МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Київський національний університет імені Тараса Шевченка
Навчально-науковий інститут філології
Катедра української мови та прикладної лінгвістики

Автоматичне укладання граматичних вправ для навчання української мови як іноземної

Кваліфікаційна робота
освітнього ступеня «бакалавр»
за спеціальністю 035 «Філологія»,
спеціалізацією 035.10 «Прикладна лінгвістика»,
галузі знань 03 «гуманітарні науки»
ОПП «Прикладна (комп'ютерна) лінгвістика та англійська мова»
студентки IV курсу
Вероніки БАБИЧ

Науковий керівник:
Микола КОСТІКОВ

«Допущено до захисту»
Протокол № 11 засідання кафедри
української мови та прикладної лінгвістики
ННІФ від 01.06.2023
Завідувач кафедри_____**Сергій Різник**

КИЇВ – 2023

ЗМІСТ

ВСТУП

Українська мова набуває все більшої популярності та вживаності на міжнародній арені. Спостерігається підвищення престижу України у світі, і відповідно – підвищення інтересу до вивчення її державної мови серед іноземних громадян. «Приєднання України 2005 року до Болонського процесу збільшило кількість програм обміну між українськими та закордонними університетами, що своєю чергою позначилося на числі іноземних студентів, охочих опанувати українську. Водночас за кордоном збільшується кількість академічних осередків (Кембриджський університет, Берлінський університет імені Гумбольдта, Український інститут у Лондоні та ін.), які дають змогу іноземцям вивчати українську мову» [35; 2].

Також із початком повномасштабного вторгнення російської федерації на територію України 24 лютого 2022 року значно підвищився інтерес пересічних громадян інших країн до вивчення української мови. Насамперед про це каже річний звіт компанії Duolingo, яка займається розробкою платформи для вивчення іноземних мов. Попит сягнув піку наприкінці березня та лишався стабільним до кінця року; доповідається про приріст як у сусідніх країнах, які найбільш активно приймали біженців, так і в далеких від конфлікту. [40]

Сьогодні у ЗВО України створено велику кількість підготовчих відділень для іноземних громадян, а також упроваджені спеціальні освітні програми для іноземців, на яких вивчається українська мова. Відповідно, розробляється велика кількість навчальних матеріалів для забезпечення освітнього процесу цих програм: як традиційні підручники, посібники, збірники вправ, розмовники та інші друковані засоби, так і різноманітні інноваційні інформаційні проєкти – електронні підручники, інформаційні сайти, застосунки, ігри. Але процес вивчення української мови іноземцями має певні проблеми, одна з яких – невідповідність більшості навчальних матеріалів європейським та українським стандартам, зокрема Стандартизованим вимогам до рівнів володіння українською мовою як іноземною A1-C2 [38].

Важливу роль у процесі навчання іноземній мові мають індивідуальні практичні завдання, що моделюють реальні мовленнєві ситуації, відповідають «єдиним державним вимогам до рівнів володіння українською мовою як іноземною (Стандартизовані вимоги), які зможуть стати основою для організації й проведення сертифікаційного іспиту (Діагностика) і підставою для запровадження документа єдиного зразка (Сертифікат), що засвідчуватиме рівень володіння українською мовою» [35; 2].

Традиційні підручники містять обмежену кількість таких вправ, тому що укладання великої кількості вправ, зокрема тестових завдань, може бути складним та трудомістким з огляду на велику кількість факторів. Актуальності та популярності у сучасній лінгводидактиці, зокрема і в галузі навчання української мови як іноземної, набувають електронні підручники, оснащені інтерактивними тестовими завданнями.

Методика створення повномасштабної автоматизованої навчальної системи з української мови вперше була розроблена у 2001 р. колективом лабораторії комп'ютерної лінгвістики Київського національного університету імені Тараса Шевченка [39]. Ця система розрахована на учнів середніх шкіл, абітурієнтів, студентів, які вивчають українську мову або хочуть систематизувати свої знання з української мови. За статистикою входження користувачів на сторінку підручника на сайті www.mova.info, цей електронний підручник користується великою популярністю як серед українських, так і зарубіжних користувачів [1].

У тенденції розвитку сучасних комп'ютерних технологій у галузі освіти перед освітянами та науковцями постає нове дидактичне завдання – створення інтерактивних електронних систем для автоматичного самостійного створення викладачем та вчителем електронних завдань для учнів та студентів. Такі методичні застосунки є вимогою нашого часу, тому що забезпечують ефективність роботи викладача, ефективність організації освітнього процесу та мають певні психологічні переваги – динамічність, цікавість, особливо для молоді, легкодоступність навчання через мережу Інтернет.

Створення Інтернет-застосунку автоматичного укладання тестових вправ з української мови як іноземної – актуальне завдання сучасної комп'ютерної лінгвістики у галузі лінгводидактики.

Мета роботи: створити електронну систему автоматичного укладання граматичних вправ з української мови. Мета передбачає виконання таких ***завдань дослідження:***

1) визначити проблеми та завдання процесу навчання української мови як іноземної;
2) охарактеризувати рівні володіння українською мовою за Стандартизованими вимогами до рівнів володіння українською мовою як іноземною A1-C2 (далі – Стандарт);
3) дослідити значущість вправ у процесі вивчення іноземних мов загалом та української мови зокрема (на прикладі сучасних навчальних матеріалів);
4) дослідити системи автоматичного укладання вправ з іноземної мови, окреслити реєстр засобів оброблення природної мови (Natural Language Processing – NLP), які використовуються у таких застосунках;
5) побудувати алгоритм створення граматичних вправ з української мови;
6) розробити baseline solution для програмного забезпечення автоматичного укладання граматичних вправ з української мови на базі заданих текстів (тема – займенники, рівень володіння мовою – A1).
7) розробити програмне забезпечення автоматичного укладання граматичних вправ з української мови на базі заданих текстів (теми – займенники й іменники), яке:
   a. виконує вимоги принципів об’єктно-орієнтованого програмуваання (далі – ООП);
   b. використовує технології баз даних (далі – БД);
   c. і таким чином має кращі можливості для масштабування.

Об’єктом дослідження є речення із займенниками й іменниками української мови.

Предметом дослідження є комп'ютерні моделі граматичних тестових завдань з опорним реченням із займенниками й іменниками української мови.

Методи дослідження: метод комп’ютерного моделювання; метод автоматичного аналізу та синтезу; методика морфологічного аналізу.

Практичне значення роботи полягає в можливості використання розробленого програмного продукту у створенні навчальних вправ з української мови для забезпечення процесу навчання української мови як іноземної великим обсягом інтерактивних індивідуальних вправ.

Теоретичне значення роботи: отримані результати дослідження є новаторськими для методики навчання української мови як іноземної у галузі сучасної лінгводидактики, а також для української комп’ютерної лінгвістики у галузі автоматичного укладання українськомовних текстів.

Інформаційна база дослідження: мова програмування Python [55] та її бібліотеки: ast [41], collections [42], customtkinter [4], nltk [50], numpy [51], pandas [53], pymorphy2 [24], pymorphy3 [54], random [56], re [57], sqlite3 [62], stanza [63], tkinter [64], tokenize_uk [58].

Матеріал дослідження: два корпуси речень обсягом 586 і 847 речень, що відповідають рівню володіння українською мовою А1.

Апробація дослідження: доповідь «Автоматичне генерування тестових завдань із граматики української мови як іноземної» на Всеукраїнській науковій конференції «Мова, література, переклад у комунікативному просторі сучасного світу» з нагоди 20-річчя Навчально-наукового інституту філології Київського національного університету імені Тараса Шевченка [4]; доповідь «Проєктування навчальної системи для вивчення українських займенників» на IX Міжнародній науково-технічній Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» [3].

РОЗДІЛ 1
ВИВЧЕННЯ УКРАЇНСЬКОЇ МОВИ ЯК ІНОЗЕМНОЇ

1.1. Особливості навчання української мови як іноземної

Можна, напевне, сказати, що кожна людина, маючи базові знання з лінгвістики або хоча б знаючи якусь іноземну мову, зможе виокремити щонайменше дві причини, чому навчання української мови (та й будь-якої іноземної мови загалом) може мати свої особливості.

Вірогідно, що труднощі пояснюються граматикою української мови, і насамперед її флективністю. Серед перших 10 країн, громадяни яких обирають навчання в Україні, більшість є такими, чиї мови виявляють дуже мало або взагалі не мають ознак флективності – азербайджанська (6,8% від загальної кількості студентів-іноземців), туркменська (відповідно 6,64%), турецька (4,68%), китайська (4,38%), узбецька (2,52%), ігбо та йоруба в Нігерії (5,44%), іврит у Ізраїлі (3,18%) [35]. Студентам із цих країн, що не знають якоїсь мови-посередника (особливо російської, оскільки інша найпоширеніша – англійська – також не є флективною), звісно, буде важко зрозуміти усі морфологічні парадигми української мови, знайти та виокремити в них якісь закономірності для запам’ятовування. В інших іноземців можуть бути проблеми у навчанні української мови з причини нерозуміння граматичних категорій, наприклад, індійцями, які складають 22,9% серед усіх іноземних студентів України та спілкуються переважно мовою гінді (57,1% населення Індії), мовою бенгалі (8,9%) та маратхі (8,2%), серед яких лише остання має категорію роду іменника (на щастя, вона має ті самі три значення, що й в українській).

Не менш важливою проблемою навчання української мови як іноземної є відмінність соціального та культурного середовища, адже мова формується під впливом культури та соціуму (Едуард Сепір [30], Бенджамін Уорф [32], Франц Боас [19]). Цей процес впливу можна помітити, якщо вивчати іноземну мову

«без посередника», наприклад, дивлячись серіали чи фільми, які розповідають про звичайне повсякденне життя. Автори навчальних матеріалів з іноземних мов не завжди вважають за потрібне брати до уваги певні культурологічні аспекти вивчення мови. Натомість, такі концепти можуть містити навіть базові побутові питання, які нам здаються докорінно зрозумілими, проте в людей, що звикли до кардинально іншого середовища, може статися «культурний шок» та з'явитися певний мовний бар'єр [14], [15]. Тому вважаємо, що надзвичайно важливим є вивчення комбінованих тем на кшталт «Похід до магазину», «Поїздка в тролейбусі», «Прогулянка в парку» тощо, які відображатимуть розвиток лексичного запасу, навичок говоріння та сприйняття на слух українського мовлення. Також існують дослідження щодо розроблення за цими темами певних рольових ігор, що мають на меті саме адаптацію, тренування невимушеного спілкування, підготовку до вирішення можливих проблемних ситуацій [13], [16].

Важливим аспектом вивчення української мови як іноземної є обізнаність із загальними фоновими знаннями про Україну. Каті Бруннер, викладачка Берлінського університету імені Гумбольдта, у 2008 р. зазначала, що у Німеччині превалює стереотип про те, що всі українці знають, розуміють та спілкуються російською мовою (і таким чином для життя в Україні цілком достатньо хоч якогось знання саме російської мови), або про те, що українська мова є всього-на-всього діалектом російської мови, або української мови взагалі не існує. Сьогодні ситуація в Німеччині кардинально змінилася: Інтернет-сторінка українського посольства в Німеччині подає інформацію про державну мову в Україні, а в розділі про відношення між країнами є стаття про рік української мови в Німеччині та покликання на сайти Українського культурного фонду, Українського інституту та Інтернет-платформу «Україна в Німеччині» із застосуванням бренду України UkraineNOW. У зазначеній статті 2008 року зазначаються також проблеми репрезентації української мови в німецьких ЗВО та обмеженість вибору посібників та підручників: на час написання статті авторка знайшла лише 6 позицій, із яких дві – самовчителі для

швидкого вивчення базової лексики з метою туристичного перебування в країні [8].

Загалом, за аналізом літератури ([17], [7]) із цієї проблематики, можна виокремити такі проблеми в галузі вивчення української мови як іноземної:

1) адаптація до нових умов життя та різкої зміни середовища, «культурний шок», культурологічні та ментальні відмінності, різні норми поведінки;

2) необхідність диференціації іноземців, які бажають вивчати українську мову, за категоріями залежно від рівня знань мови-посередника (насамперед російської та англійської) та напрямів основного навчання (для студентів);

3) відсутність окремих підручників, посібників та інших навчальних матеріалів, що відповідали б сучасним вимогам;

4) відсутність мотивації до вивчення мови, зокрема пов'язаної зі стигмою «в Україні всі знають російську» або переконання в тому, що, якщо далі навчання відбуватиметься англійською мовою, то українська не знадобиться;

5) для студентів – розроблення для кожної категорії програм курсу практичної української мови та адаптування чинних програм (зокрема упорядкування кількості навчального часу для кожної дисципліни та уніфікація форми підсумкового контролю).

З метою вирішення окреслених проблем у травні 2018 р. рішенням колегії Міністерства освіти і науки України було ухвалено Стандартизовані вимоги до рівнів володіння українською мовою як іноземною (далі Стандарт) [38]. Цей документ регламентує розроблення та уніфікацію навчальних програм дисциплін та сучасних навчальних матеріалів до них. Стандарт уніфіковує систему вимог до володіння українською мовою, чітко поділяючи їх на всі види комунікативної діяльності (слухання, читання, письмо та говоріння), які докладно описано відповідно до загальноєвропейських рекомендацій із мовної освіти.

