культурологічні дослідження та інше.
Комп’ютерна лінгвістика базується на методах та технологіях, які використовуються в інформатиці, математиці, лінгвістиці, філософії, когнітивній науці та інших галузях.

1.4. Закони Ціпфа

Закони Ціпфа - це набір статистичних закономірностей, які описують взаємозв'язок між частотою вживання слів у текстах та їх рангом. Ці закони були сформульовані американським лінгвістом Джорджем Ціпфом в 30-х роках XX століття.
Закони Ціпфа мають практичне застосування у багатьох галузях, зокрема, у лінгвістиці, криптографії, економіці та інформаційних технологіях. Вони також допомагають вивчати та аналізувати мову та текстові дані. Наприклад, на основі законів Ціпфа можна визначити найбільш уживані слова в тексті та порівняти їх частоту з відсотком рідкісних слів. Це дозволяє оцінити текст на природність та виявити можливі ознаки маніпулювання ним. У криптографії, за допомогою законів Ціпфа можна виявляти шифрограми, де букви замінені на інші символи. В економіці, ці закони застосовуються для вивчення розподілу багатства та нерівності в господарстві. У сфері інформаційних технологій, закони Ціпфа використовуються для побудови ефективних алгоритмів обробки текстових даних, таких як пошукові системи та машинний переклад.
Перший закон Ціпфа "ранг - частота". Вибирається будь-яке слово і підраховується, скільки разів воно зустрічається в тексті. Ця величина називається частота входження слова. Вимірюється частота кожного слова тексту. Деякі слова будуть мати однакову частоту, тобто входити в текст рівну кількість разів.

10

Згрупуємо їх, взявши тільки одне значення з кожної групи. Розташуємо частоти в міру їх зменшення та пронумеруємо. Порядковий номер частоти називається ранг частоти. Так, **найбільш** часто **зустрічаються** слова матимуть ранг **1, наступні за ними - 2 і** т.д. Аналітично закон Ціпф може бути виражений у вигляді
fr = **c, де f -** частота народження слова **в тексті; r -** ранг (порядковий номер) **слова в** списку;
c - емпірична **постійна величина.**

Отримана залежність графічно виражається гіперболою.
Дослідивши таким чином найрізноманітніші тексти і мови, в тому числі

мови тисячолітньої давності, Дж. Ціпф для кожної з них побудував зазначені залежності, при цьому всі криві мали однакову форму - форму гіперболічної драбини, тобто при заміні одного тексту іншим загальний характер розподілу не змінювався. Ймовірність зустріти слово (p) шляхом випадкового вибору, буде дорівнює відношенню частоти входження цього слова до загальної кількості слів у тексті.
p = f / n, де n - число слів.

Ціпф виявив цікаву закономірність. Виявляється, якщо помножити ймовірність виявлення слова в тексті на ранг частоти, то отримана величина (C) приблизно постійна!
C = (f x r) / n
Якщо трохи перетворити формулу, то можна побачити, що це функція y = k / x і її графік - рівнобічна гіпербола. Отже, за першим законом Ціпфа, якщо найпоширеніше слово зустрічається в тексті, наприклад, 100 раз, то наступне за частотою слово навряд чи зустрінеться 99 разів. Частота входження другого за популярністю слова, з високою часткою ймовірності, виявиться на рівні 50. Цей закон зберігається практично до всіх мов світу (Рис.1.4.2.).
Значення константи в різних мовах по-різному, але всередині однієї мовної групи

11

залишається незмінно, який би текст ми не взяли. Так, наприклад, для англійських текстів константа Ціпфа дорівнює приблизно 0,1. Російські тексти з точки зору законів Ціпфа не виняток. Для російської мови константа Ціпфа дорівнює 0,06-0,08.

Рис.1.1 Закон Ціпфа: Графік для частот слів зі статей російської Вікіпедії з рангами від 3 до 170

Другий закон Ціпфа "кількість - частота". Розглядаючи перший закон, факту, що різні слова входять в текст з **однаковою** частотою не розглядався. Ціпф встановив, **що частота і кількість слів, що входять в текст з цією частотою,** теж пов'язані між собою. **Якщо побудувати графік, відклавши** по одній **осі** (осі X) частоту **входження слова, а** по інший (**осі** Y) - кількість **слів у даній частоті,** то вийшла крива **буде** зберігати свої параметри для всіх **без** винятку створених людиною текстів! **Як і** в попередньому випадку, це **твердження вірне в межах однієї мови. Однак і міжмовні відмінності невеликі.** На якій би мові текст не був написаний, форма кривої Ціпфа залишиться незмінною. Можуть трохи відрізнятися лише коефіцієнти, що відповідають за нахил кривої (в логарифмічному масштабі, за винятком декількох початкових точок, графік - пряма лінія). Закони Ціпфа універсальні. В принципі, вони можуть бути застосовані не тільки до текстів. Характеристики популярності вузлів в мережі Інтернет - теж відповідають законам Ціпфа. Не виключено, що в законах відбивається "людське" походження об'єкта.

