Plag

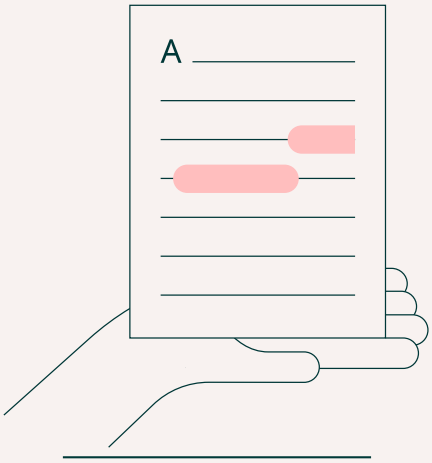
Звіт про оригінальність

● Оцінка схожості %11

● Ризик плагіату НАЙВИЩИЙ

Olga Kagalo 2025-06-19 22:49

Посилання на звіт: 10mBk / Посилання користувача: qEAc



Ось вона –
Ваша звіт про оригінальність!

Ми раді повідомити, що перевірка вашого документа завершена, і результати вже готові! Наші алгоритми старанно працювали, щоб знайти збіги в наших базах даних.

На наступних сторінках ви знайдете результати перевірки:

Бали
Збіги
Посилання

Ваш документ було перевірено за такими джерелами:

- ✓ База даних інтернет-джерел
- ✓ База даних наукових статей
- ○ Глибока перевірка (наш вдосконалений алгоритм)

Бали

Збіги тексту	11%
Перефразування	0%
Цитований текст	0%
Неправильне цитування	0%
Збігів не знайдено	89%

Ризик плагіату НАЙВИЩИЙ

Ризик плагіату вказує, як збіги тексту розподілені по документу. Вищий ризик виникає, коли збіги з'являються близько один до одного, наприклад, у тому самому абзаці або розділі.

Оцінка схожості %11

Оцінка схожості показує, скільки слів або символів у вашому документі збігаються з текстами інших документів, включаючи перефразовані тексти або неправильні цитати.

Збіги

Вступ

10 Чат-бот - це віртуальний асистент, 20 який спілкується 10 з користувачами за допомогою повідомлень і може мати 20 безліч специфічних функцій. Чат-боти 10 можна використовувати як для розсилки інформації, так і для її збору та 10 обробки.

9 Найбільшу популярність чат-боти отримали, коли почалося їх використання в месенджерах і соціальних мережах (наприклад, в Telegram, Viber, Facebook).

9 Telegram-бот – це аккаунт Telegram, різновид чат-бота, керований програмним забезпеченням, вони можуть інтегруватися з іншими службами або навіть передавати команди інших сервісів.

Типові Telegram-боти 14 відповідають на спеціальні команди в персональних і групових чатах, також вони можуть здійснювати пошук в інтернеті або виконувати інші завдання.

12 Таким чином, сфера застосування 12 Telegram-ботів 12 безмежна. Компанія Uber, безліч 12 банків, великі магазини і 12 багато інших організацій використовують Telegram-ботів для спрощення та автоматизації внутрішніх робочих процесів.

12 Метою цієї дипломної 9 роботи є створення Telegram-бота для коледжу телекомунікацій та комп'ютерних технологій, з функціями вибірки інформації про поточне знаходження студентів та викладачів під час навчальних годин на основі загальнодоступного розкладу занять. Окрім цього планується запровадити і інші функції, які можуть знадобитись студентам та викладачам коледжу.

Актуальність цієї дипломної роботи зумовлена швидким зростом популярності месенджерів і таких засобів автоматизації як чат-боти 18 серед користувачів мережі інтернет.

18 Аналіз предметної області

На сьогоднішній день, якщо грамотно використовувати технологію чат-ботів, можна реалізувати повноцінні магазини в месенджерах, технічну підтримку, і навіть внутрішні корпоративні аналітичні інструменти (робота з внутрішньою інформацією компанії).

Доцільність використання чат-ботів

Існує тип завдань і проблем, з якими чат-боти справляються краще за все. Чат-боти можуть взяти на себе ряд таких завдань: сегментація користувачів, оптимізація одноманітних завдань, цілодобова і безперебійна підтримка користувачів, продажі, виключення людського фактору, інтеграція інформаційних систем та ін.

1.1.1 Сегментація користувачів

Якщо існує кілька пропозицій для різних цільових аудиторій, відмінно працює бот, який проводить опитування і сегментує підписників 18 в залежності від реакції на той чи інший контент. Ці дані використовуються, щоб в подальшому автоматично проводити внутрішню сегментацію підписників і розсилати максимально індивідуальні пропозиції.

Наприклад, https://t.me/amicoIHT_bot - чат-бот виробника БАДів і мережі медичних консультаційних центрів. Цей бот задає питання, які виявляють болю аудиторії і дають можливість підібрати потрібні товари з асортименту.

Бот запитує про вагу, зріст, вік, режим харчування, сну і інше. Варіанти відповідей він записує в «ярлики», за якими потім проводиться тематична розсилка. Наприклад, пропозиція з курсом лікування від гіпертонії йде тільки користувачам із зайвою вагою старше 40 років і малорухливим способом життя, а курс підвищення імунітету тільки користувачам, які хворіли ГРЗ більше 3 разів за останні півроку. У будь-який момент менеджер може перехопити спілкування у бота і продовжити його у тому ж самому діалозі, без усіляких незручностей для користувача.

1.1.2 Оптимізація одноманітних завдань

2 До таких завдань належить, наприклад, перевірка відповідності документу від користувача прийнятим 1 нормам. Чат-бот може сам перевірити документ, знайти помилки, написати користувачеві і попросити його виправити рахунок і повідомити менеджеру, коли рахунок потрібного формату буде готовий. Точно так само чат-бот може обробляти заяви на відпустку, збирати у співробітників звіти про відпрацьовані години і зводити їх в таблицю.По суті, 2 будь-яка операція, яка проводиться по 1 строго окресленому алгоритму 2 і не вимагає від співробітників 1 креативних рішень може бути доручена чат-боту.

1.1.3 Цілодобова і безперебійна підтримка користувачів

Забезпечення зворотнього зв'язку с користувачами 24 години на добу в будь-яких каналах. Це, мабуть, найочевидніша перевага чат-ботів. Чат-бот буде працювати доки працює мессенджер і сервер, на якому розміщений сам чат-бот. Очевидна вигода з цього – це надійність та економія коштів.