У межах Стандарту надзвичайно важливими є каталоги – переліки компетенцій, якими мають володіти мовці, що претендують на визнання певного рівня мови. У Каталозі А викладається перелік комунікативних

намірів; Каталог Б є тематичним; Каталог В визначає зміст мовної компетенції. Каталог В є базовим для нашого дослідження, тому що в ньому подано вимоги до вмінь студентів у межах морфології та синтаксису.

1.2. Місце практичних завдань (вправ) у процесі навчання іноземної мови

Вправи грають важливу роль у вивченні будь-якої дисципліни, а особливо у вивченні мови: вони повторюють або симулюють реальні ситуації, у яких може бути застосований певний рівень умінь у відповідній сфері. Таким чином, здобувачі освіти отримують досвід використання своїх знань та вмінь, учителі (викладачі) – інструмент для перевірки цих знань та вмінь, а екзаменатори – чіткий показник рівня підготовки.

Говорячи безпосередньо про вправи в процесі навчання іноземної мови, необхідно обґрунтувати поняття інтермови (не плутати з мовою-посередником). Термін «інтермова» був уведений Ларі Селінкером [31]. Цей термін означає мову як результат навчання іноземній мові, в якому комбінуються такі складники: перенесення мови (некоректне використання характеристик першої мови (М1) у другій мові (М2)); надузагальнення (використання правил М2 у недоречних ситуаціях); передача тренування (відбиток стилю викладача, створення «правила» там, де насправді його немає); стратегія вивчення (вибір учнем якнайшвидших шляхів досягнення результатів); стратегія комунікації (фокусування на меті передати інформацію, нехтуючи правильністю). Саме інтермову перевіряють учителі та екзаменатори, коли намагаються дати оцінку рівню знань учня. Перші намагаються з’ясувати причини, наслідки та способи викорінення такого недоліку мовлення, другі – вивести ступінь її близькості до літературної, ідеальної мови. Вправи в такому разі грають роль штучного середовища для вживання інтермови та відтворення поточного рівня мовлення.

Різновиди вправ у процесі вивчення іноземної мови визначаються двома підходами до вивчення мови – граматичним та комунікаційним. Рівномірне

поєднання обох аспектів може пристосувати майбутнього мовця до усіх можливих комунікаційних ситуацій. Це також підтримується Стандартом, який регламентує комунікаційні та граматичні вимоги до кожного з рівнів володіння українською мовою.

Цікавою в цьому аспекті є праця Вівіан Кук [20], яка описує багато проблем навчання другої мови, серед яких стилі викладання. Так, є академічний стиль із фокусом на поясненні граматики та перекладі; аудіолінгвальний, що починається з усного мовлення – сприйняття на слух та власне говоріння, і вже через читання доходить до письма; комунікативний – із метою привчання учнів до інтеракції бажаною мовою; стиль завдань (“task-based”), який базується на виконанні учнями певних «дослідницьких» проєктів із метою самостійного заповнення мовного «пробілу» (“gap”); мейнстримний стиль EFL (Englishas a Foreign Language) – такий, що суміщує багато різних практик, але в основному поділяється на два етапи – презентацію вчителя та практику.

Існують й інші методи, проте авторка робить висновок, що лише останній, мейнстримний, «змішаний», може застосовуватися майже для всіх цілей. Якщо виділяти у ньому саме граматичні питання, оскільки метою нашого дослідження є укладання граматичних вправ, то можна вивести такі найбільш розповсюджені види вправ з граматики: fill-in-the-gap – вправа, у якій учень має заповнити «пропущене» слово; її різновид – вправа з альтернативними варіантами відповіді: учень не має самостійно згадувати відповідь, а лише обрати відповідь із пропонованих варіантів; sorting – вправа, у якій подано два списки, між елементами яких треба встановити відповідність; вправа з відкритою відповіддю, в якій учень має самостійно відповісти на питання, не маючи підказок.

Аналіз сучасних підручників із вивчення української мови як іноземної відомих українських педагогів таких, як В. Вінницька [9], Д. Мазурик [12], О. Антонів, Л. Паучок [2], демонструє ці три види вправ (fill-in-the-gap, sorting та з відкритою відповіддю).

На основі проаналізованих підручників робимо висновок, що граматичні вправи доволі активно використовуються, найчастіше – різні підвиди питань із вільною відповіддю (fill-in-the-gap, sorting question) і безпосередньо з вільною відповіддю.

У спілкуванні із представником онлайн-проєкту EasyMova, спрямованого на вивчення української мови іноземцями, ми з'ясували, що навчання за їхньою системою проходить за підручником Данути Мазурик, а також використовуються ще чотири допоміжні підручники. У цілому вчителі задоволені кількістю різноманітних матеріалів у підручниках, зокрема і граматичних вправ, проте інколи й самі складають додаткові, в основному з метою урізноманітнити матеріал. Новоукладені вправи можуть бути класифікаційного характеру, на виправлення особових закінчень прикметників та іменників, на утворення правильної форми займенника чи форми дієслова тощо.

Повернемося до вправ із декількома варіантами відповіді, які є найбільш цікавими для нас з огляду на відносну простоту їхнього укладання. Безсумнівно, вони найчастіше використовуються для самоперевірки, для проміжного контролю, оцінки рівня знань учнів та, згодом, для сертифікаційних іспитів. Такі вправи формуються на спеціальних бланками для внесення відповідей, тому що їх потім зручно перевіряти за допомогою комп’ютера; саме таким чином відбувається українське зовнішнє незалежне оцінювання та інші іспити такого типу в багатьох країнах. Стандарт також описує організацію сертифікаційного іспиту з української мови. Письмовий етап цього іспиту спрямований на оцінювання навичок читання, слухання та письма.

РОЗДІЛ 2
АВТОМАТИЧНЕ УКЛАДАННЯ ГРАМАТИЧНИХ ВПРАВ

2.1. Огляд систем автоматичного укладання вправ

Проблема автоматичного укладання вправ є актуальною для всіх галузей знань, адже застосування таких форм навчання дозволяє (за умови достатнього обсягу вихідних даних) уникати повторів текстового матеріалу у завданнях, додає елемент новизни до процесу навчання та тим самим робить його більш ефективним і цікавим як для учня, так і для вчителя. Майк Леві [26], який визначає завдання укладання вправ завданням прикладної лінгвістики, зазначає, що у процесі «ручного» створення вправ існує ризик помилок, пов'язаних із людським фактором: можна помилитися в основному тексті вправи або у відповіді, зробивши завдання неможливим для виконання; також можуть виникнути такі проблеми, як складність проведення межі між рівнями завдань або помилкове ухилення від рівневих норм, випадкове утворення неавтентичних висловів та ін. Таким чином, якщо автоматизувати процес укладання вправ за допомогою точно налаштованих параметрів на основі ретельно перевірених даних, то таких проблем не виникатиме.

Класифікація сучасних систем автоматичного укладання вправ у навчанні мови як іноземної, розроблена у роботах О. Феногенової та Є. Кузьменко [25], враховує такі диференційні ознаки: 1) мову вивчення; 2) тип вправи; 3) джерело укладання; 4) цільовий аспект вивчення (лексичний / граматичний). Щодо джерела укладання, то найбільш зручним і поширеним базисом визнається корпусний, який використовується більшістю сучасних систем. До корпусів часто приєднуються онтології та тезауруси; можливим є використання різноманітних словників, списків та інших видів представлення інформації, зібраної з мережі Інтернет. Порівняно новим підходом у генеруванні вправ за корпусними даними є використання методів машинного навчання (Machine

Learning, ML), що контролюється кількома параметрами, серед яких статистичні патерни, створені на основі речень.

На нашу думку, метод роботи з веб-даними може розумітися ширше – як webscraping, що передбачає використання засобів та різноманітних ресурсів автоматично обробленої текстової інформації: різноманітні програмні бібліотеки, збірки даних чи готові засоби для самостійного тренування подібних ресурсів. За опрацьованою у дослідженні літературою, можна коротко охарактеризувати сучасні системи автоматичного укладання вправ для вивчення іноземної мови.

Система Vicomtech: іспанська, баскська, англійська, французька мови; корпусний тип; лексико-граматичні вправи; користувач обирає для аналізу свої власні тексти, також можна обрати частину мови та певні морфологічні характеристики; серед NLP функцій – тегування частин мови, тлумачення лексичного значення та вкладання слів (таким чином, використовується машинне навчання); три види вправ: із вільною відповіддю (можливі підказки), з альтернативними відповідями (за допомогою word2vec), складання речення з перемішаних слів; створені вправи можуть бути експортовані для застосування в інших навчальних системах; у застосунку, що створює вправи, можна також і відповідати та отримувати свій результат [29].

Система Егейського університету: англійська мова; онтологічний тип Semantic Web technologies; лексичний тип; питання з декількома варіантами відповіді (але основою є питання «Оберіть правдиве твердження»); варіанти відповідей генеруються за допомогою SimpleNLG – інтерфейсу Natural Language Generation [28].

Системи Ниського університету: англійська мова; онтологічний тип, Semantic Web technologies; лексичний тип; автори, посилаючись на дослідження, описане в попередньому пункті, говорять про складнощі утворення синтаксично коректних речень методами NLP, пропонують використання «шаблонів питань» («Яке із запропонованих визначень відповідає концепту x?», «Прочитайте абзац x нижче та вирішіть, який із концептів він

описує?» тощо); використовується для навчання медиків професійної англійської мови [21].

Система Вищої школи економіки: англійська мова; на основі корпусів (British National Corpus, British Academic Written EnglishCorpus); лексичний тип; 5 видів питань: знаходження відповідників, альтернативні відповіді, відкрита відповідь (без варіантів та зі словами для довідок, на утворення правильної деривативної форми); із використанням word2vec та даних Wiktionary (пошук деривативів) [25].

Застосунок для AndroidWordGap, Бременський університет: англійська; корпусний тип; лексичні вправи (вибір синонімів); питання з декількома варіантами відповідей; користувач завантажує тексти із власного файлу або з сайту; доступне налаштування частини мови (іменник, дієслово, прикметник або прийменник); доступне оцінювання; збереження незнайомих слів; використання WordNet, The Natural Language ToolKit (NLTK), Nodebox Linguistics [23]. Схема процесу автоматичного оброблення природної мови системи WordGap подана на Мал. 2.1.

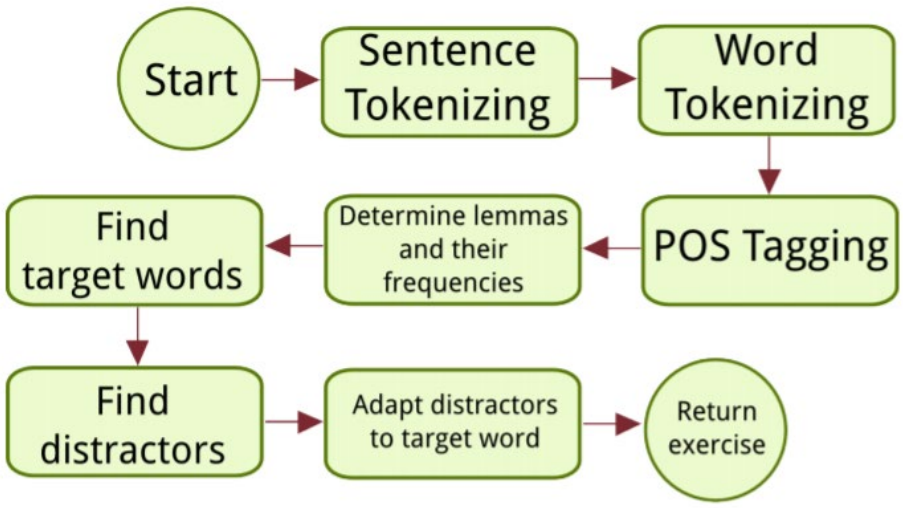


Figure 2: The NLP processing pipeline of the WordGap server

Малюнок 2.1. Схема NLP на сервері WordGap [24; 43]

Система Дж. Хілл та Р. Сімга (університет Джорджа Вашингтона): англійська мова; корпусний тип; питання з декількома варіантами відповіді;

призначена для тренування навичок читання; використання векторних моделей (GloVe) та Google Books n-grams [22].

ELLE – цифрова бібліотека вправ для вивчення мови: не генерує вправи самостійно, тому не обмежується мовою та не має типу за методом укладання; граматично-лексичний тип; 7 типів вправ: з альтернативними відповідями, на знаходження відповідності, класифікаційні (розподіл за категоріями чи між собою в певному порядку), fill-in-the-blank (записати у порожнє поле), власне з вільною відповіддю та на правильне написання; є можливість обирати рівень складності завдання; система дозволяє вчителям вносити власні вправи [33].

ArikIturri: баскська та англійська мови; наукова галузь; перспектива використання системи для багатьох мов; корпусний тип; лексико-граматичний тип; питання з декількома варіантами відповіді, fill-in-the-blank (записати у порожнє поле), утворення словоформ на виправлення помилок; використання онтологій (зокрема WordNet), словників, різноманітних методів NLP, 5-грам [18].