12

Рис.1.2 Закон Ціпфа: Графіки для частот слів зі статей Вікіпедії різних мов світу в логарифмічній шкалі

1.5. Закон Парето

Закон Парето, також відомий як закон 80/20, є одним з ключових понять у соціолінгвістиці та прикладній лінгвістиці. Цей принцип стверджує, що в будь-якому явищі або системі, близько 80% результатів (наприклад, використовуваних слів або говорінь) походять від 20% причин (наприклад, найбільш часто вживаних слів або говорів). У лінгвістиці, принцип Парето застосовується, наприклад, у дослідженнях розподілу слів в текстах та мовленні. Згідно з цим принципом, найбільш вживані слова становлять значну частину тексту або мовлення, тоді як менш вживані слова відіграють менш значну роль. Це дозволяє лінгвістам та мовознавцям краще розуміти природу мови та її ефективне використання.
Принцип Парето також використовується в прикладній лінгвістиці для визначення найбільш важливих елементів мови, які потрібні для ефективного вивчення мови або для автоматичного перекладу. Наприклад, якщо 20% слів використовуються в 80% випадків, то зосередження на вивченні цих слів може дати значні результати у вивченні мови.

13

Принцип Парето має застосування в мовному аналізі, стилістиці та

лінгвістичному проектуванні. За допомогою цього принципу можна виявити найбільш ефективні методи та засоби використання мови в конкретних контекстах.

1.6. Закон Гіпса

Закон Гіпса, також відомий як "закон довжини слова", є одним з ключових принципів прикладної лінгвістики та статистичного аналізу мови. Цей закон стверджує, що в мові більш довгі слова зустрічаються рідше, ніж короткі слова. Зокрема, він стверджує, що частота вживання слова в залежності від його довжини може бути описана математичною формулою, де частота вживання зворотньо пропорційна довжині слова. Закон Гіпса має практичне застосування в різних галузях, зокрема, в прикладній лінгвістиці, стилістиці та машинному навчанні. Він допомагає вивчати та аналізувати мову з точки зору її структури та властивостей.
У **комп'ютерній лінгвістиці емпіричний закон** Г.С. Гіпса (H. **S. Heaps) пов'язує обсяг документа з об'ємом словника унікальних слів, які входять в цей документ. Здавалося б, словник унікальних слів повинен насичуватися, а його обсяг стабілізуватися при збільшенні обсягів тексту. Виявляється, це не так! Для всіх відомих сьогодні текстів відповідно до** закону Гіпса, **ці значення пов'язані** співвідношенням:
$v(n) = \alpha n^{\beta}$,
де **v - це обсяг словника унікальних слів, складений з тексту, який складається з n унікальних** слів, **α і β - певні емпірично параметри. Для європейських** мов **α приймає значення від 10 до 100, а β - від 0.4 до 0.6.**

14

Рис.1.3 Типовий графік, який зображає закон Гіпса

Закон Гіпса не обмежується лише унікальними словами, але стосується багатьох інших інформаційних об'єктів. Це логічно, оскільки закон Гіпса є наслідком закону Ціпфа.

15

РОЗДІЛ 2. ПРОГРАМНА ЧАСТИНА

2.1. Об'єктно-орієнтовне програмування

Підхід до програмування, який базується на використанні "об'єктів", індивідуальних елементів програми, що містять дані та функції для роботи з цими даними, називається об'єктно-орієнтованим програмуванням (ООП). У ООП, програми розробляються з використанням класів, які визначають об'єкти, що можуть створюватися в програмі, і кожен клас описує характеристики та можливості цих об'єктів. Об'єкти можуть взаємодіяти один з одним через виклик методів, які визначені в класах. ООП дозволяє розбити програму на менші компоненти, кожен з яких може бути використаний в інших програмах, і забезпечує високий рівень абстракції та модульності. Цей підхід дозволяє розробникам створювати більш складні програми, ефективно підтримувати їх та взаємодіяти зі складними системами та базами даних. Мови програмування, які підтримують ООП, включають Java, C++, Python, Ruby, C#, Objective-C та багато інших. Об'єктно-орієнтоване програмування є важливим підходом до програмування в сучасному світі. ООП також забезпечує більш високий рівень безпеки програмного коду та полегшує його перевикористання. Крім того, ООП може бути використаний для моделювання реальних об'єктів та процесів, що дозволяє розробникам створювати більш точні та складні системи. ООП є підходом до програмування, який став основою для багатьох сучасних програмних систем та технологій, таких як веб-розробка, мобільна розробка та штучний інтелект. Розуміння основ ООП є важливим для будь-якого програміста, який бажає створювати складні програми, що працюють зі складними системами та базами даних. Загалом, ООП є потужним та ефективним підходом до програмування, який дозволяє розробникам створювати складні програмні системи та технології, які можуть працювати зі складними системами та базами даних. Цей підхід дозволяє розбити програму на менші компоненти, що полегшує їх розробку та підтримку, а також забезпечує більш високий рівень безпеки програмного коду та полегшує його перевикористання.