1.1.4 Продажі

На відміну від додатків або сайтів, в месенджерах спілкування ведеться за допомогою діалогу, і людям не потрібно вивчати новий інтерфейс. Більшість великих месенджерів вже випустили або планують випустити рішення, що дозволяють, не виходячи з чату, здійснювати безпечну оплату товарів і послуг. Тому месенджери стають новим каналом продажів і маркетингу, а інструментом для цього каналу якраз і є чат-боти.

1.1.5 Виключення людського фактору

Виключення людського фактору з комунікації. Ця функція чат-ботів вже показала себе в HR. Вони беруть на себе роль анонімного і безособногго співрозмовника, щоб прибрати з процесу зайві емоції. Це стосується і таких завдань, як збір зворотного зв'язку, примус співробітників до своєчасної здачі звітів, щорічна оцінка персоналу і так далі. Компанія Amazon вже кілька років використовує роботів для таких задач.

1.1.6 Інтеграція інформаційних систем

Чат-боти легко інтегруються з будь-якими інформаційними системами. Це означає, що співробітники можуть спілкуватися з усіма цими системами через чат-бота. Їм не потрібно навчатися використанню різних інтерфейсів і витрачати час на введення або пошук даних в різних системах.

Все частіше люди використовують месенджери як найшвидший і простий спосіб спілкуватися в середовищі, де немає нічого зайвого. І тепер, таким чином, можна комунікувати не тільки з друзями, але і з брендами, отримувати доступ до найбільш необхідних послуг, минаючи пошукові системи і сайти

Чат-бот може виконувати роботу асистента - аналізувати дані, створювати звіти, заповнювати форми, задаючи власнику навідні запитання. Цими здібностями ботів користуються, наприклад, фінансисти, готуючись до нарад, або рекрутери, використовуючи чат-ботів для того, щоб кандидати могли в режимі діалогу заповнювати форми і не кидали цей процес.

Сфери застосування чат-ботів

На рис. 1.1 зображений ландшафт індустрії використання чат-ботів, створений

американським сервісом KeyReply у 2017 році.

Рисунок 1.1 – Ландшафт індустрії використання чат-ботів

Компанії розподілені по осях залежно від функцій, які виконують їх боти (з правої сторони – підтримка, з лівої - маркетинг), і від того, де вони створені (знизу - в самій компанії, зверху - у студії на замовлення).

Коло в центрі - це популярні месенджери, які дозволяють створювати ботів через надійні API-інтерфейси.Наступне коло Brands - компанії, які запустили чат-ботів вказаного типу.

Providers - компанії, які допомагають брендам створювати ботів, а Tools - інструменти, якими користуються провайдери, бренди або розробники для запуску чат-ботів.

Сфери застосування чат-ботів необмежені, проте найбільш популярні з них: сфера продажів, служби підтримки, служби доставки, таксі, засоби масової інформації, банківська сфера

Сфера продажів

Чат-боти можуть створюватися для автоматизованого продажу квитків (у кіно, театр, на концерт, на транспорт), одягу, косметики та інших. Чудовим прикладом запровадження такого чат-бота в Україні є telegram-бот RailWayBot. Він пропонує купити квитки приблизно так само, як і на офіційному сайті Укрзалізниці, проте, якщо квитки на необхідний напрямок відсутні, цей бот дозволяє провести моніторинг. Кожні пять хвилин він буде провіряти наявність квитків, і якщо хтось здасть білет, бот надсилає користувачеві повідомлення про те, що квиток тепер доступний до продажу.

1.2.2 Служби підтримки

Використання чатботів 7 для служби підтримки - повсюдно впроваджувана практика. Автоматизація процесів спілкування з клієнтами допомагає заощадити безліч часу і сил «живих» працівників компанії і виділити їм більше часу на вирішення складних завдань.

7 Чат-бот “Зоряна” мобільного оператора Київстар щомісяця заощаджує близько 20 тисяч доларів, обробляючи всього 10% запитів, які надходять в компанію. Щомісяця чат-ботом “Зоряна” користуються більше 100 000 користувачів в месенджерах Facebook , Telegram і Viber.

Схожі чат-боти використовує Державна міграційна служба України, Нова Пошта, оператор Lifecell, Укрпошта та безліч інших великих українських компаній.

Автоматичні співрозмовники також 1 можуть проводити аналіз статистики розмов і знаходити зони в системах компанії, які найчастіше створюють проблеми для користувачів.

1 1.2.3 Служби доставки та таксі

Чатботи по доставці (квітів, подарунків і т.д.) користуються у клієнтів дуже великою популярністю. І пальму першості тут тримає служба доставки їжі.

Наприклад, у піцерії Domino є свій чатбот. Щоб замовити піцу, потрібно ввести в чаті слово pizza, і бот на основі історії повідомлень автоматично підбере бажаний користувачем тип цієї італійської страви. Якщо замовник хоче якусь конкретну піцу, йому потрібно буде вказати свої переваги в налаштуваннях. Звичайно ж, перед цим необхідно ввести свій номер телефону та адресу.

Чатботи можна використовувати і службам виклику таксі, - вони роблять процес замовлення і очікування машини зручнішим для клієнтів. Популярний зараз сервіс Uber використовує чат-бот в Facebook Messenger: за допомогою нього клієнт може викликати таксі і простежити за траєкторією його руху, - так він буде знати певно, що за ним їдуть і скільки часу залишилося чекати.Слідом за Uber чат ботів розпочали відкривати і інші компанії з цієї сфери, також з’явились боти-агрегатори таксі.

1.2.4 Засоби масової інформації

Чатботи використовуються багатьма газетами, журналами і телеканалами. Наприклад, бот від CNN показує статті з обраної користувачем теми, а при підписці щодня радує читача новинами і свіжими публікаціями.

Також у Telegram з’явився бот MediaMonitoringBot, 3 який моніторить згадки в українських інтернет-виданнях. Він повідомляє користувача про згадки в медіа не пізніше, ніж за п'ять 3 хвилин після появи матеріалу.

3 Щоб запустити моніторинг, потрібно додати ключове слово, за яким користувач хоче відстежувати згадки. Наприклад, назва компанії чи бренду, прізвище засновника і т.д. У разі виходу нового матеріалу з ключовим словом, бот надішле повідомлення про це. У повідомленні буде зазначено, яке саме ключове слово було згадано, посилання на матеріал, кількість переглядів та потенційний прогноз охоплення публікації.