AEGIS: англійська мова; корпусний тип (вже розмічені дані); лексико-граматичний тип; питання з альтернативними відповідями, fill-the-gap (заповнити пробіли) та на виправлення помилок; розподіл за рівнем складності (визначення рівня та зміна рівня складності завдання, відповідно до рівня учня, враховуючи кількість правильних відповідей для різних питань), координування виконання із вчителем [27].

За короткими довідками систем автоматичного укладання вправ, ставимо завдання дати відповідь на такі питання: по-перше, чи можна класифікувати системи за джерелами їх створення; по-друге, які методи автоматичного укладання доречно використовувати для укладання граматичних вправ.

Щодо першого питання, на нашу думку, класифікацію за джерелами укладання можна вважати коректною за умови, що корпуси текстів мають глибоку лінгвістичну розмітку, яка виконує функцію даних для укладання вправ за тестовим принципом – слова-кандидати для вправ з альтернативними відповідями. Якщо ж система має працювати із «сирим» нерозміченим текстом,

то мають бути використані певні лінгвістичні автоматичні аналізатори, що дозволяють анотувати текст різноманітними методами: за допомогою програмних бібліотек, що мають широкий обсяг функцій, серед яких в основному застосовується токенізація речень та слів, а також тегування частин мови; за допомогою онтологій та семантичних системи; за допомогою методів машинного навчання, серед яких насамперед вкладання слів (векторне представлення слів) та n-грами – ймовірнісні контекстні схеми. У табл. 2.1. подано порівняння частоти використання методів автоматичного укладання вправ.

На жаль, доступна література з цього питання не подає описів автоматичних систем укладання власне граматичних вправ, таких, що не мають лексичного функціоналу. Якщо аналізувати системи, що мають і граматичний складник укладання (таких дві – Vicomtech та ArikIturri, не враховуючи власне корпусні), то можна зазначити, що вони використовують усі перераховані методи; проте припускаємо, що використання онтологій потрібне лише для лексичного типу вправ. Методи ML, на нашу думку, можна застосовувати і для граматики; як мінімум, NLP-бібліотеки будуються вже на їх основі, тому применшувати їх роль у цьому процесі не варто.

Таблиця 2.1. Порівняння методів програмування автоматичного укладання вправ (жовтим позначені системи утворення лексичних вправ, зеленим – лексичних та граматичних).

	1	2	3	4	5	6	7	8	9
власне корпусний							+		+
NLP-функції	+	+		+	+			+	
використання онтологій	+	+	+		+			+	
використання ML	+			+		+		+	

2.2. Розроблення baseline програмного забезпечення автоматичного укладання тестових завдань

Проаналізовані у першому параграфі цього розділу системи автоматичного укладання тестових вправ та методи автоматичного аналізу дозволяють зробити висновки щодо вибору методів та можливостей їх застосування для досягнення поставленої мети дослідження. Було вирішено розробити baseline solution для такої програми – початкову версію, яка має на меті показати загальний задум розробки. Такий підхід також дозволив виявити проблеми й можливі слабкі місця. Насамперед таким чином треба було дослідити програмні бібліотеки, які мали допомогти виконати окремі завдання роботи, дізнатися їхні переваги й недоліки й обрати серед них ті, які дають найкращі результати. Це також дозволило абстрагуватися від такого аспекту розробки програмного забезпечення, як його інтерфейс – оскільки це не фінальний результат, то він би не використовувався би повноцінно цільовим споживачем.

Проєктування програмного забезпечення системи автоматичного укладання тестових вправ було розпочато зі створення алгоритму укладання (повний опис алгоритму подано у Додатку В).

За алгоритмом було розроблено програмне забезпечення. Повну директорію проєкту можна переглянути за посиланням у Додатку Д. У кореневій папці 6 файлів: config.py, task_generator.py, pronouns.csv, test1.txt, test2.txt, test3.txt. Перші два файли Python: програмне забезпечення системи автоматичного укладання вправ написане мовою програмування Python [55] і повинно підтримуватися усіма версіями 3.x. Перший файл config.py має 3 основні функції, які забезпечують ієрархію та побудову програми; другий – виконавчий, що забезпечує інтеракцію з користувачем. Файл pronouns.csv містить автоматично укладену базу даних (БД) 457 граматичних вправ. Фрагмент БД подано у Додатку А. База даних формується у формі текстового

файлу для представлення табличних даних із розширенням CSV (Comma-SeparatedValues).

Три текстові файли призначені для тестування роботи програми. Файл test3.txt містить 586 речень (див. фрагмент у Додатку Б), за якими здійснюється автоматичне укладання тестових вправ. Ці речення були написані нами разом із колегами Юлією Дашинич та Еліною Півкіною за темами, визначеними Стандартом для рівня володіння українською мовою A1 [35, 7]. Два інші текстові файли test1.txt та test2.txt слугували для перевірки повторень у результатах роботи програми.

Спочатку планувалося створення алгоритму укладання трьох варіантів вправ – з альтернативними відповідями, fill-in-the-gap та на виправлення помилок. Але був здійснений для першої версії проєкту тільки перший варіант; другий є дещо складнішим (про це нижче за текстом); а третій, на нашу думку, як і варіант зі встановленням правильного порядку слів у реченні, є заскладним для рівня володіння мовою A1. Це питання вимагає подальшого дослідження із викладачами-методистами.

Система автоматичного укладання граматичних вправ за темою «Займенник» будувалася на базі трьох функцій:

1. Оброблення текстових даних: токенізація речень та слів, тегування частин мови, знаходження цільових слів, визначення кандидатів-дистракторів (інших словоформ лексеми) для формування вправ із декількома варіантами відповіді.
2. Вибір користувачем джерела вправи.
3. Укладання вправ разом із функціями вибору бажаної кількості тестових завдань та типу оцінювання.

Розглянемо порядок реалізації усіх основних функцій програми. Реаліазацію першого пункту алгоритму – оброблення текстових даних – забезпечує функція textworker. У реалізації цієї функції виникла проблема з токенізацією речень. Було апробовано дві програмні бібліотеки: NLTK [50] та tokenize_uk [58], жодна з яких не дала ідеального результату. При застосуванні

NLTK було отримано помилковий результат сегментації восьми речень у тест-кейсі з 586 речень (див. Додаток Г, п. 1). У всіх випадках речення закінчувалося на одноградемне слово з крапкою (програма визначає їх як скорочення або ініціали). Інший модуль - tokenize_uk - помиляється в сегментації 98 речень (Додаток Г, п. 2), але більшість помилок можна звести до проблеми розділових знаків: знак питання наприкінці речень та знак тире, з якого починаються репліки в реченнях діалогічного мовлення. Також повторюється проблема зі словами з однієї-двох графем, проте лише з латинськими літерами. Зважаючи на недоліки та переваги двох бібліотек, для створюваної системи було обрано модуль NLTK.

Бібліотека pymorphy2 [24] була використана для тегування частин мови (знаходження серед них займенників) та утворення відмінкових форм – дистракторів – для вправ з альтернативними відповідями. Цей модуль насамперед розроблявся для російської мови, тому для автоматичного морфологічного аналізу української мови його функції є дещо обмеженими.

По-перше, відсутній параметр score (оцінка ймовірності правильності аналізу). Значення цього параметру для всіх слів української мови дорівнює 1, і тому усі варіанти сортуються не за ним, а за частинами мови та відмінками – у такому порядку, як вони зазначені у списку грамем OpenCorpora [52] (див. Додаток Г, п. 3).

По-друге, якщо для російської мови існує позначення для другого варіанта відмінків, то для української мови воно не використовується (зокрема для займенників); місцевий відмінок від *ваш* має два варіанти: *вашім і вашому*, і при запиті на відмінювання слова *ваш* програма знаходить перший за алфавітом. Це, на нашу думку, може викликати труднощі при виборі правильного варіанта відповіді у людей, які знають українську мову лише на рівні А1. Також хочемо зауважити, що в описі тегів частин мови тег «NPRO», що використовується програмою для усіх без винятку займенників, позначений як «займенник-іменник»: «іменниками» в такому разі вважаються, наприклад, *скільки, який, багато, один* тощо; крім того єдиний тип займенників, що

позначається окремою грамемою – особові, що не відображено в документації бібліотеки.

Для визначення у тексті займенників рівня А1 була створена змінна з початковими формами усіх цих займенників. Насправді, класифікація займенників у Стандарті не зовсім точна – нема повного списку, крім того особові займенники *він, вона, воно, вони* вивчаються не раніше рівня В1 (Рубіжного), але Стандарт подає приклади речень із цими займенниками на нижчих рівнях володіння українською мовою. Також виникають проблеми із займенниками *його* та *її*, які можуть виступати як незмінні присвійні або як форми особових *він* та *вона* (див. Додаток Г, п. 3). Було вирішено не вводити до списку початкових форм ці два займенники, адже в будь-якому разі вони визначатимуться як словоформи від *він* та *вона*, проте в перспективі потрібно буде вирішувати проблему з незмінними присвійними займенниками. Створює проблему і граматична омонімія займенникових форм і сполучників *та, тому*. Тому БД містить сім таких помилок, і у випадку введення користувачем його власних текстів програма може зробити таку помилку.

У результаті було сформовано список початкових форм: *я, ми, ти, ви, він, вона, воно, вони, мій, моя, моє, мої, твій, твоя, твоє, твої, наш, наша, наше, наші, ваш, ваша, ваше, ваші, їхній, їхня, їхнє, їхні, той, та, те, ті, цей, ця, це, ці*. Програма оминає словоформи орудного відмінка в реченнях, оскільки його вживання не вивчається на цільовому рівні, і залежно від відмінкової форми в реченні формує список варіантів інших відмінків слова, які не повторюються.

Варто зазначити, що таке жорстке визначення цільових слів у програмному коді є виявом hardcoding і не є хорошою практикою при програмуванні. Також у певних випадках було застосовано fine-tuning: специфічно у певних місцях налаштовано програму сприймати дані не так, як визначають програмні бібліотеки. Ця практика також має застосовуватися із виключною обережністю, оскільки вона може погіршувати результати роботи програми із іншими наборами даних. Для початкового рішення такі особливості

можна було вважати нормальними, оскільки за мету було поставлено укладати вправи з певними визначеними параметрами (себто на тему «займенники» рівня володіння мовою українською мовою як іноземною A1) і для досягнення цієї мети «чисті» результати роботи програмних бібліотек не давали задовільного результату. Проте для повноцінних версій програми такі рішення мали б обов'язково бути уникнуті для уможливлення розширення її можливостей. Це й стало основним поштовхом для пошуку інших програмних бібліотек, які б давали кращі результати без запровадження сумнівних практик написання програмного коду.

Друга функція – вибір користувачем джерела вправи (db_worker). У розробленні цієї функції було використано модуль pandas [53], що призначений для роботи з файлами. Тип структури даних DataFrame дозволяє виконувати певні операції набагато простіше, ніж із модулем csv. Вихідні дані першої та вхідні дані третьої функції мають структуру списку списків, у яких перший елемент – порядковий номер цільового слова у вигляді цілого числа (int), другий – токенізоване речення у вигляді списку (list), третій – правильне цільове слово у вигляді текстового рядку (string), четвертий – список варіантів-кандидатів (list). Оскільки дані із CSV-файлу зчитуються як текстові змінні, ми використовуємо методи pandas, щоб конвертувати їх у потрібні нам типи даних: перша і третя колонки – вбудованими в бібліотеку методами, друга і четверта – за допомогою методу literal_eval вбудованого модуля ast [41], який у конвертованому в текст списку розпізнає список та перетворює його на такий. Також БД містить заголовки для колонок, роботу з якими pandas спрощує.

Третя функція – укладання вправ (task_grading_executor), відповідає за формування вправ, реєстрацію відповідей, оцінювання та супутні функції. Ця функція використовує вбудовані модулі: random (для вибору псевдовипадкових вправ та дистракторів), re та ast – для роботи з регулярними виразами (склеювання речення), перевірки типів змінних речення та відповіді.

При розробці програмного коду також було залучене стороннє програмне забезпечення, а саме DeepSource – утиліта для виправлення наявних помилок у

коді, передбачення можливих і загального його покращення шляхом уникнення сумнівних практик програмування, створення коментарів і документації [45]. Ця утиліта використовувалась через платформу GitHub, яка базується на системі Git [46], якою було вирішено скористатися задля кращих можливостей розповсюдження програмного продукту, зберігання його версій, а також, у можливому майбутньому – легкого масштабування його розробки до повноцінного програмного продукту для ринку, із залученням інших розробників для колаборації [47].

Отже, дослідивши методи оброблення природної мови, що використовуються в системах автоматичного укладання вправ, ми створили baseline solution для системи автоматичного укладання, яка будує правильний текст тестових вправ для вивчення займенників української мови для рівня A1. Також у процесі його створення було з'ясовано слабкі місця використаних бібліотек, пов'язані зокрема із обробкою природної мови. До того ж, було визначено, яким чином ці проблеми обмежують розвиток функціоналу програми напряму та через заходи, яких доводиться уживати, щоб уникнути деяких помилок.