16

2.2. Мова програмування C#

C# (вимовляється "сі шарп") є об'єктно-орієнтованою мовою програмування,

розробленою Microsoft у 2000 році для розробки додатків для платформи Microsoft .NET. C# має високу продуктивність та ефективність, що робить його популярним в індустрії програмного забезпечення. В наш час, C# використовується для розробки різних додатків, від десктопних до веб-додатків та ігор.

1. Історія мови програмування C#

C# був розроблений в 1999-2000 роках командою під керівництвом Андерса Гейлсберга в рамках проекту Microsoft .NET Framework. Мета проекту полягала у створенні платформи, яка дозволяла більш ефективно розробляти програмне забезпечення для операційних систем Windows. Одним з ключових рішень, які були зроблені на початку проекту, було створення нової мови програмування, яка могла би бути використана для розробки додатків для .NET Framework.

Саме так з'явився C#. Назва мови програмування походить від музичного терміна C-sharp (до#). Перша версія C# була випущена у 2002 році разом з першою версією .NET Framework. З тих пір було випущено багато версій C#, остання з яких, на момент написання цього тексту, - C# 10, випущена в 2021 році.

2. Синтаксис мови програмування C#

Синтаксис мови C# в основному ґрунтується на синтаксисі мови Java, але він також має впливи від мов C і C++. Основна структура програми C# складається з класів, які містять дані та методи, які працюють з цими даними. Основними елементами синтаксису мови C# є:

- Змінні: змінні в C# визначаються за допомогою ключового слова "var" або вказуючи їх тип наприклад, int, string, bool тощо.
- Оператори: мова C# підтримує всі стандартні оператори, такі як арифметичні, логічні, порівняння тощо.

- Умовні конструкції: умовні конструкції в C# включають if-else, switch, тернарний оператор.
- Цикли: мова C# підтримує цикли while, do-while, for, foreach.
- Масиви: в C# можна використовувати масиви, які дозволяють зберігати більше одного значення у змінній.
- Класи: класи є основними будівельними блоками програми в C#. Класи включають дані та методи, які можуть бути використані для роботи з цими даними.
- Об'єкти: об'єкти в C# є інстансами класів, що містять дані та методи класу.
- Наслідування: наслідування дозволяє створювати новий клас, який успадковує властивості та методи класу-батька.
- Інтерфейси: інтерфейси в C# є контрактом між класами. Вони містять методи та властивості, які мають бути реалізовані класами, які реалізують інтерфейс.

3. Розробка програм на C#

Розробка програм на C# може виконуватися в різних середовищах, таких як Visual Studio, Visual Studio Code, SharpDevelop, MonoDevelop тощо. Середовище Visual Studio є одним з найпопулярніших середовищ для розробки програм на C# та має багато корисних інструментів, які дозволяють розробникам ефективно створювати та тестувати програми.

Розробка програм на C# включає кілька етапів:

- Розробка а лгоритму: на цьому етапі визначається логіка програми та встановлюються вимоги до її функціональності.

Розробка інтерфейсу: створення інтерфейсу користувача, який дає можливість користувачам легко взаємодіяти з програмою.

Написання коду: програміст використовує C# для написання коду програми згідно з вимогами, визначеними на попередньому етапі.

Тестування: після написання коду програма піддається різним видам тестування, включаючи модульні та інтеграційні тести, для перевірки правильності роботи програми.

Підтримка: після випуску програми її необхідно підтримувати, виправляти помилки та додавати нові функції.

4. Інструменти для розробки програм на C#

Для розробки програм на C# доступний ряд інструментів, включаючи:

Microsoft Visual Studio: це інтегроване середовище розробки (IDE), яке надає засоби для розробки, налагодження та тестування програм на C#.