1.2.5 Банківська сфера

У банківській сфері чат-боти використовують найбільше. Банки дають можливість в своїх чат-ботах шукати банкомати і відділення, які знаходяться поруч, відстежувати курси валют та ін.

“Приватбанк” має 6 різних чат-ботів. Вони дозволяють перечислювати гроші безпосередньо через месенджер з прив’язкою до Приват24, приймають заявки, дозволяють оформлювати кредити та проводять консультацію користувачів.

По заявленням експертів, чат-боти «ПриватБанка» можуть обробляти близько 85% звернень, що дозволило вкоротити кількість операторів і дозволяє економити близько $2,2 млн на рік.

Також чат-ботами обзавелись такі українські банки як: “Альфа-банк”, “Райффайзен-банк”, “Таскомбанк”, “ОТП-Банк”, “Ощадбанк” та інші.

1.3 Класифікація чат-ботів

1.3.1 Класифікація чат-ботів за складністю

1. Прості. Функціонують на основі правил. Такий тип чат-ботів, як

правило, реагує на точні і конкретні команди. Якщо ж послати невірний або некоректний запит, то система не зможе його зрозуміти.

2. Складні. Функціонують на основі машинного навчання. даний тип

володіє штучним інтелектом. Штучний інтелект – це 16 науковий напрям, 3 в рамках якого ставляться 16 і вирішуються завдання апаратного або програмного моделювання тих видів людської діяльності, які традиційно вважаються інтелектуальними .

Працюючи з ним, вже немає необхідності максимально точно і конкретно писати свій запит, так як системи другого типу розуміють не тільки вузькі команди, але і саму природню мову. В результаті спілкування з людиною, такий чат-бот стає розумнішими.

1.3.2 Класифікація чат-ботів за функціоналом

1. Комунікаційні чат-боти. До них відносяться всі боти, які виступають в ролі консультантів і спілкуються з клієнтами від імені компанії, або ж внутрішньокорпоративні боти, які структурують і спрощують процес комунікації між співробітниками. Також до них відносяться боти, які виконують маркетинг-функції.

2. Функціональні чат-боти. До них відносяться всілякі боти-помічники, в тому числі - розважальні. Їх функції безмежні: знайти, забронювати, замовити, купити, порівняти, підібрати, розважати, навчати та інше. Бот, створений у рамках цього дипломного проекту, відноситься саме до цього виду.

1.3.3 Класифікація чат-ботів за інтерфейсом

1. Кнопкові чат-боти. Прості в освоєнні, але малофункціональні. Адже якщо зробити панелі зі складними, розгалуженими меню, розібратися в них буде зовсім нелегко. У цих ботів ініціатором діалогу виступає бот, а користувач для отримання необхідної йому допомоги, зобов'язаний послідовно виконувати вимоги свого робота-співрозмовника.

2. Розмовні (текстові) боти. Вони реагують на репліки співрозмовника і можуть вести з ним діалог. Розмовні чат-боти можуть бути засновані на штучному інтелекті або на мовній моделі та правилах.

Розмовні чат-боти з штучним інтелектом здатні навчатися. Але для цього потрібні величезні обсяги спеціально підготовлених даних, зібрати які під силу далеко не кожному банку. Тому через недостатність навчання такі чат-боти можуть вести себе непередбачувано, неправильно відповідати на питання, невірно оцінювати наміри клієнта і т.п.

3. Комбіновані чат-боти. Виходячи з усього вищесказаного, очевидно, що оптимальний варіант бота – це “Гібрид”, у якому, наприклад, частина функцій винесена на кнопки, але при цьому вони паралельно ведуть діалог зі співрозмовником і виконують текстові команди. До цього виду відноситься бот, створений у рамках цього дипломного проекту.

Класифікація чат-ботів за платформоювпровадження

Чат боти, впроваджені у месенджерах та соціальних мережах (Telegram, Facebook Messenger, Viber, VK)

Чат боти, впроваджені у мобільному застосунку (FakeTalk, Neiron, SimSimi)

Чат боти, впроваджені на веб-сайтах (при вході на безліч сайтів відкривається вікно чату, де користувача консультує бот, а не оператор підтримки.)

Чат боти, що відповідають на смс-запити (СМС-служби банків і мобільних операторів, впроваджені найдавніше, на сьогоднішній день поступово відходять на другий план)

1.4 Опис засобів розробки

1.4.1 Середовище розробки Microsoft VisualStudio 2019 Community

Microsoft VisualStudio 2019 Community – це спеціальна безкоштовна версія Microsoft VisualStudio для використання в навчальних цілях, для розробки проектів з відкритим кодом та у п’яти копіях для використання в некорпоративних організаціях з річним доходом меншим, ніж 1 млн. доларів США, та кількістю ПК, меншою ніж 250 одиниць.

Також існують версії Professional та Enterprise, проте вони досить дороговартісні і жодна з її переваг не зможе знайти застосування у цьому проекті.

Основні переваги цього IDE:

-Community версія є абсолютно безкоштовною і не має термінів дії

-Вбудована підтримка ASP.NET, Javascript, Python, C#/, C++, F#, Java, TypeScript, VisualBasic.

-Підказки IntelliSence дозволяють набагато швидше і зручніше писати код, також є підказки, що вказують на рекомендовані дії по вдосконаленню існуючого 6 коду, наприклад пропонують перейменувати функцію, створити параметр, відформатувати фрагмент коду тощо.

-CodeLens 6 допомагає легко отримувати важливі аналітичні відомості, наприклад про внесені в код зміни, про їх наслідки та про модульне тестування методів.

-Зручні вбудовані інструменти налагодження.

-Зручний 17 інтерфейс для роботи з наборами тестів допомагає впорядковувати дані, аналізувати обсяг тестованого коду і миттєво бачити результати. Функції тестування коду прямо під час введення дозволяють швидко дізнаватися наслідки кожної внесеної зміни.

-Вбудовані функції роботи з системами контролю версій

2 Розробка інформаційного чат-бота

2.1 Реєстрація чат-бота у Telegram

Для розробки Telegram-бота для початку його необхідно зареєструвати у сервісі Telegram. Для цього існує спеціальний Telegram-бот під назвою «BotFather». Приклад реєстрації Telegram-бота 4 зображений на рисунку 2.1.