2.3. Демонстрація роботи baseline системи автоматичного укладання тестових вправ

Укладання вправ. Користувач взаємодіє із програмою через посередництво файлу task_generator.py. Запускаючи цей файл, користувач потрапляє до консолі, де програма послідовно ставить питання, відповіді на які користувач уводить за допомогою клавіатури у те саме вікно. Спочатку йдуть ті питання, що налаштовують роботу системи (див. Мал. 2.2).

```
Введіть 1, якщо Ви хочете використовувати вже наявні тексти.
Введіть 2, якщо Ви хочете використати свій текст і внести його до бази даних.
Введіть 3, якщо Ви хочете використати свій текст без внесення його до бази даних.
2

Введіть назву файлу зі своїм текстом або повний шлях до нього:
test2.txt

Якщо ви НЕ хочете користуватися стандартною базою даних pronouns.csv, введіть назву файлу з вашою БД.
Якщо ви хочете користуватися стандартною БД, натисніть Enter.


Введіть бажану кількість питань. Наразі доступно: 470.
4

Введіть 1, якщо Ви хочете отримувати правильні відповіді одразу після завдань.
Введіть 2, якщо Ви хочете отримувати відповіді одразу після завдань, а також оцінку наприкінці.
Введіть 3, якщо Ви хочете отримувати оцінку та правильні відповіді наприкінці.
3
```

Малюнок 2.2. Вибір користувачем налаштувань програми

Користувач може обирати джерело вправ: 1) із попередньо укладеної БД вправ; 2) із власних текстів, якими можна поповнювати локальну БД; 3) із власних текстів без збереження їх у локальну БД. При виборі варіантів, що залучають власні тексти, користувач має ввести назву або повний шлях до текстового файлу. Якщо користувач укладав або хоче укласти власні – локальні БД з іншим змістом, він також може обрати цей варіант. Крім того, користувач обирає бажану кількість вправ, отримавши при цьому інформацію про загальну доступну кількість, і бажану форму оцінювання. При цьому для всіх питань, на які має відповісти користувач, передбачена можливість надання користувачем неприйнятної відповіді (до прикладу, через одрук). В такому разі програма перехоплює процес, не даючи помилці статися, і просить користувача ввести відповідь знову.

У систему закладено три типи оцінювання: 1) правильні відповіді надаються одразу після відповіді користувача; 2) правильні відповіді надаються одразу після відповіді користувача, проте після виконання всіх тестових вправ користувач отримує зведений результат: кількість та відсоток правильних відповідей відносно усіх виконаних вправ; 3) користувач отримує список усіх вправ, у яких зроблено помилку, з правильними відповідями, а також зведений

результат: кількість та відсоток правильних відповідей відносно всіх виконаних вправ.

Тестування за вправами. За обраними опціями система автоматично генерує тестові завдання, які використовуються у навчанні (див. Мал. 2.3 та 2.4).

```
___ п'ю каву.
А)  Я
Б)  Мені
В)  Мене
А

Не підкажете, котрий з ___ (є) другий (2)?
А)  вони
Б)  них
В)  їх
б
```

Мал. 2.3. Робота користувача із вправами (1)

Передбачається, що користувач уведе літеру (будь-якого регістру), яка відповідає правильному варіанту відповіді.

```
___ не підкажете найближчий?
А)  Вам
Б)  Ви
В)  Вас
Ви
Вибачте, ви ввели відповідь у некоректному форматі. Будь ласка, спробуйте ще раз.

___ не підкажете найближчий?
А)  Ви
Б)  Вам
В)  Вас
а



___ родина любить проводити час разом.
А)  Моя
Б)  Мою
В)  Моїй
б
```

Мал. 2.4. Робота користувача із вправами (2)

Якщо користувач уводить не літеру одного з варіантів відповіді, а щось інше (саме слово, чи будь-яку іншу літеру) – програма фіксує цю помилку та дозволяє користувачеві її виправити. По завершенню тестування програма генерує користувачеві результат його роботи (див. Мал. 2.5).

```
Ваша відповідь була неправильною у таких реченнях:
У реченні "___ родина любить проводити час разом." мав бути займенник "Моя".

Ви виконали 3 з 4 завдань правильно. Процент правильних відповідей: 75.0%
Натисніть Enter, щоб зупинити програму (можливо, зачиниться вікно програми).

Process finished with exit code 0
```

Мал. 2.5. Результат роботи користувача

Згідно з обраними налаштуваннями перед процесом тестування програма зазначає користувачеві його помилки, а також підраховує кількість та відсоток правильних відповідей. На цьому тестування завершується.

2.4. Розробка системи укладання тестових вправ із граматики української мови як іноземної

Відповідно до проблем, виявлених при розробці baseline solution, і загальних бажаних векторів розвитку системи, було виявлено чотири основні фокуси при розробці повноцінної версії системи: підбір програмних засобів (бібліотек мови програмування Python) для обробки природної мови, які б краще відповідали меті розробки; побудова повноцінної реляційної бази даних для запису, зберігання та зчитування даних про тестові вправи; створення віконного інтерфейсу користувача для взаємодії із системою та дотримання принципів об’єктно-орієнтованого програмування (ООП) при написанні програмного коду.

NLP-бібліотеки. Виходячи із помилок, які продукували протестовані програмні бібліотеки при розробці baseline solution, і небажаних практик програмування, яких довелося вжити для мінімізації ефекту цих помилок, було визначено необхідність пошуку їхніх альтернатив.

Насамперед нас цікавили системи, які при аналізі беруть до уваги контекст. Ця властивість дозволяє таким системам розводити омонімію словоформ (що було категорично неможливо із програмною бібліотекою pymorphy2). Також можливість отримувати точніші дані про не тільки

морфологічні, а й синтаксичні властивості словоформ потенційно могла б уможливити автоматичне укладання вправ із відкритою відповіддю, із постановкою слова у правильну граматичну форму, із декількома правильними варіантами відповіді тощо. Це можна реалізувати за рахунок постаналізу відповідей користувача на відповідність морфологічним властивостям словоформи із оригінального речення, на основі якого укладалось завдання; а також за рахунок пошуку можливих непорозумінь суб'єкт/об'єкт, до прикладу: «*Вони* купили шоколад» → варіанти «вони», «їх», «їм» → обрано «їм» → False negative.

Подібних бібліотек, які працюють із українською мовою, було знайдено дві. Одна – brown_uk/nlp_uk базується на Браунівському корпусі української мови та розробляється безпосередньо для української мови, українськими та українськомовними розробниками. Проте, на жаль, вона дозволяє здійснювати хіба базове зняття омонімії (заявлено про лише близько тисячі найпростіших випадків), а також базується передусім на Groovy та Java, тож потребувала б додаткових інструментів для підв'язки до Python [36]. Тому від неї було вирішено відмовитися.

Натомість ми обрали stanza, розроблену Стенфордською групою NLP. Ця бібліотека працює із більш ніж 70 мовами, використовуючи для тегування Universal Dependencies (UD) (treebank для української виданий у 2016 році [63]). Модель показує дуже високі показники ефективності для основних задач, які нас цікавлять: 99.78% і 97.48% – токенізація токенів-слів і речень відповідно; 96.74% – визначення частини мови за UD; 92.47% і 96.23% відповідно – визначення морфологічних характеристик і леми. Проте, на жаль, точність визначення синтаксичних зв'язків перебуває на рівні 87.83%, а певні морфологічні характеристики визначаються набагато гірше за інші; тому було вирішено на цьому етапі не розробляти для проєкту інші варіанти вправ. [48]

Також у програмному застосунку існує необхідність визначення словоформ слів. Цієї задачі, на жаль, stanza не виконує; brown_uk/nlp_uk

виконує, проте не підходить для зручного використання через описані вище причини. Тому було вирішено скористатись можливостями pymorphy3 [54], покращеної версії попереднього pymorphy2. Структурно змін вона не має та так само не має значень score (частотної імовірності відповідного морфологічного розбору слова) для української мови, проте, на відміну від оригіналу, підтримує новіші версії Python (включно з Python 3.11) і новіші версії Великого електронного словника української мови (ВЕСУМ) [34].

19 січня поточного року spacy, ще одна популярна бібліотека для обробки природної мови у Python [59], випустила реліз, який містив у собі пайплайни для української мови [65]. Однією з загальновідомих переваг цієї бібліотеки є її швидкість; також у її документації зазначено про такі можливості обробки української, як токенізація, векторне представлення слів, морфологічне тегування (за Universal Dependencies), визначення синтаксичних залежностей, лематизацію, розпізнавання іменованих сутностей (named entity recognition – NER), а також окремий засіб для finetuning правил визначення всіх інших тегів. На жаль, на момент випуску підтримки цією бібліотекою української мови ми вже визначили бібліотеки, із якими працюватимемо, і почали роботу із ними; проте у майбутньому доцільним буде дослідити її можливості, переваги й недоліки, і залучити до розробки замість наявних бібліотек або в комбінації із ними.

Реляційна база даних. Для більш зручної роботи із дійсно великими обсягами різноманітних даних необхідно залучати бази даних. Оскільки нашою метою, зокрема, було розширення можливостей програми та додання до неї підтримки інших частин мови та рівнів володіння українською мовою, це питання постало особливо гостро.

Було вирішено скористатися можливостями системи керування базами даних (СКБД) SQLite [61]. Ця система є легкою та швидкою СКБД, яка підтримує SQL (structured query language) — мову запитів для роботи з реляційними (табличними) базами даних. Вона має обмеження по типах даних,

а робота з нею легко інтегрується у мову програмування Python за допомогою вбудованої бібліотеки sqlite3 [62]. Обмеження в типах даних зіграли роль при написанні деяких методів роботи із БД – зокрема, у таблицях рівнів для частин мов зберігаються дані типу INT – цілі числа; проте нас цікавили значення Boolean (True/False, 1/0). Тому до таблиці записувалися значення 1 або 0, а після зчитування типи даних перетворювалися. Натомість таке спрощення системи типізації дозволило простіше визначити типи для текстових даних, оскільки зникла потреба турбуватися про максимальні обсяги збереженої інформації. Також для покращеної роботи із записом, переглядом і фільтруванням даних БД було залучено програмне забезпечення SQLite Expert Personal [60].

Поточна структура бази даних складається із 9 таблиць. 5 із них є обов'язковими та відповідають за базовий функціонал програми. Таблиця level містить у собі рівні володіння мовою – від A1 до C2, таблиця set – відомості про набори вправ із назвами й описами (укладені при реалізації проєкту та додані користувацькі). Таблиця sentences містить повні тексти оброблених програмою речень і слугує насамперед для уникнення зайвого повторення аналізу однакових речень, які можуть траплятися між різними наборами даних; натомість всі речення прив'язуються до всіх своїх наборів за допомогою проміжної таблиці sentence_set, яка емулює зв'язок між таблицями типу «багато-до-багатьох». Таблиця token містить дані про безпосередні складники вправ – усі токени, що у них містяться з указанням тексту, номеру речення, порядковому індексу у відповідному реченні, частини мови, а також ідентифікаційного номеру леми та форми (відмінкової, відмінково-числової тощо) для цільових частин мови (див. Мал. 2.6).

rowid	token_id	text	sentence_id	token_index	pos	pos_id	form
Click here to define a filter							
26	26	Також	3	0	adverb	(null)	(null)
27	27	у	3	1	adposition	(null)	(null)
28	28	мене	3	2	pronoun	1	gen
29	29	є	3	3	verb	(null)	(null)
30	30	бабуся	3	4	noun	(null)	(null)
31	31	та	3	5	conjunction	(null)	(null)
32	32	дідусь	3	6	noun	4	nom_s
33	33	.	3	7	punct	(null)	(null)

Мал. 2.6. Уривок з таблиці token

Також для кожної цільової частини мови (себто такої, укладання вправ для якої передбачене проєктом) є по дві таблиці. До прикладу, для займенників таблиця pronoun містить для кожного займенника його відмінкові форми, а також відомості про число та рід або особу (незмінні граматичні властивості для можливості розведення омонімії) (див. Мал. 2.7); а таблиця pronoun_level – дані про наявність кожної відмінкової форми у вимогах до опанування для кожного із рівнів володіння українською мовою як іноземною (див. Мал. 2.8). Відповідні дві таблиці – noun і noun_level було створено й для іменників. Скріншоти всіх таблиць, не поданих в основному тексті, містяться у Додатку Е.

rowid	pronoun_id	nom	gen	dat	acc	ins	loc	gender	number	person
Click here to define a filter										
1	1	я	мене	мені	мене	мною	мені	(null)	singular	1
2	2	він	його	йому	його	ним	ньому	masculine	singular	3
3	3	це	цього	цьому	це	цим	цім	neutral	singular	(null)
4	4	вона	її	їй	її	нею	ній	feminine	singular	3
5	5	хто	кого	кому	кого	ким	кім	masculine	singular	(null)
6	6	ти	тебе	тобі	тебе	тобою	тобі	(null)	singular	2