Visual Studio Code: це безкоштовний текстовий редактор, який підтримує розширення для розробки програм на C#.

MonoDevelop: це IDE для розробки програм на C#, яке є відкритим джерелом та

доступне для Windows, macOS та Linux.
JetBrains Rider: це IDE для розробки програм на C# від JetBrains, який підтримує Windows, macOS та Linux.
5. Приклад програми на C#
Нижче наведений приклад простої програми на C#, яка виводить рядок "Hello, world!" у консоль:

```
using System;

19

class HelloWorld
{
static void Main()
{
Console.WriteLine("Hello, world!");
}
}
```
Ця програма складається з класу `HelloWorld`, який містить метод `Main()`. Метод `Main()` є точкою входу у програму. У цьому методі викликається метод `Console.WriteLine()`, який виводить рядок "Hello, world!" у консоль.
C# є мовою програмування, яка дозволяє розробляти широкий спектр програм, включаючи додатки для Windows, веб-сайти, веб-сервіси, мобільні додатки та інше. Його високоякісний синтаксис, розширена бібліотека класів та ефективна збірка сміття дозволяють розробникам писати код швидко та легко.
Крім того, мова C# є частиною платформи .NET, що дозволяє розробникам використовувати її для створення програм, які працюють на різних операційних системах, таких як Windows, macOS та Linux. .NET також дозволяє використовувати різні мови програмування, що дозволяє розробникам використовувати найбільш зручну мову для кожного конкретного завдання.
Наступні версії C# включають нові можливості та функціональність, такі як асинхронне програмування, LINQ (Language-Integrated Query), розширення для роботи зі зв'язками баз даних та багато іншого. C# також підтримує розробку програм з використанням патернів проектування та забезпечує багато засобів для реалізації об'єктно-орієнтованого програмування.

20

У загальному, мова C# є потужним інструментом для розробки програм, який дозволяє розробникам писати код ефективно та ефективно працювати з різними типами даних та ресурсів. З його допомогою розробники можуть створювати широкий спектр програм, які можуть працювати на різних платформах та забезпечувати потреби користувачів в різних сферах життя.
2.3. Microsoft .Net Framework

Microsoft .NET Framework є платформою розробки програмного забезпечення, розробленою Microsoft, що працює на операційних системах Windows. .NET Framework надає розробникам засоби для створення різноманітних додатків, включаючи консольні програми, десктопні програми, веб-додатки, мобільні додатки та інші.
1. Архітектура .NET Framework
.NET Framework складається з декількох компонентів, включаючи такі як:
- CLR (Common Language Runtime) - це віртуальна машина, що виконує код .NET. CLR відповідає за керування пам'яттю, безпекою, роботою з потоками і т.д. CLR гарантує, що код буде виконано в контрольованому середовищі, що зменшує можливість помилок та збій у програмі.
- CTS (Common Type System) - це система типів, яка визначає формати даних та їх поведінку. Ця система дозволяє взаємодіяти між різними мовами програмування, які підтримують .NET Framework.
- CLS (Common Language Specification) - це набір правил, які визначають мінімальний набір функцій та обмежень для мов програмування, які мають бути підтримані для взаємодії між різними мовами програмування в .NET Framework.
- BCL (Base Class Library) - це набір класів, що містять багато корисних функцій та об'єктів, які можуть бути використані в розробці програмного забезпечення.
2. Розробка програм з використанням .NET Framework

21

Розробка програм на .NET Framework включає такі етапи:
- Вибір мови програмування - .NET Framework підтримує декілька мов програмування, включаючи C#, VB.NET, F# та інші.
- Вибір середовища розробки - для розробки програм на .NET Framework можна використовувати різноманіті середовища розробки, такі як Microsoft Visual Studio, Visual Studio Code, JetBrains Rider, Xamarin Studio та інші. Найпоширенішим серед них є Microsoft Visual Studio, яке має широкі можливості для розробки різноманітних програм на .NET Framework.
Крім того, .NET Framework має ряд інструментів для побудови, установки та керування додатками, які підтримуються платформою, такі як Visual Studio Installer Projects, InstallShield, WiX Toolset та Advanced Installer. Ці інструменти допомагають розробникам створювати професійні інсталяційні пакети для своїх додатків на платформі .NET Framework.
Важливо зазначити, що .NET Framework має свої обмеження та недоліки, зокрема обмеження в підтримці платформ, низьку швидкодію в порівнянні з нативними додатками, обмежені можливості управління пам'яттю та інші. Однак, розробники продовжують використовувати платформу .NET Framework для розробки додатків, оскільки вона надає багато переваг в порівнянні з іншими платформами.
.NET Framework також має декілька важливих компонентів, які дозволяють розробникам створювати різноманітні додатки. Наприклад, **Common Language Runtime (CLR) є віртуальною машиною, яка** виконує код, написаний на **мовах** програмування, які підтримуються платформою . **NET Framework.** CLR також забезпечує **управління пам'яттю, обробку** винятків, оптимізацію коду та інші функції.
Інший важливий компонент - Class Library, який містить набір класів та інших ресурсів, які можуть бути використані при розробці додатків на .NET Framework. Class Library включає різноманітні класи для роботи з файловою системою, мережею, базами даних та багато інших корисних функцій. Крім того, .NET