Рисунок 2.1 – Приклад реєстрації Telegram-бота

Щоб розпочати реєстрацію, необхідно ввести команду «/newbot», після чого ввести довільну після чого ввести довільну назву (name) для свого бота. Ця назва буде лише декоративною і виводитись 17 для користувачів. Після введення назви необхідно ввести його ім’я (username). Ім’я чат-бота обов’язково мусить закінчуватись на «Bot» або «_bot», це ім’я буде використовуватись у посиланні на бота. Після завдання назви та імені, «BotFather» видає токен. 11 Токен – це унікальний набір символів для доступу до TelegramBot API.

Тепер можна встановити додаткові параметри, хоча вони не є обов’язкоми.Для налаштування додаткових параметрів існують спеціальні команди, описані у таблиці 2.1.

Таблиця 2.1. – Перечислення та опис команд 11 BotFather

11 Команда

11 Опис команди

11 /newbot

11 Створити нового бота

11 /mybots

11 Переглянути 11 список 11 ботів

11 /setname

11 Змінити ім’я

11 /setdescription

11 Задати чи 11 змінити опис бота. Опис бота буде виводитись при першому його відкритті, після фрази “Whatcanthisbotdo?” (Що може робити цей бот?)

/setabouttext

Задає чи змінює текст у полі “Про бота:”

/setuserpic

Задає картинку - “аватар” для бота.

/setcommands

Задає підсказку зі списком команд і їх призначенням для користувача

/deletebot

Видаляє бот

/setinline

Вмикає чи вимикає Inline-режим для бота. У Inline-режимі бота можна буде викликати із

будь-якого чату. При його згадуванні до бота прийде спеціальний API-запит.

/setinlinefeedback

Дозволяє отримувати інформацію про кількість обраних користувачами команд

/setinlinegeo

Дозволяє або забороняє у Inline-бота можливість отримувати геолокацію користувачів

/setprivacy

Вмикає або вимикає режим, у якому бот спілкується зі всіма користувачами групового чату лише приватно

/setjoingroup

Дозволяє або забороняє боту бути приєднаним то групового чату

/token

Отримати токен бота

/revoke

Видалити токен бота

2.2. Проектування моделі взаємодії клієнта і серверу

Для початку необхідно спроектувати модель взаємодії чат-бота з телеграм-сервером. Існує два способи отримання повідомлень від Telegram: LongPolling та WebHooks. На рисунку 2.2. зображено схему взаємодії чат-боту, серверу Telegram і клієнта за допомогою LongPolling.

.

Рисунок 2.2 – Схема взаємодії чат-боту, серверу Telegram і клієнта з використаннямLongPolling

Якщо з’явилось повідомлення, чат-бот його обробляє і надсилає відповідь до серверу Telegram, який в свою чергу надсилає її до клієнта. Недоліком цього варіанту є те, що бот буде функціонувати лише тоді, коли на ПК запущений консольний застосунок цього бота.

На рисунку 2.3 зображено схему взаємодії чат-боту, серверу Telegram і клієнта за допомогою методу SetWebHook().

Суть моделі з використанням вебхуків 4 полягає в тому, що чат-бот поміщається на домен. Сервісу Telegram повідомляється адреса цього домену, і він створює для себе строгу відповідність бота і хостингу.

Коли клієнт надсилає боту повідомлення, сервіс Telegram самостійно звертається до бота на хостингу, щоб той обробив повідомлення. Бот обробляє повідомлення і надсилає відповідь до сервісу Telegram, який в свою чергу направляє цю відповідь до клієнта.

Рисунок 2.3 - Схема взаємодії чат-боту, серверу Telegram і клієнта з використанням WebHooks

Модель взаємодії чат-боту і серверу Telegram за допомогою вебхуків є надійнішою, проте у цій роботі використана модель взаємодії з використанням LongPolling, оскільки бюджет не дозволяє розмістити чат-бота на домені.

Проте, якби ця можливість з'явилась, це можна буде швидко реалізувати, змінивши кілька стрічок коду і повідомивши BotFather адрес домену після розміщення бота на домені.

2.3 Написання програми

Для початку написання програми необхідно було створити проект як .Net консольний застосунок, після чого встановити NuGet-пакет під назвою Telegram.bot. У цій роботі використано найновішу 4 на даний момент версію цієї бібліотеки (15.5.1).

Telegram.bot – це бібліотека, яка є оболонкою над запитами до TelegramBot API. Використання цієї бібліотеки дозволяє у певній мірі спростити код програми, оскільки не потрібно з нуля створювати безліч классів і функцій із прямим зверненням до TelegramBotApi.

Після встановлення цього пакету, його було підключено до Program.cs

На рисунку 2.4 зображено список підключених бібліотек

Рисунок 2.4 – Список підключених бібліотек

Варто наголосити, що директива using – не те саме, що директива include в інших С-подібних мовах, і підключених бібліотек насправді не сім, а дві. Ця директива просто розгортає простір імен, даючи можливість звертатися до функцій і методів без префікса імені простору.

Коли бібліотеки підключені, можна приступати до оголошення змінних контексту класу,

які в ООП є аналогом глобальних змінних у імперативному програмуванні. Зазвичай такі змінні не є бажаними в ООП, та у цьому випадку вони допустимі. На рисунку 2.5 зображено фрагмент коду з оголошенням змінних контексту класу.

4 Рисунок 2.5 – Оголошення змінних контексту класу

botClient – це екземпляр об'єкту, що реалізовує інтерфейс ITelegramBotClient. Інтерфейс ITelegramBotClient є інтерфейсом над запитами до TelegramBotApi, вньому описано безліч різних подій (наприклад, отримання повідомлення чи натискання Inline-кнопки), задач (наприклад, відправлення повідомлення з текстом чи медіа, дії над учасниками чату чи запити на отримання оновлень) та змінних (наприклад, час очікування для LongPolling чи унікальний ідентифікатор бота). Цей інтерфейс описаний у просторі імен Telegram.Bot, його опис продемонстровано на рисунку 2.6.

4 Рисунок 2.6 – Інтерфейс ITelegramBotClient

Змінна контексту класу d буде отримувати своє значення 4 в залежності від поточного дня тижня, це необхідно для реалізації більшості функцій. Цю змінну було створено тому, що ці функції для виводу інформації будуть використовувати масиви, один з індексів яких відповідає дню тижня. В цьому випадку пряме використання методу DayOfWeek неможливе, і було необхідно створити саме змінну типу int для ідентифікації дня тижня.