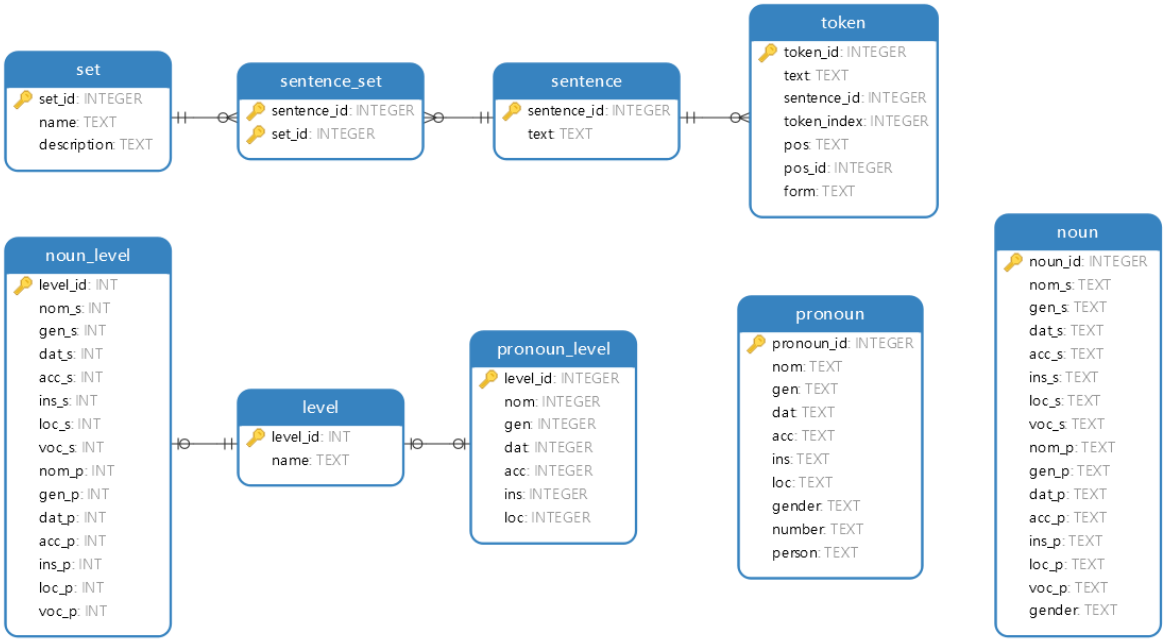
Мал. 2.7. Приклад таблиці лем для частини мови (уривок з таблиці pronoun)

rowid	level_id	nom	gen	dat	acc	ins	loc
Click here to define a filter							
1	1	1	1	1	0	0	0
2	2	1	1	1	1	1	1
3	3	1	1	1	1	1	1
4	4	1	1	1	1	1	1
5	5	1	1	1	1	1	1
6	6	1	1	1	1	1	1

Мал. 2.8. Приклад таблиці відповідності словоформ до рівнів для частини мови (таблиця pronoun_level)

Варто зазначити, що на рівні А2 люди, які вчать мову, мають мати базове уявлення про всі відмінки в українській мові; проте при цьому вони можуть обмежуватися знаннями про відмінювання окремих груп слів за цими відмінками (до прикладу, іменників твердої групи чоловічого роду) або про використання цих відмінків лише у певних комунікаційних ситуаціях (наприклад, для опису професії) чи синтаксичних контекстах (до прикладу, із певними прийменниками). У таблицях відповідності граматичних форм частин мови рівням після рівня А2 включно «активні» всі форми, тому вправи, укладені для цих рівнів, будуть однакові. Для більш детального налаштування укладання вправ відповідно до рівнів треба залучати, відповідно до описаних вище проблем, аналіз парадигм слів, лексичну та синтаксичну складові. Також гарним вектором розвитку проєкту було б доєднання лексичного плану у масштабі окремих лем для визначення їхньої належності до комунікативних ситуацій, розуміння яких вимагається із певного рівня. Для цього необхідно мати як мінімум словники такої лексики, розбитої по рівнях (тут також можливе залучення векторних представлень слів для пошуку можливих синонімічних слів, не долучених до словників, до прикладу, через word2vec [49]) і як максимум – утиліт для аналізу текстів на відповідність рівням володіння мовою.

Між таблицями бази даних також було укладено відповідні зв’язки «один-до-одного» й «один-до-багатьох» (див. Мал. 2.9).



Мал. 2.9. Схема створеної реляційної бази даних

Інтерфейс користувача. Очевидно, що загальні враження від будь-якого програмного продукту базуються не тільки на якості його безпосередніх функцій, а також на зручності роботи із ним. Порівняно із консольним інтерфейсом графічний (GUI) має суттєві переваги: структурованість, налаштованість, можливість відображення різних типів даних і в різних форматах тощо. Також наразі (порівняно із початком розвитку комп'ютерів) переважна більшість людей починають знайомство із комп'ютерними технологіями саме із графічних інтерфейсів, тому вони однозначно виступатимуть більш знайомими та простішими для опанування пересічною людиною.

Основних варіантів, у якому напрямку рухатись при розробці GUI, ми мали два: віконний або web-інтерфейс. Ми обрали перший варіант через такі причини:

1. Він є простішим для виконання, враховуючи наше знайомство із вбудованою бібліотекою Python для його створення tkinter [64]. За допомогою цієї бібліотеки код інтерфейсу можна розташовувати безпосередньо в коді основної програми, тісно їх пов'язуючи шляхом

виклику, до прикладу, методів одного класу всередині методів іншого класу, коли необхідно отримати новий вид інформації. Зокрема, ми також мали попередній досвід створення зв’язки між засобами tkinter та sqlite3, яка дозволяла відображати елементи баз даних у інтерфейсі та працювати із ними;

2. За допомогою tkinter інтерфейс доволі легко налаштовувати, оскільки для всіх елементів існують вбудовані стилі; також є багато вбудованих додаткових налаштувань, таких як messagebox – невелике вікно із коротким повідомленням для користувача. Також існує бібліотека customtkinter [4], яка має на меті «осучаснити» вигляд оригінального tkinter, який вперше вийшов у 1991 році та досі має стилі, схожі на ті, які Microsoft востаннє залучила у Windows 2000. «Сучасна» версія містить всі оригінальні функції бібліотеки та має функціонувати так само; проте вона робить стилі інтерфейсу більш мінімалістичними, наслідуючи популярний нині «плаский» дизайн, а також вміє підлаштовувати їх під налаштування системи (світлу/темну тему, кольорові акценти). Цю бібліотеку й було в результаті обрано для створення віконного інтерфейсу програми.
3. У випадку застосування віконного інтерфейсу користувач для запуску програмного продукту матиме завантажити його; таким чином, всі обчислювальні потреби програми мають покриватися комп’ютером користувача. Це дозволяє не витрачати ресурси (зокрема грошові) на пошук, створення й утримання серверу для обчислень.

Проте web-інтерфейс також мав би свої переваги. Насамперед він давав би швидший результат обчислень при внесенні нових наборів вправ у базу даних: в основі роботи stanza покладено neural pipeline – сукупність процесорів, кожен із яких виконує певну лінгвістичну задачу при обробці природної мови. Кожен із цих процесорів має певні вимоги та займає різний обсяг оперативної пам’яті при виконанні обчислень; чим більші обсяги оброблюваної інформації –

тим більший і обсяг зайнятої оперативної пам’яті. Також треба враховувати, що різні комп’ютери мають різний обсяг доступної пам’яті і старі можуть справлятися із запитами програми набагато гірше, ніж нові; це, зокрема, мало б значення у контексті використання машин, які маються на базі освітніх закладів. Тож якби для обчислень, необхідних програмі, було б залучено сторонній сервер із віртуальною машиною, це потенційно дало б уникнути проблем із довгим очікуванням результату обробки або взагалі її неможливістю.

Це також має значення у питанні доступності та поширеності програмного засобу: наразі багато людей можуть виявити бажання користуватися застосунком із своїх смартфонів, враховуючи їхню портативність і постійну доступність. В такому разі застосунок з інтерфейсом на базі tkinter або customtkinter був би недоступним і довелося б розробляти окреме рішення (до того ж, враховуючи різноманітність операційних систем смартфонів і різні вимоги до створення та розповсюдження застосунків для використання на них). Web-інтерфейс, натомість, був би доступний на будь-якому пристрої із підключенням до Інтернету та web-браузером.

Також можна зауважити, що віконний інтерфейс, на відміну від web-інтерфейсу, дозволяв би працювати із застосунком поза підключенням до мережі; натомість тут виявляється і мінус у вигляді необхідності додаткових махінацій для оновлення версій такого програмного забезпечення на комп’ютерах користувачів, коли публікуються покращення результатів обробки, бази даних, самого інтерфейсу та програмного коду в цілому.

Дотримання принципів об’єктно-орієнтованого програмування. Ще на етапі розробки baseline було визначено, що архітектура його програмного коду визначалася обмеженим предметом дослідження – цей код мав видавати вправи виключно для займенників рівня володіння мовою A1; і ця архітектура жодним чином не могла б бути дотриманою при його розширенні. Натомість для створення багатьох зв’язків між оброблюваними об’єктами природної мови,

зокрема закладення початкових архітектур об’єктів і детальнішого визначення їхніх підтипів за допомогою наслідування влучно можна було б застосувати принципи об’єктно-орієнтованого програмування. В результаті побудови програми за саме такими принципами можна описати визначені класи (типи об’єктів) у програмі.

Один із класів, SQL, повністю відповідає за взаємодію із базою даних і містить усі використані у програмі SQL-запити та методи створення БД вправ. Всі методи цього класу є статичними: вони можуть використовуватися без створення екземплярів класу, що є для нас дуже зручним, оскільки ми не знаємо, які саме методи класу знадобляться при використанні програми (користувач може вирішити лише завантажити вправи у БД або лише виконати вправи) і створювати окремий об’єкт цього класу для кожного використання було б поганою практикою, оскільки займало б зайву оперативну пам’ять. Основних методів клас має два: перший, choose_tasks, відповідає за вибір вже наявних вправ із бази даних за заданими користувачем параметрами (а саме – цільова частина мови, цільовий рівень володіння мовою та обраний набір вправ); другий, add_tasks, натомість дає можливість завантажити нові вправи до БД.

При розробці першого методу виникли певні проблеми, які було згодом вирішено. По-перше, треба було уникати сценарію, за якого одне речення може містити два і більше цільових слів, і користувач, відповідно, отримує дві і більше вправ на базі одного речення. Таким чином він може запам’ятати слова із нього і результат тестування не буде достовірним. Цього не було передбачено у попередній версії розробки, проте у цій така функція стала цілком можливою для розробки завдяки наявності бази даних, де кожне слово має приписаний ідентифікатор речення та може бути перевірене на наявність інших цільових слів поруч із собою. По-друге, майже та сама причина спродукувала ще одну помилку – коли у реченні було наявно декілька цільових слів, проте деякі з них не відповідали за формою обраному цільовому рівню, вони не записувалися і в

результаті були відсутні у реченні, що надавалося до вправи. Це було вирішено модифікацією SQL-запиту. Загалом цей метод містить і виконує 3 запити до бази даних.

Другий метод має 6 запитів до БД, а також викликає додатковий, «вкладений» у нього метод add_tokens, який вставляє записи про конкретні токени у відповідну таблицю. Цей метод має 2 запити і також посилається на ще один вкладений метод – add_pos, який додає дані про відсутні леми до БД і містить 2 запити. Загалом функція запису до бази даних із залученням цих трьох методів має уникати всіх можливих помилок, насамперед спроб вставити до таблиці дані, що дублюють наявні, або створювати від однієї таблиці зв'язок до таких даних іншої, яких не існує.

Також наразі є два невеликих методи (get_levels і get_sets), які слугують для оформлення інтерфейсу, а саме заповнення можливих опцій для вибору користувачем цільових параметрів і відображення користувачеві вже зайнятих назв наборів. Ще один метод, check_sets, має на меті уникнути можливі помилки при спробах записати до БД дані про нові набори вправ, які назвами дублюють вже наявні. Ці методи містять по одному запиту кожен.

Наступні декілька класів відповідають за аналіз тексту; результат обробки методів цих класів передається до класу SQL для запису до бази даних. Тут було вирішено, за аналогією до даних, які записуються до БД, створити такі класи, як: Text для обробки цілого користувацького тексту, Sentence для обробки окремих токенів-речень і Token для обробки токенів-слів. Sentence має наслідування від Text, оскільки вони мають певні спільні змінні; а Token передбачений як основний parent class для всіх можливих частин мови: тих, які вважаються цільовими (іменник – Noun і займенник – Pronoun) або потребують особливої обробки (pluralia tantum – Pluralia та власні назви – ProperNoun).

Клас Text виконує такі функції обробки тексту, як: його зчитування із файлу (метод read_text); уніфікація лапок й апострофів у ньому задля

уникнення проблем із SQL, де одинарні й подвійні лапки мають синтаксичне значення (метод clean_text, який всі подвійні лапки перетворює на відповідні закриваючі чи відкриваючі подвійні французькі лапки-ялинки; одинарні лапки перетворює на апострофи або одинарні лапки-ялинки у випадку вкладеного цитування); токенізацію й створення об'єктів класу Sentence і передання їх на запис до БД (analyse_text). Об'єкти дочірнього класу Sentence мають лише два методи: один (analyse_tokens) обробляє токени й створює об'єкти для кожного з них; інший (get_dict) збирає докупи результат обробки всього речення для передачі батьківському класу.