22

Framework містить багато інших компонентів, таких як Windows Communication Foundation (WCF), Windows Presentation Foundation (WPF), Windows Workflow Foundation (WF) і Windows CardSpace.
WCF - це технологія, яка дозволяє створювати розподілені застосунки на основі послуг, які можуть бути використані через мережу. WCF надає можливість створювати послуги, які можна використовувати як на локальному комп'ютері, так і на віддаленому комп'ютері через мережу.
WPF - це технологія, яка дозволяє розробляти графічні інтерфейси користувача (GUI) для Windows додатків. WPF надає багато можливостей для створення складних та інтерактивних інтерфейсів користувача, включаючи візуальні ефекти, анімацію та 3D-графіку.
WF - це технологія, яка дозволяє створювати та виконувати бізнес-процеси на основі правил. WF дозволяє створювати послідовності кроків, які можуть виконуватися автоматично або з допомогою людини, та керувати різними видами діяльності.
Windows CardSpace - це технологія, яка дозволяє користувачам зберігати та керувати своїми персональними інформаційними картками, такими як паролі, сертифікати, адреси електронної пошти та інші деталі. Це дозволяє користувачам забезпечити безпеку своїх персональних даних та обмінюватися ними з іншими користувачами та системами.
Microsoft .NET Framework - це платформа, яка дозволяє розробляти та виконувати програми на мові програмування C# та інших мовах. .NET Framework містить багато компонентів, які дозволяють створювання високопродуктивних та безпечних програм з використанням різноманітних технологій, таких як Windows Forms, ASP.NET, WPF, ADO.NET та інших.
2.4. Visual Studio

23

Visual Studio - це інтегроване середовище розробки (IDE), розроблене компанією Microsoft для розробки програмного забезпечення. Visual Studio дозволяє розробникам створювати додатки для різних платформ, включаючи Windows, Android, iOS, Linux, macOS, Web та багато інших.
Visual Studio був випущений вперше у 1997 році компанією Microsoft і з того

часу він став одним з найпопулярніших інструментів розробки програмного забезпечення в світі. Він постійно оновлюється та вдосконалюється, що дозволяє розробникам створювати більш якісне та ефективне програмне забезпечення.
2. Основні функції Visual Studio
Visual Studio має безліч корисних функцій для підвищення продуктивності розробки. Основні з них:
- IntelliSense - це функція, яка надає підказки для кодування, що допомагає розробникам швидше та ефективніше писати код.
- Debugging - це функція, яка дозволяє розробникам відлагоджувати свій код, шукаючи та виправляючи помилки.
- Version Control - це функція, яка дозволяє розробникам зберігати та керувати версіями свого коду, що дозволяє зберегти прогрес та забезпечити легкий доступ до раніше написаного коду.
- Code Refactoring - це функція, яка дозволяє розробникам автоматично переписувати код, зберігаючи його функціональність, але поліпшуючи його читабельність та ефективність.
- Unit Testing - це функція, яка дозволяє розробникам тестувати свій код, щоб переконатися, що він працює правильно і не має помилок. Visual Studio має вбудовану підтримку для створення тестів одиниць, які можна виконувати в автоматичному режимі. За допомогою Unit Testing можна виявляти та виправляти помилки в коді на ранніх етапах розробки, що дозволяє зекономити час та зусилля в подальшій розробці.