Після цих змінних оголошується три масиви. 4 На рисунку 2.6 їх оголошення є згорнутим:

-Teachers - викладачі

-Groups - групи

-Students – студенти

Для початку, потрібно обґрунтувати необхідність введення цих масивів як глобальних змінних. Даний випадок – це дуже рідкісне виключення, коли такі дії є доцільними в ООП. Це виключення пов'язано з тим, що при створенні telegram-бота ми маємо лише два метода, вкладених у клас program: метод Main та метод BotOnMessage.

Метод BotOnMessage підписаний на подію botClient.OnMessage, іншими словами, цей метод виконується кожного разу, коли виникає подія botClient.OnMessage.

Через певні особливості мови програмування C# створювати змінні t і d всередині методу BotOnMessage не має сенсу, оскільки функції не могли би з ними працювати. В іншому випадку ці змінні можна було би вписати в параметри методу BotOnMessage, та 6 в даному випадку це неможливо, оскільки параметри методу BotOnMessage є суворо

регламентованими.

На початку проектування 6 для зберігання даних про викладачів, студентів і групи планувалось використовувати базу даних, та пізніше було складено такий спосіб зберігання цих даних, при якому їх автоматизована вибірка буде найбільш раціональною, а структура максимально інтуїтивно зрозумілою, щоб будь-яка людина при бажанні змогла відредагувати ці дані. В цьому способі дані зберігаються в масивах 15 масивів.

15 Масив масивів - це масив, елементи якого самі є масивами. Елементи масиву масивів можуть мати різні розміри та вимірність. Масиви масивів також називають нерегулярними масивами. Масиви масивів варто відрізняти від паралельних масивів. Паралельні масиви вважаються ознакою погано написаного коду і піддаються критиці зі всіх сторін, і не безпідставно, оскільки паралельні масиви насправді не пов’язані між собою, займають багато місця і знаходяться у різних комірках пам’яті.

На рисунку 2.7 зображено фрагмент масиву teachers

4 Рисунок 2.7 – Фрагмент масиву teachers

Кожен підмасив teachers[i] вказує на викладача, прізвище якого вказано у елементі teachers[0][0,i]. Рядки вказують на день тижня, а стовпці на пару. Наприклад, елемент teachers[1][0,0] вказує на першу пару в понеділок викладача за прізвищем teachers[0][0,1], тобто за прізвищем Семянистий, оскільки індексація розпочинається з нуля. Також варто згадати за нововведену константу V, яка вказує на відсутність пари.

На рисунку 2.8 зображений фрагмент масиву groups. Його структура масиву майже повністю аналогічна масиву teachers.

Різницю структури масиву groups від масиву teachers складає лише той факт, що в першому підмасиві масиву groups описуються не викладачі, а групи.

4 Рисунок 2.8 – Фрагмент масиву groups

Його підмасиви також є двовимірними, індексація також аналогічна: елемент groups[1][0,0] вказує на першу пару в понеділок групи groups[0][0,1], тобто ЗІ-12.

Від цих двох масивів досить сильно відрізняється за структурою масив students. Його фрагмент зображено 4 на рисунку 2.9

4 Рисунок 2.9 – Фрагмент масиву students

Перший підмасив цього масиву пустий по тій причині, що немає сенсу його

заповнювати, оскільки в такому випадку він у повній мірі повторював би перший підмасив масиву groups. Повторювати його сенсу немає, так як у місцях, де необхідно звернутись до першого підмасиву масиву students ми можемо просто звертатись до першого підмасиву масиву groups.

Структура цього масиву досить проста: всі його підмасиви є одновимірними, в них описуються прізвища студентів групи, зазначеної в коментарях. Кожен підмасивstudents[i] описує прізвища студентів групи groups[0][0,i]. При перебиранні цього масиву циклом for необхідно мати на увазі, що довжина кожного підмасиву відрізняється, і її необхідно отримувати з властивості students[i].length.

На жаль, списки студентів деяких груп у цьому масиві відсутні, оскільки дані для його заповнення брались із загальнодоступних джерел

Після завдання глобальних змінних, наступним кроком було написання методу Main. Цей фрагмент коду 4 зображений на рисунку 2.10.

4 Рисунок 2.10 – Метод Main

У методі Main об’єкту botClient присвоюється посилання на об’єкт TelegramBotClient, що знаходиться у однойменному класі TelegramBotClient, який наслідує інтерфейс ITelegramBotClient.

4 Рисунок 2.11 – Об’єкт TelegramBotClient

На рисунку 2.11 зображено фрагмент коду, у якому оголошується об’єкт TelegramBotClient. 4 На цьому рисунку видно, що даний об’єкт може мати два параметра. Перший параметр – це токен бота, він є обов’язковим. Другий параметр – це проксі сервер. Задання цього параметру не є обов’язковим, проте він потрібен коли бот розміщується у країні, для якої доступ у телеграм заблокований. 19 Прикладом таких країн є Росія, Китай, Пакістан та Оман. Необхідно про це пам’ятати, якщо розміщувати бота на серверах цих країн.

Після завдання токену було задано таймаут. Таймаут – це час, протягом якого після запиту GetUpdates() телеграм пришле оновлення при наявності нових повідомлення. Якщо нові повідомлення були ще до надсилання запиту GetUpdates(), то телеграм надасть відповідь моментально, після чого послідує наступний запит GetUpdates. Технічно, таймаут не є обов’язковим, проте якщо його не вказати то бот буде постійно з надзвичайно малим інтервалом надсилати запити до TelegramBotApi, це спричинить до навантаження на комп’ютер та навантаження на сервер телеграму, яке може автоматично розцінитись як DDoS-атака і IP-адреса, з якої надходять ці запити, може бути заблокована.

Задача GetMeAsync() має властивості, які дозволяють отримати самі різні дані про бота чи іншого користувача, такі як: унікальний ідентифікатор, ім'я, прізвище, інформація чи є цей користувач ботом, інформація чи може бот читати всі повідомлення в груповому чаті і чи може він взагалі приєднуватись до цих чатів, інформація про обрану користувачем мову, тощо. Вся ця інформація, звичайно ж, береться з серверів телеграму. Це чудова можливість для тестування з'єднання, що і було реалізовано.

Новооголошена змінна me – це короткий шаблон для отримання інформації від бота. Якщо підключення справне, то в консоль відбудеться вивід значень me.Id (унікальний ідентифікатор бота) та me.FirstName (ім'я бота). Результати тестування зображено у розділі 3.