Клас Token має такі методи, як get_pos, get_form, get_forms і get_features. Перші два визначають для кожного слова його частину мови й для цільових частин мови – форму; другі два визначають повний список словоформ леми слова й постійні морфологічні характеристики. Об'єктами цього класу стають слова, які не є цільовими або не обробляються окремо, тому єдиний «активний» метод у цього класу – перший: було вирішено для кожного токену, навіть нецільового, визначати його частину мови для можливого простішого доповнення БД новими цільовими частинами мови у майбутньому. Також клас має (аналогічно до класу для речень) метод get_dict, який повертає відомості про токен для подальшого «зшивання» та додавання до БД.

Дочірні від Token класи переймають від нього ініціалізацію й метод повернення даних у вигляді словника. Проте інші методи для цих класів мають визначені команди. get_pos одразу повертає частину мови. get_form визначає для цільових частин мови їхню форму; у класі Pronoun це перші три літери назви відмінку: до прикладу, знахідний → accusative → "acc"; у Noun через нижнє підкреслення також додається перша літера назви числа: таким чином, знахідний відмінок однини – "acc_s". get_forms підбирає серед можливих морфологічних розборів слова від pymorphy3 потрібну лему й укладає словник із її словоформ (назви словоформ відповідають таким самим правилам, як

описано для get_form). get_features визначає для займенників їхнє число, рід і особу (де такі категорії передбачені парадигмами); для іменників – їхній рід.

Дочірні класи Token Pluralia та ProperNoun введені для уникнення можливих помилок розборів (насамперед у pymorphy3). Для форм pluralia tantum неможливо утворити форми однини; певні власні імена є абревіатурами, які варто оминати, а певні – є у деяких формах омонімічними до загальних назв і заважають пошукові потрібних лем (до прикладу, словоформа «кота» в першу чергу визначиться як утворена від леми «Кот» – прізвища; таким чином ми не зможемо отримати бажану лему «кіт»). Для обидвох цих винятків програма призначає окреме значення частини мови (“pluralia” та “proper_noun” відповідно); цим можна у майбутньому скористатися й розробити алгоритм залучення цих частин мови до укладання вправ.

Також є два класи, які власне утворюють графічний віконний інтерфейс програми. Клас Application наслідує customtkinter.CTk і створює вікно для інтерфейсу, визначаючи також його назву й основні поля, і створює об’єкт класу Body. Клас Body наслідує customtkinter.CTFrame й містить всі елементи інтерфейсу. Зокрема, при ініціалізації він запускає метод starting_screen, який відображає для користувача вітання та дві опції для роботи з програмою – початок тестування або завантаження нових прав. Кожна з цих опцій виконується завдяки окремим методам. Перша – запуском методу configure_testing, який дає користувачу обрати цільові частину мови, рівень і набір вправ; із цих даних підраховується кількість доступних вправ і запускається метод get_tasks, який дає обрати бажану кількість. Потім методом start_testing вікно переналаштовується на відображення вправ; у сполученні з методом play_task ітеруються всі псевдовипадково обрані вправи. Також залучено метод make_task, який для кожної вправи створює завдання: збирає із токенів речення, при цьому «приховуючи» цільове слово маскою «___» і перевіряючи необхідність пробілів до та після токенів (оскільки певні токени – деяка пунктуація не відділяються пробілами). Також метод show_results

наприкінці тестування відображає його результати; якщо при його проходженні користувач припустив помилки, то він має змогу також переглянути їх і правильні варіанти відповіді, що забезпечується методом show_mistakes.

Друга ж опція має дещо простішу структуру, оскільки для неї вікно оновлюється лише один раз. Метод upload_tasks керує цим оновленням і створює поля із підказками для внесення користувачем даних про його набір: назви, опису й текстового файлу, на основі якого бажається укласти вправи. Вибором файлу при цьому керує метод choose_file, який повертає повний шлях до обраного у Провіднику файлу; метод process_tasks створює об’єкт класу Text, який, як було описано раніше, здійснює аналіз тексту та запис його результатів до БД. Також при створенні набору метод show_names дозволяє переглянути вже зайняті, недоступні назви.

Ще два класи – Spinbox і Messagebox слугують шаблонами для віджетів інтерфейсу. Перший наслідує customtkinter.CTkFrame є полем для вводу чисел із кнопками для збільшення та зменшення вказаного числа та використовується у Body.get_tasks() для зручного вибору кількості вправ. Код для нього був сформований на основі інструкції із документації customtkinter [43]. Другий клас має на меті замінити діалогове вікно (messagebox) із стандартного tkinter, оскільки обрана нами основна бібліотека інтерфейсу не має такого функціоналу. Клас Messagebox наслідує customtkinter.CTkToplevel. Також варто зазначити, що із tkinter було використано один вид віджету, відсутній у customtkinter – tkinter.ttk.Separator. Він, на відміну від діалогового вікна, гарно підлаштовується під колір інтерфейсу та не потребував заміни.

Поза класами також існують такі команди та визначення, як: імпортування всіх потрібних програмних бібліотек до програми; змінні, що містять у собі визначені відповідності між певними значеннями, які присвоєні об’єктам у самій програмі, базі даних або бібліотеках (зокрема, між позначеннями доступних для укладання вправ частин мов і частин мов загалом, і граматичних категорій роду, числа, особи й відмінку). Також наприкінці

програмного коду є команди для безпосереднього запуску застосунку, запуску pymorphy3 і створення зв’язку із базою даних.

На цьому етапі розробки також було залучено технології DeepSource, Git та GitHub; насамперед, задля безперервності розробки, було передбачено, що й baseline solution, і повноцінна розробка, і всі можливі подальші версії продукту будуть міститися в одному репозиторії. Також було використано можливості моделі ChatGPT для пошуку помилок у програмному коді, шляхів їхнього розв’язання, отримання порад стосовно кращих практик програмування тощо.

За допомогою утвореного програмного засобу було укладено вправи на матеріалі текстів рівня володіння українською мовою рівня A1, зібраних Еліною Півкіною. У базі даних, яка долучена до проєкту, цей набір вправ має назву “corpus”. Всього було проаналізовано 847 речень. Кількість вправ, які можуть бути укладені на їхній базі, відрізняється залежно від обраних користувачем налаштувань; до прикладу, для рівня A1 і цільової частини мови – займенника вона становить 426 вправ; для тієї ж частини мови, але рівня A2 – 457 вправ.

На прикладі укладених вправ також було помічено деякі недоліки засобів автоматичного морфологічного аналізу. Через них, зокрема, не було знайдено леми для 160 уживань іменників. Прослідковуються такі патерни проблем, як:

1. Нездатність pymorphy3 розібрати слова, що містять апострофи;

2. Нездатність stanza виокремити singularia tantum;

3. Невірно визначений відмінок у stanza;

4. Власні назви, не визначені як такі stanza;

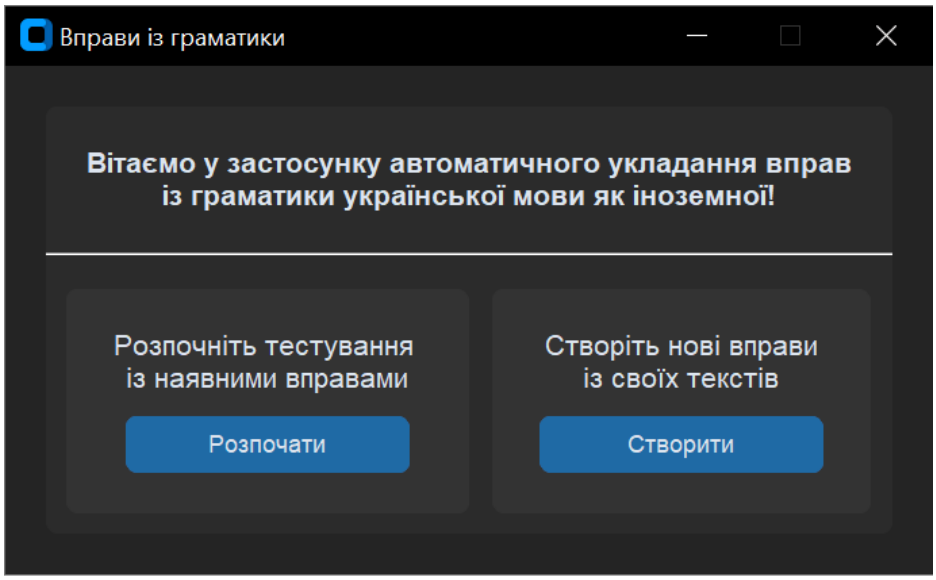
Повний перелік словоформ, для яких програмою не було визначено леми, разом із реченнями, звідки вони походять, подано у Додатку Є.

Посилання на повний програмний код застосунку міститься у Додатку Д. Програмний код також містить документацію: docstring на початку скрипту,

описи для кожного класу, кожного методу класів і у методах із великим обсягом – коментарі стосовно команд, виконаних на певних рядках.

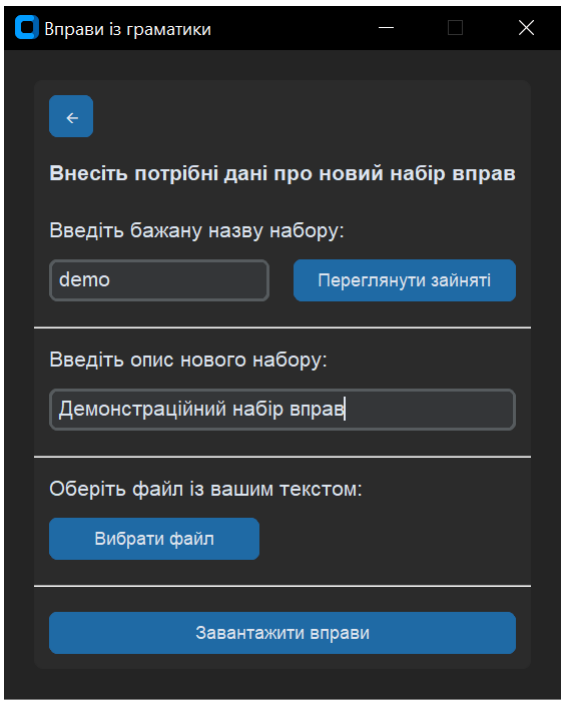
2.5. Демонстрація роботи системи укладання тестових вправ із граматики української мови як іноземної

Початок роботи. Користувач взаємодіє із програмою через посередництво файлу main.py. Запускаючи цей файл, він бачить віконний інтерфейс програмного застосунку із початковим екраном. Тут він може обрати з двох опцій: розпочати тестування або укласти новий набір вправ із свого тексту (див. Мал. 2.10). Обираючи будь-яку опцію, він має можливість на будь-якому етапі (окрім безпосередньо тестування) повернутись до попереднього етапу, в тому числі – початкового екрану, за допомогою спеціальної кнопки.

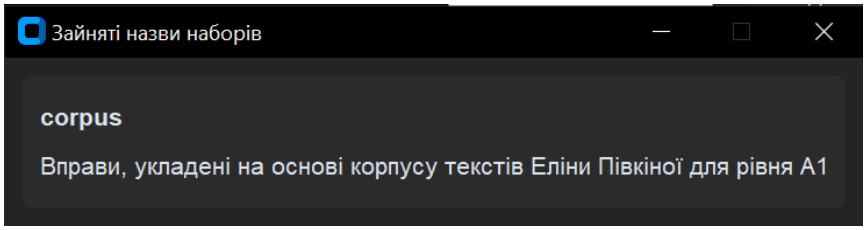


Мал. 2.10. Початковий екран віконного інтерфейсу програми

Укладання вправ. Обираючи укладання вправ, користувач бачить оновлене вікно, де він має ввести дані про свій новий набір: його назву, опис, та за допомогою окремої кнопки обрати текстовий файл із своїм текстом (див. Мал. 2.11). Також він може переглянути інформацію стосовно вже зайнятих назв наборів, натиснувши спеціальну кнопку (див. Мал. 2.12).