6. Особливості роботи з Visual Studio
Visual Studio дозволяє працювати з різними мовами програмування, такими як C#, Visual Basic, C++, F#, Python та інші. Кожна мова має свої відповідні шаблони проектів та інструменти, які дозволяють розробникам створювати програми відповідно до їхніх потреб, також дозволяє збирати та керувати версіями програмного забезпечення за допомогою систем контролю версій, таких як Git та SVN. Це дозволяє розробникам працювати в команді та зберігати історію змін, що відбулися в проекті. Крім того, Visual Studio має вбудовану підтримку для роботи з різними хмарними сервісами, такими як Microsoft Azure, що дозволяє розробникам створювати та розгортати свої додатки в хмарі. Visual Studio також має інтегрований механізм дебаггінгу, який дозволяє розробникам відслідковувати та виправляти помилки в коді. Режим дебаггінгу дозволяє зупиняти виконання програми на певному кроці та досліджувати значення змінних та об'єктів в поточному контексті. Visual Studio - це потужне інтегроване середовище розробки, яке дозволяє розробникам створювати **програмне забезпечення на різних мовах програмування, таких як C#, Visual Basic,** C++ та інші
2.5. N-грами
Програма призначена для побудови статичних характеристик текстів, які описуються першим і другим законами Ціпфа, а також законом Парето. Докладнішу інформацію про реалізацію програми, її алгоритми та вхідні параметри можна отримати шляхом аналізу вихідного коду самої програми.
Перший блок
Перший блок у програмі, який представлений у файлі Program.cs, є важливим і початковим кроком додатку. Він виконує наступні дії:

1. Імпортує необхідні простори імен (`using` директиви): За допомогою `using` директив ми імпортуємо необхідні простори імен, які містять класи та функції, необхідні для роботи додатку. Це включає класи для роботи з формами, колекціями, роботи зі стрічками тощо.

2. Клас `Program`: Це статичний клас, який містить головну функцію `Main` і є точкою входу у вашу програму.
3. Атрибут `[STAThread]`: Цей атрибут вказує на режим однопотокової моделі COM, який необхідний для взаємодії з деякими стандартними діалоговими вікнами та компонентами Windows.
4. Метод `Main`: Це головний метод програми, який викликається при запуску програми. Він визначений з ключовим словом `static`, що означає, що його можна викликати без створення екземпляру класу. В цьому методі виконується початкова налаштування програми та запуск головної форми.
5. `Application.EnableVisualStyles()`: Цей метод дозволяє використовувати

візуальні стилі оперативної системи для елементів керування форми. Він надає вигляд вашому додатку, який відповідає стилю операційної системи, на якій він виконується.
6. `Application.SetCompatibleTextRenderingDefault(false)`: Цей метод встановлює режим сумісного відображення тексту за замовчуванням. Він визначає, чи буде використовуватись стандартний механізм відображення тексту, який може відрізнятись у різних версіях платформи .NET.
7. `Application.Run(new MainForm())`: Цей метод створює новий екземпляр головної форми `MainForm` і запускає основний цикл обробки подій для вашого додатку. Він забезпечує відображення головного вікна програми та обробку подій, таких як натискання кнопок, рух миші, ввід користувача тощо.

26

Цей перший блок коду Program.cs встановлює необхідні налаштування та запускає програму, підготовлюючи її до відображення головного вікна та обробки подій.

27

Другий блок
У другому блоку програми ми маємо клас `NGramm`, який представляє об'єкт N-грами. Цей клас містить властивості для збереження тексту, кількості та відносної частоти N-грами.
Клас `NGramm` має наступні властивості:
1. `Text` - текст N-грами, зберігається у вигляді рядка.
2. `Count` - кількість входжень N-грами у тексті, зберігається у вигляді цілого числа.
3. `RelativeFrequency` - відносна частота N-грами у тексті, зберігається у вигляді дійсного числа.
Клас `NGramm` використовується для представлення та обробки окремих N-грам у текстових даних. Він дозволяє зберігати інформацію про текст N-грами, кількість його входжень та відносну частоту. Цей клас є важливою складовою частиною програми, оскільки він надає можливість маніпулювати окремими N-грамами, аналізувати їх у контексті тексту та виконувати подальші обчислення з ними, наприклад, порівнювати N-грами за їхньою кількістю входжень або відносною частотою.
Третій блок
У третьому блоку програми ми маємо клас `Helpers`, який надає додаткові функції для обробки та сортування даних.
Клас `Helpers` має наступні методи:
1. `SortByVal(Dictionary&lt;string, int&gt; _strings)` - метод для сортування словників з рядками як ключами та цілими числами як значеннями за спаданням значень.
2. `SortByVal(Dictionary&lt;double, double&gt; _strings)` - метод для сортування словників з числами з плаваючою точкою як ключами та числами з плаваючою точкою як значеннями за спаданням значень.