Стрічка «botClient.OnMessage += Bot_OnMessage» – це "підписування" методу Bot_OnMessage на подію botClient.OnMessage. Іншими словами, при виникненні події botClient.OnMessage виконується тіло методу Bot_OnMessage.

Подія botClient.OnMessage виникає тоді, коли з'являється інформація про нове повідомлення для бота від користувача. Ця подія описана у просторі імен Telegram.Bot.Args (рис. 2.12).

Рисунок 2.12 – Опис події OnMessage

При виклику методу botClient.StartReceiving бот-клієнт розпочинає отримувати повідомлення, використовуючи LongPolling, тобто API-метод GetUpdates().

Завершує метод Main команда Console.ReadKey(). Ця команда необхідна 6 для того, щоб консоль не закривалась одразу після виводу інформації про ID та ім'я бота, а очікувала натиснення клавіші.

Основна частина програми – це метод BotOnMessage. Цей метод є підписаним на подію botClient.OnMessage, тобто він виконується кожного разу, коли бот отримує оновлення про нове повідомлення.

Метод BotOnMessage був оголошений автоматично, за допомогою підказки IntelliSense від інтегрованого середовища розробки Microsoft VisualStudioCommunity. Єдиною зміною після стандартного оголошення було додавання ключового слова async, що вказує на асинхронність методу, це дозволяє винести завдання обробки повідомлень з основного потоку програми, тобто виконувати їх одночасно з іншими частинами програми.

Фрагмент коду з оголошенням методу Bot_OnMessage зображено на рисунку 2.13

Рисунок 2.13 – Фрагмент коду з оголошенням методу Bot_OnMessage

Параметри методу Bot_OnMessage:

Objectsender – це користувач, який надіслав повідомлення.

MessageEventArgs e – це саме повідомлення.

Ці параметри також були створені автоматично за допомогою IntelliSense і в жодному разі не підлягають зміні.

Після оголошення методу Bot_OnMessage було оголошено змінну text. Змінна text – це текст повідомлення, надісланого користувачем боту. Варто розуміти, що текст і повідомлення – це не одне і те саме, оскільки повідомлення може також містити:

-Зображення

-Аудіофайли

-Голосове повідомлення

-Відеоповідомлення

-Відеофайли

-Стікери

-Геотег

-Контакт якоїсь особи

Ми відокремлюємо текст повідомлення від всіх вище перелічених речей, а якщо текст відсутній – бот не буде реагувати на повідомлення. Також, при відправленні повідомлення, воно буде приходити до нас у консоль з вказанням унікального ідентифікатора чату. Це потрібно для тестування з’єднання між ботом та сервером Telegram.

Наступним кроком було оголошення наступних функцій:

-Функція визначення дня тижня DetermineDay()

-Функція пошуку викладача FindTeacher()

-Функція пошуку групиFindGroup()

-Функція пошуку студентаFindStudent()

-Функція визначення парного/непарного тижняDetermineWeek()

Функція визначення дня тижня DetermineDay() – це проста функція, призначена для задання значення змінній для змінної d.

Функція DetermineDay() виконується не одноразово, а викликається при виконанні функції з пошуку викладача FindTeacher(), функції з пошуку студента FindStudent() та функції з пошуку групи FindGroup(), а також при зчитуванні функціональних кнопок, щоб заблокувати доступ до цих функцій у суботу та неділю. Це зумовлено тим, що день тижня постійно змінюється, тому значення для d не можна задати лише один раз. Ця функція зображена

на рисунку 2.14.

Рисунок 2.14 – Функція DetermineDay()

Відлік для змінної d розпочинається з нуля, оскільки ця змінна потрібна для індексації елементів масивів teachers та groups, перший рядок у підмасивах цих масивів за індексом 0 відповідає понеділку, за індексом 1 – вівторку і т.д.

Наступна функція – функція пошуку викладача FindTeacher(). Якщо текст введеного користувачем повідомлення збігається з прізвищем викладача teachers[0][0, i], то відбувається вивід елементу виду teachers[i][d,x]. Фрагменти цієї функції зображено на рисунках 2.15 та 2.16.

Рисунок 2.15 – Фрагмент функції FindTeacher()

Лічильник «i» вказує на викладача, змінна d, отримана після виконання функції DetermineDay() – на день. X – не змінна, цей індекс задається вручну, він вказує на поточну пару, а визначається за допомогою порівняння виду «кінець минулої пари < поточний час > кінець пари». Якщо перша пара ще не почалась або остання пара вже закінчилась, то користувачу повідомляється про це.

исунок 2.16 – Фрагмент функції FindTeacher()

Якщо текст написаного користувачем повідомлення не співпадає із жодним із елементів teacher[0][0,i], користувачу також про це повідомляється. У цій умові існує виключення text != “Знайти викладача”, воно потрібно по тій причині, що при натисканні функціональної кнопки присвоюється значення для t, і тому текст кнопки підходить під умови для початку функції.

Наступна функція – функція пошуку групи FindGroup(). Вона досить схожа на функцію пошуку викладача FindTeacher(). Якщо текст введеного користувачем повідомлення збігається з назвою групи groups[0][0, i], то відбувається вивід елементу виду groups[i][d,x]. Фрагмент цієї функції зображено на рисунку 2.17.

Лічильник «i» вказує на групу, змінна d, отримана після виконання функції DetermineDay() – на день. Індекс X – не змінна, він вказує на поточну пару, а визначається за допомогою порівняння виду «кінець минулої пари < поточний час > кінець пари».

Якщо перша пара ще не почалась або остання пара вже закінчилась, то користувачу повідомляється про це.

Рисунок 2.17 – Фрагмент функції FindTeacher()

Якщо текст написаного користувачем повідомлення не співпадає із жодним із елементів groups[0][0,i], користувачу також про це повідомляється. У цій умові існує виключення text != “Знайти групу”, воно потрібно по тій причині, що при натисканні функціональної кнопки присвоюється значення для t, і тому текст кнопки також підходить під умови для запуску функції.

Наступна функція – функція пошуку студента FindStudent(). Вона досить відрізняється від попередніх двох функцій, в першу чергу тим, що у ній використовується подвійний цикл for. Фрагмент цієї функції зображено на рисунку 2.18.

Рисунок 2.18 – Фрагмент функції FindStudent()

Звичайно, можна було б організувати ідентифікацію студента не лише за прізвищем, а і за ім’ям. Проте, користувач не завжди може знати ім’я та фамілію, а об’єм даних для заповнення в такому випадку збільшився би у два рази, так що було вирішено що користувачеві просто буде надсилатись інформація про групу кожного знайденого студента, що спростить його ідентифікацію серед інших студентів із таким же прізвищем.