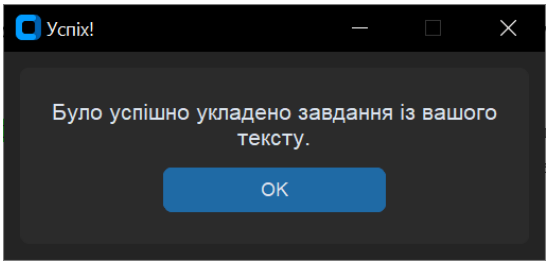


Мал. 2.11. Внесення даних про новий набір вправ



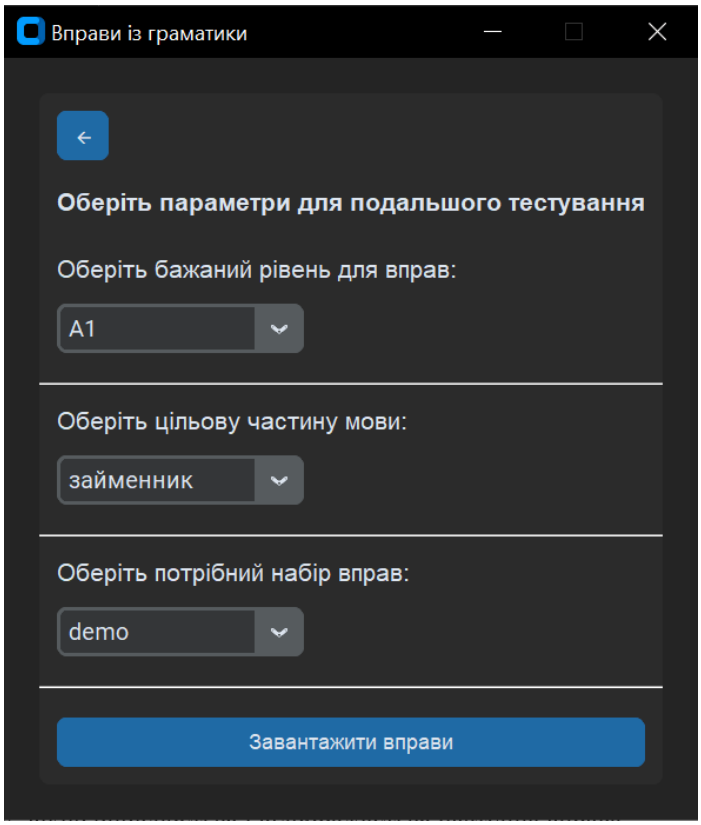
Мал. 2.12. Вже зайняті назви наборів вправ разом із описами, відображені в окремому вікні

Якщо вправи було успішно укладено – користувачу відображається діалогове вікно із повідомленням про це (див. Мал. 2.13) і він знову опиняється на початковому екрані. Подібні діалогові вікна також передбачені на цьому й подальших етапах для випадків, коли відбуваються помилки (програма не змогла обробити текст) або перехоплюються потенційні помилки (до прикладу, користувач спробував увести вже зайняту назву набору).

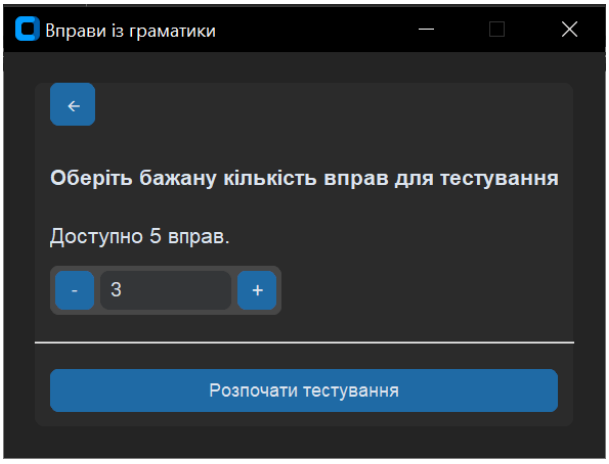


Мал. 2.13. Діалогове вікно із повідомленням про успішне укладання вправ на основі тексту, завантаженого користувачем

Налаштування тестування. Коли користувач хоче розпочати тестування, спочатку треба його налаштувати. Для цього обираються цільові рівень володіння українською мовою як іноземною та частина мови, а також бажаний набір вправ. (див. Мал. 2.14). Згідно з обраними параметрами програма підраховує доступну кількість вправ і дає можливість обрати, скільки хоче виконати користувач (див. Мал. 2.15).

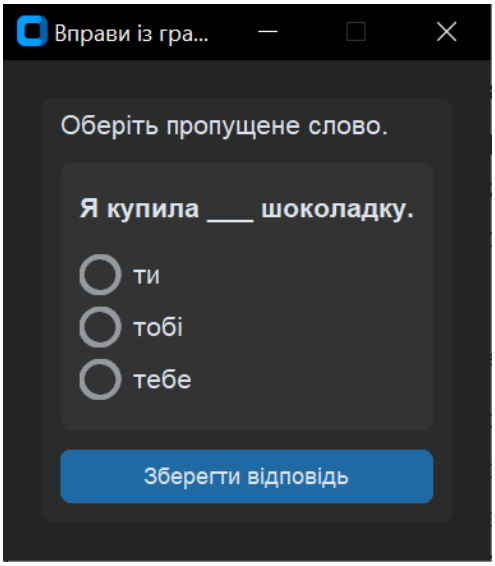


Мал. 2.14. Вибір бажаних опцій для тестування



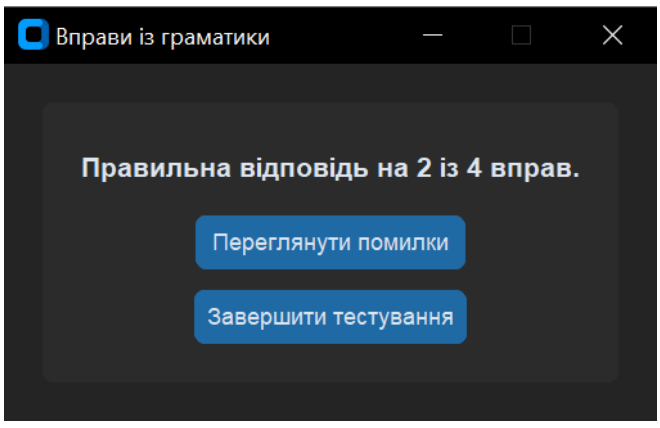
Мал. 2.15. Вибір бажаної кількості вправ для тестування

Тестування та результати. Тепер розпочинається сам процес тестування. У вікні користувач бачить речення, де цільове слово приховане «___», і три варіанти відповіді (див. Мал. 2.16). Коли він натискає кнопку «Зберегти відповідь», вікно оновлюється і відображається наступна вправа.

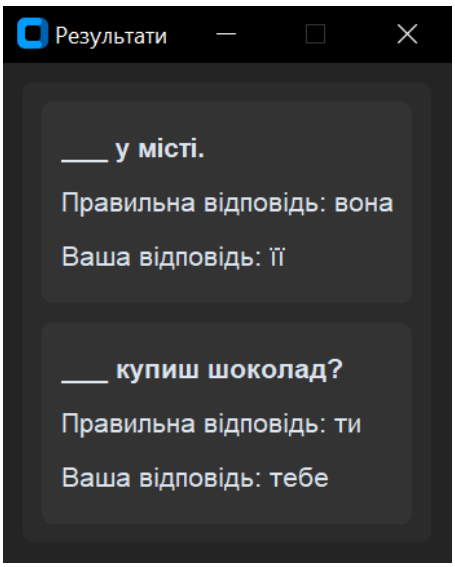


Мал. 2.16. Вікно з однією із вправ при тестування

Коли всі вправи було збережено, користувач може переглянути свій результат (див. Мал. 2.17). Якщо під час тестування він припустився помилок, то також має можливість переглянути їхній список разом із виправленнями (див. Мал. 2.18). Після цього його повертає на початковий екран.



Мал. 2.17. Діалогове вікно із результатом тестування



Мал. 2.18. Вікно із помилками, яких користувач припустився при проходженні тестування, та правильними варіантами відповіді

ВИСНОВКИ

Здійснене дослідження ґрунтується на поєднанні міждисциплінарних методів лінгводидактики та комп'ютерної лінгвістики та демонструє інтеграцію сучасної лінгвістики із методикою викладання української мови як іноземної та з інформаційними технологіями. Розроблена автором система автоматичного укладання тестових вправ – перша в українському прикладному мовознавстві навчальна лінгвістична система, яка автоматично конструюється за заданими текстами та параметрами.

Для дослідження, зокрема, було здійснено аналіз потреби й вимог до вивчення граматики української мови як іноземної; проаналізовано зміст Стандартизованих вимог до рівнів володіння української мови як іноземної; з’ясоване місце вправ, зокрема граматичних, у процесі вивчення іноземних мов. Також був здійснений аналіз подібних розробок, як для граматичного аспекту мовлення, так і для лексичного, для декількох мов світу, і з’ясовано засоби, якими автори цих розробок досягали мети.

Для початкового окреслення можливого фунціоналу програми, з’ясування принципів її роботи та пошуку можливих перепон було створено baseline solution – версію проєкту, обмежену лише однією темою (займенники рівня володіння мовою A1), без складної об’єктно-орієнтованої структури, інтерфейсу та реляційної бази даних. Після цього було побудовано план для розробки повноцінного програмного продукту.

У процесі розроблення програмного забезпечення було окреслено низку проблем, пов’язаних із недосконалістю програмних бібліотек для обробки української мови (NLTK, pymorphy2, pymorphy3, tokenize_uk, stanza). Ці проблеми не дають можливості заявляти про абсолютну точність і правильність укладених програмою граматичних вправ; тому вважаємо, що для гарантування гарних результатів навчання за допомогою системи необхідна підтримка викладачів української мови як іноземної – щоб вони фільтрували вправи, які

надаються учням. Ці проблеми частково були розв’язані там, де було можливо виокремити певні незадовільні результати та з’ясувати алгоритм їхнього перехоплення.

Проте кращим результатам роботи програми посприяло б покращення роботи використаних програмних бібліотек; імовірно, що доєднання інших розробок, зокрема brown-uk/nlp_uk або spacy також допомогло б збільшити точність. Певні їхні функції, зокрема автоматична синтаксична розмітка у spacy, допомогла б і розробити новий функціонал – формування інших видів вправ (fill-in-the-gap із підказкою та без, виправлення помилок, multiple choice тощо). Також покращене розуміння контексту у реченнях допомогло б вточнити приналежність словоформ до конкретних рівнів володіння мовою.

Дуже необхідними для покращення можливостей пристосування програми до різних рівнів видаються матеріали (словники, корпуси текстів) та інструменти для визначення рівневої приналежності лем і речень. Також у перспективі можна залучити й повноцінну лексичну складову аналізу тексту, зокрема векторні представлення слів (word2vec) для вправ із відкритою відповіддю для відкритих класів слів (до прикладу, іменників).

Розробка повноцінної реляційної бази даних для збереження укладених вправ, а також переведення програми у парадигму об’єктно-орієнтованого програмування створили міцну базу для легкого подальшого розширення предмету дослідження; насамперед – укладання вправ для інших цільових частин мови.

Було розроблено віконний графічний інтерфейс користувача для покращення взаємодії із системою. У перспективі програму може бути оформлено також як web-застосунок, більш доступний завдяки кросплатформеності та меншим вимогам до обчислювальних потужностей для укладання вправ. Можливе також і створення мобільного застосунку. Також потенційно надзвичайно корисною функцією була б розробка окремого інтерфейсу для викладачів, який би дозволяв відбирати конкретні (не псевдовипадкові) вправи та утворювати із них тестові аркуші для друкування.

Корисним для подальшого розвитку застосунку було б проведення дослідження ринку, а саме опитувань людей, які вчать українську мову як іноземну та людей, які її викладають. Це дозволило б визначити перелік функцій, у яких насамперед зацікавлені потенційні користувачі, а також слабкі місця, зокрема такі, що пов’язані із методикою викладання української мови як іноземної. Також корисним став би повноцінних аналіз існуючих вправ – тих, які є у підручниках з української мови як іноземної й укладених викладачами самостійно.

Також, при достатній кількості укладених вправ і залученні додаткових сил для перевірки їхньої правильності, відповідності Стандартизованим вимогам до рівнів володіння української мови як іноземної та загальним принципам лінгводидактики та методики навчання іноземних мов, можна було б розмітити укладені вправи та використати технології машинного навчання для переходу із укладання вправ на основі заданих текстів до генерування подібних вправ.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Алексієнко Л. А., Дарчук Н. П., Зубань О. М., Сорокін В. М. Методика створення автоматизованої навчальної системи з української мови // Педагогічний процес: теорія і практика. Збірник наукових праць. – Вип.3. – Київ, 2006. – С. 11-22.
2. Антонів О. В., Паучок Л. М. Українська мова для іноземців. Модульний курс: навчальний посібник. Київ: ІНКОС, 2012. 268 с.
3. Бабич В. О., Костіков М. П. Проєктування навчальної системи для вивчення українських займенників // Матер. IX Міжнар. наук.-техн. Internet-конф. «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», 25 листоп. 2022 р. – К.: НУХТ, 2022. – С. 121–122.
4. Бабич Вероніка, Автоматичне генерування тестових завдань із граматики української мови як іноземної // Всеукраїнська наукова конференція «Мова, література, переклад у комунікативному просторі сучасного світу» з нагоди 20-річчя Навчально-наукового інституту філології Київського національного університету імені Тараса Шевченка, 4–5 листопада 2021, – С. 16. URL: https://docs.google.com/document/d/1D16SvrG_cMIEYNz2af4_Uv3ox-zr9CokXSroySD5_Xw (дата звернення: 14.06.2023).
5. Баран Н. А. Окремі аспекти викладання української мови як іноземної. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 3-16.
6. Белевцова С. О. Проблеми щодо вивчення української мови в іншомовній аудиторії. Збірник наукових праць Харківського національного педагогічного університету імені Г. С. Сковороди «ПРАВО». Харків, 2018, №29. С. 147-150.