28

3. `SortByVal(Dictionary&lt;int, int&gt; _strings)` - метод для сортування словників з цілими числами як ключами та цілими числами як значеннями за спаданням значень.
4. `SortByKey(Dictionary&lt;int, int&gt; _dict)` - метод для сортування словників з цілими числами як ключами та цілими числами як значеннями за зростанням ключів.
5. `SortByKey(Dictionary&lt;double, double&gt; _dict)` - метод для сортування словників з числами з плаваючою точкою як ключами та числами з плаваючою точкою як значеннями за зростанням ключів.
Ці методи дозволяють виконувати сортування різних типів словників за різними критеріями. Вони використовують методи LINQ для сортування та повертають нові словники, що містять відсортовані значення.
Четвертий блок
У четвертому блоку програми "NGramm" ми знайдемо код, який містить класи `Statistics`, `MainForm` і `ListForm`, а також методи `GetZipf1Stats`, `GetZipf1StatsL`, `GetZipf2Stats`, `GetParetroStats`, `GetHipsStats`. Цей блок відповідає за обчислення та збереження статистичних показників для аналізу N-грам, таких як статистика Ціпфа та статистика Парето, а також відображення результатів у графічному інтерфейсі користувача.

1. Клас `Statistics`:
   - Цей клас містить методи `GetZipf1Stats`, `GetZipf1StatsL`, `GetZipf2Stats`, `GetParetroStats` та `GetHipsStats`.
   - Методи `GetZipf1Stats`, `GetZipf1StatsL`, `GetZipf2Stats`, `GetParetroStats` та `GetHipsStats` виконують обчислення статистичних показників для N-грам.
   - Вони розраховують ранг, відносну частоту та інші характеристики для кожного N-грама, використовуючи різні формули та алгоритми.
   - Результати обчислень зберігаються та повертаються у вигляді словників.

2. Клас `MainForm`:
   - Цей клас відповідає за головне вікно програми та графічний інтерфейс користувача.
   - Він містить методи для створення та управління елементами форми, такими як кнопки, текстові поля, списки.
   - Використовуючи цей клас, користувач може взаємодіяти з програмою, вводити текст, обробляти N-грами, виконувати статистичний аналіз та отримувати результати.
3. Клас `ListForm`:
   - Цей клас відповідає за вікно зі списком N-грам та їх статистичними показниками.
   - Він містить методи для відображення списку N-грам, сортування за різними критеріями та відображення статистичних показників.
   - За допомогою цього класу, користувач може бачити результати статистичного аналізу N-грам у зручному форматі.
Таким чином, блок статистики та графічного інтерфейсу програми "NGramm" включає класи `Statistics`, `MainForm` та `ListForm`, а також методи `GetZipf1Stats`, `GetZipf1StatsL`, `GetZipf2Stats`, `GetParetroStats`, `GetHipsStats`. Вони спільно забезпечують обчислення та відображення статистичних показників для аналізу N-грам у зручному інтерфейсі користувача.

РОЗДІЛ 3. РОЗРОБЛЕННЯ ПРОГРАМИ

3.1. N-грами

Програма призначена для побудови статичних характеристик текстів, які описуються першим і другим законами Ціпфа, законом Парето, а також законом Гіпса.

Побудуємо характеристику тексту під назвою «Володар Перснів» автор Джон Роналд Руел Толкін та символьну характеристику за текстом «Дон Кіхот».

1. Відкриємо текст.

. рис. 3.1

2. Побудова характеристики тексту за буквами. Довжиною n-грам = 1

рис. 3.2

рис. 3.3

3. Побудова характеристики тексту за символами. Довжиною n-грам = 1

Рис. 3.4

Рис.3.5

4. Побудова характеристики тексту за словами. Довжиною n-грам = 1

Рис.3.6

34

Рис. 3.7

3.2. Закони Ціпфа
Дослідження текстів «Дон Кіхот» та «Робінзон Крузо» проведемо за законами Ціпфа. Також проілюструємо графік у логарифмічному та лінійному масштабі.

1. Відкриємо текст автора Мігель де Сервантес «Дон Кіхот».

рис 3.8

35

2. Реалізація першого закону Ціпфа при довжині n-грам = 1.

рис 3.10. логарифмічний масштаб

3. Реалізація першого закону Ціпфа при довжині n-грам = 1.

рис 3.11. лінійний масштаб

36

4. Реалізація другого закону Ціпфа при довжині n-грам = 1.

рис. 3.12. логарифмічний масштаб

5. Реалізація другого закону Ціпфа при довжині n-грам = 1.

Рис. 3.13. лінійний масштаб

37

6. Відкриємо текст автора Данієль Дефо «Робінзон Крузо».

Рис. 3.14.

7. Реалізація першого закону Ціпфа при довжині n-грам = 1.

Рис. 3.15 логарифмічний масштаб

38

8. Реалізація першого закону Ціпфа при довжині n-грам = 1.