Коли визначено, прізвище якого студента містить текст повідомлення, і котра в даний момент пара, проходить перевірка, чи присутня ця пара в розкладі його групи, чи ні. Якщо пара відсутня, то до користувача надходить повідомлення, що в даний момент пара у студента певної групи за певним прізвищем відсутня. Якщо пара є, то надходить повідомлення, що студент за певним прівищем певної групи мусить знаходитись у певній аудиторії. При цьому, алгоритм пошуку прізвища студента серед елементів масиву students не закінчується, і таких повідомлень може бути кілька.

Якщо прізвище студента, введене користувачем, не співпадає з жодним із елементів масиву students, користувач отримує повідомлення про це. У цій умові існує виключення text != “Знайти студента”, воно потрібно по тій причині, що при натисканні функціональної кнопки присвоюється значення для t, і тому текст кнопки також підходить під умови для розпочинання функції. Наступна функція – функція видачі

розкладу GiveSchedule(). З першого погляду вона здається дуже простою, проте насправді її реалізація досить цікава і нестандартна. Код функції GiveSchedule() зображено на рисунку 2.19.

Рисунок 2.19 – Функція видачі розкладу GiveSchedule().

Було перебрано і перевірено 9 різних хостингів картинок, і на жодному з них ніяким чином в посиланні на картинку не зберігалась оригінальна назва. Тоді було звернено увагу на репозиторій 13 GitHub.

13 GitHub - це сервіс онлайн-хостингу репозиторіїв, що володіє всіма функціями розподіленого контролю версій і функціональністю управління вихідним кодом, зазвичай цей сервіс використовують саме за цим призначенням, проте є безліч інших варіантів його використання.

Із загальнодоступних джерел було отримано зображення розкладу занять станом на другий семестр 2020 року. Їх було обрізано на окремі 24 зображення – по одному для кожної групи, і завантажено на репозиторій GitHub. Вигляд репозиторію із зображеннями розкладу зображено на рисунку 2.20.

Рисунок 2.20 – Репозиторій GitHub

Цим зображенням були видані імена по такому принципу, щоб вони збігались з індексом «i» в елементах groups[0][0,i], які вказують на назви груп. Таким чином, зображення з розкладом групи ЗІ-12 має назву 1.jpg, зображення з розкладом групи ЗІ-13 має назву 2.jpg і т.д.

На стадії проектування планувалось, що в цій функції вручну буде задано 25 тижнів – з них 12 парних і 13 непарних, інформація про їх чередування була взята з загальнодоступних джерел.

Проте, під час реалізації цієї функції, було помічено, що чередування навчальних тижнів є повністю передбаченим і без виключень вони чередуються кожні 7 днів, після цього було розроблено алгоритм функції, який зображений на рисунку 2.21.

Спочатку було задано стартову дату, яка є початком першого парного тижня, і дату, яка є початком першого непарного тижня. Також було задано фінішну дату, після якої навчання закінчується. Для зручності, ці дати було задано у зручному форматі «ДД.ММ» у змінних типу string, і після цього конвертовано у тип DateTime за допомогою функції ConvertToDateTime().

Всередині нескінченного циклу do... while(true) постійно задаються змінні par2 (кінець парного тижня) і unpar2 (кінець непарного тижня), після чого відбувається перевірка

умов: Якщо par<= поточний тиждень > par2, відбувається вивід повідомлення про те, що поточний тиждень – парний, після чого цикл завершується. Якщо unpar<= поточний тиждень > unpar2, відбувається вивід повідомлення про те, що поточний тиждень – непарний, після чого цикл завершується.

Функція DetermineWeek() зображена на рисунку 2.21.

Рисунок 2.21 – Функція DetermineWeek()

Тепер, коли функції оголошено, можна підходити до заключної частини програми – до самої реакції на повідомлення. Першим повідомленням в діалозі з будь-яким ботом завжди є команда /start. До першого введення команди /start любий чат-бот є призупиненим і не може надсилати повідомлення до користувача, команда /start – це згода на діалог з ботом. При цьому користувач завжди може знову призупинити бота за допомогою спеціальних кнопок, які є вбудованими у телеграм-клієнті, і знову запустити бота за допомогою команди /start.

У нашому випадку при введенні користувачем команди /start відбувається ініціалізація функціональної клавіатури та виведення привітального повідомлення.

У офіційній документації до TelegramBot API говориться, що у Telegramіснує два типи кнопок: Reply-кнопки та Inline-кнопки. Reply-кнопки утворюють клавіатуру, що знаходиться підполем вводу повідомлення. Натискання Reply-кнопки повністю аналогічне ручному написанню її тексту. Перевірка їх натиснення також аналогічна перевірці введеного тексту. Вигляд Reply-кнопок зображено на рисунку 2.22

Рисунок 2.22 – Reply-кнопки

Inline-кнопки є частиною повідомлення від бота, це повідомлення може бути динамічним і змінюватись в залежності від натискання кнопок. Натискання Inline-кнопок не має нічого спільного з надсиланням повідомлення, грубо кажучи, воно викликає подію. Вигляд Inline-кнопок зображено на рисунку 2.23.

Рисунок 2.23 – Inline-кнопки

У нашому випадку більш доцільним буде використання Reply-кнопок, оскільки Inline-кнопки краще всього використовувати при складній деревоподібній структурі функцій, а в цьому проекті планується, що буде 5 чітких функцій, після вибору яких текст набагато зручніше буде вводити вручну. Код ініціалізації клавіатури та виведення привітального повідомлення зображено на рисунку 2.24.

Рисунок 2.24 – Ініціалізація клавіатури та привітального повідомлення

Параметр resizeKeyboard : true вказує на те, що розміри клавіатури будуть підлаштовуватись під кількість кнопок і їх розмір їх тексту.

Оскільки «/start» – це команда початку роботи та ініціалізації клавіатури, то після її обробки в програмі розпочинається обробка натискання кнопок. Фрагмент коду з обробки натискання кнопок зображено на рисунку 2.25.