Рис. 3.16. лінійний масштаб

9. Реалізація другого закону Ціпфа при довжині n-грам = 1.

Рис 3.17. логарифмічний масштаб

39

10. Реалізація другого закону Ціпфа при довжині n-грам = 1.

Рис. 3.18. лінійний масштаб

3.3 Закон Парето
Одне із досліджень проведемо за законом Парето. Цей закон ми реалізуємо на тексті під назвою **«Володар Перснів»** автор **Джон Роналд Руел** Толкін.

1. Відкриємо текст автора Джон **Роналд Руел Толкін «Володар Перснів».**

Рис 3. 19

40

2. Візуалізація закона Парето в тексті «Дон Кіхот» при довжині n-грам = 1.

Рис 3.20. лінійний масштаб

3. Візуалізація закона Парето в тексті «Дон Кіхот» при довжині n-грам = 1.

Рис. 3.21. логарифмічний масштаб

41

3.4 Закон Гіпса
Перевіримо тексти «Робінзон Крузо » , «Дон Кіхот», «Володар Перснів» на вміст унікальних слів по співвідношені об'єму документа за законом Гіпса.

1. Застосовуємо закон Гіпса в тексті «Робінзон Крузо».

Рис. 3.22. лінійний масштаб

42

Рис. 3.23. логарифмічний масштаб

2. Застосовуємо закон Гіпса в тексті «Дон Кіхот».

Рис. 3.24 логарифмічний масштаб

43

Рис. 3.25 лінійний масштаб

3. Застосовуємо закон Гіпса в тексті «Володар Перснів».

Рис. 3.26 логарифмічний масштаб

44

Рис. 3.27 лінійний масштаб

45

ВИСНОВКИ
В рамках дипломної роботи "Програма вивчення лінгвістичної статистики" було розроблено програмне забезпечення, призначене для аналізу лінгвістичної статистики тексту. Розробка проводилась з використанням мови програмування C# в середовищі Visual Studio з допомогою Microsoft .Net Framework на базі об'єктно-орієнтованого програмування.
В результаті було реалізовано функціонал для отримання статистичних даних про текст, таких як частота використання слів, символів, довжина речень, кількість різних слів та інших параметрів. Крім того, програма включає інтерактивний інтерфейс користувача, що дозволяє зручно працювати з отриманими результатами. Розроблене програмне забезпечення може бути використане в галузі лінгвістики та мовознавства для дослідження текстів різної природи, а також в інших галузях, де вивчення статистики тексту є важливим етапом дослідження. В цілому, розробка програми дозволила поглибити знання про обробку тексту та мови програмування C# на прикладі розробки корисного програмного забезпечення.

46

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ
1. G. Zipf, Selective Studies and the Principle of Relative Frequency in Language (Cambridge, Mass, 1932);
2. Апресян Ю.Д. Вибрані праці, том II. Інтегральне опис мови і системна лексикографія. - М.: Школа "Мови російської культури", 2005.
3. Попов Е.В. Спілкування з ЕОМ на природній мові. М. Наука. 2000.
4. Основи статистичної лінгвістики: Навчально-методичний посібник / Відп.

ред. проф. Ф.С. Бацевич.— Видавничий центр ЛНУ імені Івана Франка, 2008.—124 с.
5. Кочерган М. П. Загальне мовознавство: підручник / Михайло Петрович Кочерган. — Київ: Академія, 2003. — С. 398.6.
6. В. С. Перебийніс. Математична лінгвістика // Українська мова : енциклопедія. — К. : Українська енциклопедія, 2000. — ISBN 966-7492-07-9.
7. Бук С. Основи статистичної лінгвістики: Навчально-методичний посібник / Відп. ред. проф. Ф. С. Бацевич.— Львів: Видавничий центр ЛНУ імені Івана Франка, 2008.— 124 с.
8. Дарчук Н. П. Комп'ютерна лінгвістика (автоматичне опрацювання тексту): підручник.— К.: Видавничо-поліграфічний центр “Київський університет”, 2008.— 351 с.
9. Карпіловська Є. А. Вступ до комп’ютерної лінгвістики.
10. Ланде Д. В. Елементи комп’ютерної лінгвістики в правовій інформатиці. — К.: НДІІП НАПрН України, 2014. — 351 с. — ISBN 978-966-2344-33-2
11. Математична лінгвістика : навч. посіб. Кн.1 : Квантитативна лінгвістика / В. В. Пасічник, Ю. М. Щербина, В. А. Висоцька, Т. В. Шестакевич ; за ред. В. В.
12. Статистичні параметри стилів. К., 1967: Головин Б. Н. Мова і статистика. М., 1971;