Рисунок 2.25 – Фрагмент коду з обробки натискання кнопок

Спочатку обробляються кнопки тих функцій, які не будуть заблоковані у вихідні дні. Це функції видачі розкладу занять та визначення навчального тижня. При виборі кнопки видачі розкладу занять користувачеві пропонується ввести назву групи, а при виборі кнопки визначення навчального тижня розпочинається функція DetermineWeek()

Після цього відбувається перевірка дня тижня. Якщо день тижня – субота чи неділя, і при цьому була нажата кнопка пошуку викладача, пошуку групи чи пошуку студента, то користувач отримує повідомлення про те, що обрана ним функція заблокована.

Якщо прийшло нове повідомлення, а змінна t відрізняється від нуля, це означає що користувач вже обрав функцію перед цим повідомленням, отже повідомлення мусить містити прізвище викладача, студента, або назву групи. Розпочинається обробка цих повідомлень: в залежності від значення змінної t запускається функція пошуку викладача FindTeacher(), функція пошуку студента FindStudent(), функція пошуку групи FindGroup() або функція видачі розкладу GiveSchedule(). Код обробки цих повідомлень зображений на рисунку 2.26.

Рисунок 2.26 – Фрагмент коду обробки повідомлень

3 Тестування інформаційної системи та інструкції щодо її використання

Для початку роботи з інформаційним телеграм-ботом КТКТ необхідно мати аккаунт у сервісі Telegram і знайти цього бота у пошуку або перейти за посиланням «t.me/InformationKTKTBot». Інструкції щодо пошуку чат-бота в сервісі Telegram зображені на рисунку 3.1.

Рисунок 3.1 – Інструкції щодо пошуку чат-бота в сервісі Telegram

Після знаходження чат-бота, його можна буде побачити у “сплячому” режимі. У цьому режимі бот не може надсилати до вас повідомлення, тому його необхідно “пробудити”. Для цього потрібно натиснути вбудовану кнопку «СТАРТ».

Описаний вище процес є обов’язковим для будь-якого чат-бота у Telegram. Виключення можуть складати лише Inline-боти. У випадку з Inline-ботами, ім’я бота, починаючи зі знаку «@» (наприклад, @InformationKTKTBot), згадується у будь-якому

загальнодоступному чаті з кількома користувачами, після чого бот сам приєднується до цього чату. Демонстрація дій щодо запуску бота зображена на рисунку 3.2.

Рисунок 3.2 – Демонстрація дій щодо запуску бота

Натиснення кнопки «СТАРТ» є аналогічним команді «/start», і обробляється як команда «/start», отже в нашому випадку після її натиснення розпочинається ініціалізація клавіатури.

Також користувач завжди може знову призупинити бота за допомогою спеціальних кнопок, які є вбудованими у телеграм-клієнті, і знову запустити бота за допомогою команди «/start».

Після ініціалізації клавіатури можна обирати потрібну функцію. Функції знаходження викладача, студента та групи доступні лише у будні дні. Функція визначення тижня і функція видачі розкладу доступні у будь-який день.

Після обрання функцій пошуку викладача, студента чи групи, а також після вибору функції видачі розкладу користувачеві пропонується ввести певні дані: прізвище викладача, прізвище студента чи назву групи, в залежності від обраної функції. Демонстрація загальних інструкцій щодо роботи з функціями зображена на рисунку 3.3.

Рисунок 3.3 – Демонстрація загальних інструкцій щодо роботи з функціями

Якщо дані були введені неправильно, бот про це повідомляє. Якщо після цього користувач бажає спробувати ввести дані ще раз, йому необхідно перед цим ще раз обрати необхідну функцію. У іншому випадку бот повідомить про це користувачу. Це зумовлено тим, що при неправильному введенні даних прапорець t також скидується, і функція вважається виконаною.

Якщо функцію не обрано, а користувач ввів повідомлення, бот повідомляє йому, що необхідно обрати функцію.

У функції знаходження викладача необхідно ввести лише прізвище викладача в називному відмінку з великої літери, без імені та ініціалів.

У рамках функцій з пошуку групи та видачі розкладу користувачеві пропонується ввести назву групи. Демонстрація роботи функцій з видачі розкладу та пошуку групи зображена на рисунку 3.4.

Рисунок 3.4 – Демонстрація роботи функцій з видачі розкладу та пошуку групи

Для повідомлення з назвою групи від користувача є лише одне правило, і помилитись

важко: назва групи має бути введена через дефіс. Регістр у цьому повідомленні значення не має, оскільки при перевірці воно все одно підноситься до верхнього регістру.

Функції знаходження викладача, студента та групи блокуються у вихідні дні. На рисунку 3.5 продемонстровано блокування цих функцій та робота функцій видачі розкладу і визначення навчального тижня у вихідний день.

Рисунок 3.5 – Демонстрація роботи функцій чат-бота у вихідні дні

Функція пошуку студента відноситься до функцій, що працюють лише у будні дні. У цій функції необхідно ввести лише прізвище викладача в називному відмінку з великої літери, без імені та ініціалів. У випадку, якщо в коледжі існує кілька студентів з таким прізвищем, чат бот почергово виведе місцезнаходження всіх студентів з цим прізвищем, із вказанням групи, у якій навчається кожен з них.

Демонстрація роботи функції пошуку студента зображена на рисунку 3.6.

Рисунок 3.6 – Демонстрація роботи функції пошуку студента

Посилання

Це джерела виділених збігів у вашому документі. Кожен збіг позначено темно-зеленим числом, яке відповідає вказаному тут джерелу. Джерела впорядковані за схожістю — чим вищий бал, тим сильніше збіг.

#	Джерело	%
1	science.kname.edu.ua	2.0%
2	victoria.lviv.ua	1.7%
3	shotam.info	1.0%
4	ena.lpnu.ua	1.0%
5	dut.edu.ua	0.6%
6	essuir.sumdu.edu.ua	0.6%
7	eprints.library.odeku.edu.ua	0.5%
8	sap.pstu.edu	0.4%
9	sci.ldubgd.edu.ua	0.4%
10	elar.tsatu.edu.ua	0.4%
11	dut.edu.ua	0.4%
12	ir.nmu.org.ua	0.3%
13	seraf.ru	0.3%
14	ztec.com.ua	0.3%
15	lib.pu.if.ua	0.3%
16	repository.hneu.edu.ua	0.3%
17	essuir.sumdu.edu.ua	0.2%
18	ela.kpi.ua	0.1%
19	ir.kneu.edu.ua	0.1%
20	ela.kpi.ua	0.1%

Plag

Дякуємо, що перевірили
свій документ за допомогою
Plag!