7. Бєй Л. Б., Тростинська О. М. Проблеми викладання української мови різним категоріям іноземних студентів. Викладання мов у вищих навчальних закладах освіти на сучасному етапі. Міжпредметні зв'язки. Харків, 2008, №12. С. 42-49.
8. Бруннер К. Актуальні проблеми викладання української мови як іноземної. Теорія і практика викладання української мови як іноземної. Львів, 2008, №3. С. 12-15.
9. Вінницька В. М. Українська мова як іноземна (книга перша). Розмовно-фонетичний курс. Київ: ВПЦ "Київський університет", 2017. 184 с.
10. Дугин С. П., Коньок О. П., Косенко Ю. Г. Learn Ukrainian Now. Teach-yourself book. 3-те вид., стер. Суми: Університетська книга, 2020. 120 с.
11. Костіков М. П. Інформаційна технологія підтримки процесу навчання граматики іноземної мови у ВНЗ : дис. ... канд. техн. наук : 05.13.06. – К., 2016. — 160 арк.
12. Мазурик Д. В. Українська мова для іноземців. Крок за кроком. Харків: Фоліо, 2017. 288 с.
13. Погоріла С. Г. Елементи мовної підготовки іноземних студентів у процесі вивчення української мови як іноземної в умовах довузівської підготовки. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 51-53.
14. Рудакова Т. М. Мовний бар’єр на заняттях з української мови як іноземної: причини виникнення та способи подолання. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 60-61.
15. Самусенко О. М. Розмовна практика як спосіб подолання мовного бар’єра у вивченні іноземної мови. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 62-64.

16. Світлик М. Д. Ігрові вправи та рольові ігри – ефективні методи викладання української мови як іноземної. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 60-61.
17. Тимчук І. М. Проблеми навчання іноземних студентів у процесі вивчення української мови в умовах довузівської підготовки. Актуальні проблеми викладання української мови як іноземної: матеріали міжнародної науково-практичної конференції, м. Біла Церква, 19 квітня 2019 р. Біла Церква, 2019. С. 70-73.
18. Aldabe I. Automatic Exercise Generation Based on Corpora and Natural Language Processing Techniques. San Sebastián, 2011. 302 pp.
19. Boas F. Introduction / Franz Boas // Handbook of American Indian Languages / Franz Boas. – Washington: Government Print Office (Smithsonian Institution, Bureau of American Ethnology), 1911. – (1). – PP. 1–83.
20. Cook V. Second Language Learning and Language Teaching. Fourth Edition. London, 2008. 317 pp.
21. Cubric M., Tosic M. Towards automatic generation of e-assessment using semantic web technologies. International Journal of E-Assessment, May 2011. PP. 1-9.
22. Hill J., Simha R. Automatic Generation of Context-Based Fill-in-the-Blank Exercises Using Co-occurrence Likelihoods and Google n-grams. Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications. San Diego, June 2016. PP. 23-30.
23. Knoop S., Wilske S. WordGap - Automatic Generation of Gap-FillingVocabulary Exercises for Mobile Learning. Proceedings of the second workshop on NLP for computer-assisted language learning at NODALIDA 2013. Oslo, 2013. PP. 38-47.

24. Korobov M. Morphological Analyzer and Generator for Russian and Ukrainian Languages. Analysis of Images, Social Networks and Texts, 2015. PP. 320-332.
25. Kuzmenko, E., & Fenogenova, A. (2016). Automatic generation of lexical exercises.
26. Levy M. Computer-assisted language learning. Context and Conceptualization. New York, 1997. 232 pp.
27. Mine T., Shoudai T. The Design and implementation of Automatic Exercise Generator with Tagged Documents based on the Intelligence of Students: AEGIS. 2000. 9 p.
28. Papasalouros A., Kanaris K., Kotis K. Automatic Generation Of Multiple Choice Questions From Domain Ontologies. IADIS International Conference e-Learning 2008, Amsterdam, The Netherlands, July 22-25, 2008. Amsterdam, 2008. PP. 427-434.
29. Perez N., Cuadros M. Multilingual CALL Framework for Automatic Language Exercise Generation from Free Text. Proceedings of the EACL 2017 Software Demonstrations, Valencia, Spain, April 3-7 2017. Valencia, 2017. PP. 49-52.
30. Sapir E. The Status of Linguistics as a Science / Edward Sapir. // Language. – 1929. – №5(4). – PP. 207–214.
31. Selinker L. Language transfer. General Linguistics. State-College, 1969, №9. PP. 67-92.
32. Whorf B.L. The relation of habitual thought and behavior to language. Language, Thought, and Reality. Selected Writings of Benjamin Lee Whorf ; edited by J.B. Carroll. Cambridge, Massachusetts : MIT Press, 1956. PP. 134–159.
33. Wu S., Witten I., Edwards E., Nichols D., Aquino R. A Digital Library of Language Learning Exercises. iJET International Journal of Emerging Technologies in Learning, January 2006. 7 pp.

ЕЛЕКТРОННІ ДЖЕРЕЛА

34. Великий електронний словник української мови. URL: https://r2u.org.ua/vesum/ (Last accessed: 14.06.2023).
35. Іноземні студенти в Україні. URL: https://studyinukraine.gov.ua/zhittya-v-ukraini/inozemni-studenti-v-ukraini/ (дата звернення: 14.06.2023).
36. Корпус сучасної української мови (БрУК): brown-uk/nlp_uk. URL: https://github.com/brown-uk/nlp_uk (Last accessed: 14.06.2023).
37. Проект Стандарту української мови як іноземної. Рівні загального володіння та діагностика. URL: https://base.kristti.com.ua/?p=1442 (дата звернення: 14.06.2023).
38. Стандартизовані вимоги до рівнів володіння українською мовою як іноземною A1-C2 // Схвалено рішенням колегії Міністерства освіти і науки України протокол від 22.05.2018 № 5/1-7 https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-materiali-ukrainskim-shkolam-za-kordonom (дата звернення: 14.06.2023).
39. Сучасна українська мова для старшокласників та абітурієнтів. Електронний підручник з інтерактивним тестуванням. – 2005. http://www.mova.info/pidruchn.aspx (дата звернення: 14.06.2023).
40. 2022 Duolingo Language Report. URL: https://blog.duolingo.com/2022-duolingo-language-report/?fbclid=IwAR1RxI5LFkaRnsSpGEdmIFugHQPLa-Q1AB_sNkSaUkwhSN9SR0iJGPRMXng (Last accessed: 14.06.2023).
41. Ast. URL: https://docs.python.org/3/library/ast.html (Last accessed: 14.06.2023).
42. Collections. URL: https://docs.python.org/3/library/collections.html (Last accessed: 14.06.2023).
43. Create Spinbox | CustomTkinter. URL: https://customtkinter.tomschimansky.com/tutorial/spinbox/ (Last accessed: 14.06.2023).

44. Customtkinter. URL: https://pypi.org/project/customtkinter/0.3/ (Last accessed: 14.06.2023).
45. DeepSource. URL: https://deepsource.com/ (Last accessed: 14.06.2023).
46. Git. URL: https://git-scm.com/ (Last accessed: 14.06.2023).
47. GitHub. URL: https://github.com/ (Last accessed: 14.06.2023).
48. Model Performance - Stanza. URL: https://stanfordnlp.github.io/stanza/performance.html (Last accessed: 14.06.2023).
49. Models: lang-uk. URL: https://www.lang.org.ua/en/models/ (Last accessed: 14.06.2023).
50. Natural Language Toolkit. URL: http://www.nltk.org/ (Last accessed: 14.06.2023).
51. NumPy. URL: https://pypi.org/project/numpy/ (Last accessed: 14.06.2023).
52. OpenCorpora: открытый корпус русского языка. Граммемы. URL: http://opencorpora.org/dict.php?act=gram (Last accessed: 14.06.2023).
53. Pandas. URL: https://pypi.org/project/pandas/ (Last accessed: 14.06.2023).
54. Pymorphy3. URL: https://pypi.org/project/pymorphy3/ (Last accessed: 14.06.2023).
55. Python 3.x. URL: https://docs.python.org/3/ (Last accessed: 14.06.2023).
56. Random. URL: https://docs.python.org/3/library/random.html (Last accessed: 14.06.2023).
57. Re. URL: https://docs.python.org/3/library/re.html (Last accessed: 14.06.2023).
58. Simple python lib to tokenize texts tokenize_uk. URL: https://github.com/lang-uk/tokenize-uk (Last accessed: 14.06.2023).
59. Spacy. URL: https://spacy.io/ (Last accessed: 14.06.2023).
60. SQLite Expert. URL: https://www.sqliteexpert.com/ (Last accessed: 14.06.2023).
61. SQLite. URL: https://www.sqlite.org/index.html (Last accessed: 14.06.2023).
62. SQLite3. URL: https://docs.python.org/3/library/sqlite3.html (Last accessed: 14.06.2023).

63. Stanza. URL: https://stanfordnlp.github.io/stanza/ (Last accessed: 14.02.2022).
64. Tkinter. URL: https://docs.python.org/3/library/tkinter.html (Last accessed: 14.06.2023).
65. Ukrainian. spaCy Models Documentation. URL: https://spacy.io/models/uk (Last accessed: 14.06.2023).

ДОДАТКИ

Додаток А. Фрагмент БД вправ (файлу pronouns.csv)

Перша колонка - індекс цільового слова у реченні, друга – токенізоване речення, третя – саме цільове слово, четверта – можливі варіанти дистракторів.

Повний файл за покликанням:

https://github.com/your-lithium/task_generator/blob/2067b7b9485113d563d61d7a5d3a28b8d3dd9b6a/pronouns.csv

word_number	sentence	pronoun	variants
0	['Моєму', 'старшому', 'брату', '(', 'є', ')', '29', 'років', '.']	Моєму	['Мій', 'Мого']
1	['У', 'мене', 'є', 'мама', ',', 'тато', ',', 'троє', 'братів', 'і', 'дві', 'сестри', '.']	мене	['я', 'мені']
0	['Я', 'шукаю', 'супермаркет', '.']	Я	['Мене', 'Мені']
1	['У', 'мене', 'також', 'є', 'бабуся', 'і', 'дідусь', '.']	мене	['я', 'мені']
1	['Тут', 'я', 'дізнаюся', 'дуже', 'багато', 'нового', 'і', 'цікавого', '.']	я	['мене', 'мені']
0	['Мені', 'потрібен', 'йогурт', 'і', 'вівсянка', '.']	Мені	['Я', 'Мене']
0	['Це', 'твій', 'одногрупник', '?']	Це	['Цього', 'Цьому', 'Цім']
1	['Це', 'твій', 'одногрупник', '?']	твій	['твого', 'твоєму']
0	['Я', 'навчаюся', 'на', 'другому', '(', '2', ')', 'курсі', '.']	Я	['Мене', 'Мені']
0	['Я', 'люблю', 'свою', сім'ю , '.']	Я	['Мене', 'Мені']
5	['Щоб', 'донести', 'продукти', 'додому', ',', 'тобі', 'потрібен', 'пакет', '.']	тобі	['ти', 'тебе']
1	['Як', 'тебе', 'звати', '?']	тебе	['ти', 'тобі']
0	['Вона', 'пенсіонерка', '(', 'на', 'пенсії', ')', '.']	Вона	['Її', 'Їй', 'Ній']
0	['Вона', 'складається', 'з', '5', 'чоловік', ':', 'мама', ',', 'тато', ',', 'я', ',', 'брат', 'і', 'сестра', '.']	Вона	['Її', 'Їй', 'Ній']
10	['Вона', 'складається', 'з', '5', 'чоловік', ':', 'мама', ',', 'тато', ',', 'я', ',', 'брат', 'і', 'сестра', '.']	я	['мене', 'мені']

Додаток Б. Фрагмент файлу test3.txt

Повний файл подано за покликанням:
https://github.com/your-lithium/task_generator/blob/2067b7b9485113d563d61d7a5d3a28b8d3dd9b6a/test3.txt

У мене є мама, тато, сестра і брат.

У мене є мама, тато, сестра і брат. Також є бабуся і дідусь. У *моєї сестри/мого брата* є діти, тому в мене є племінник і племінниця.

У мене є мама, тато, сестра і брат. Також є бабуся і дідусь з маминого боку. У *моєї сестри/мого брата* є діти, тому в мене є племінник і племінниця. Я заручена, тому скоро ми з моїм нареченим можемо створити власну сім'ю. Також у нас є домашній улюбленець — *кіт/собака*, якого ми також вважаємо родиною.

У мене (є) велика сім'я. Я маю маму і тата. Я маю брата і сестру. Я маю старшого брата. Я маю молодшу сестру. У мене є бабуся і дідусь. Я люблю свою сім'ю.

У мене не (є) велика сім'я. Моя сім'я — це мама, тато і я. Мою маму звати Олена. Мого тата звати Іван. У мене немає братів і сестер. Я люблю своїх батьків. Я також маю бабусю. Її звати Олена. Їй 70 років. Вона пенсіонерка (на пенсії).

Моя бабуся живе за містом. Вона (бабуся) живе у приватному будинку. Я часто приїжджаю до неї (бабусі). Вона завжди пригощає мене солодощами. Я дуже її люблю.

Моя сім'я живе у Києві. Вона складається з 5 людей: мама, тато, я, брат і сестра.

Мою сестру звати Ірина, їй 20 років. Ірина старша від мене. Вона (Ірина) вчиться в університеті. Вона (Ірина) живе у Харкові. Мій брат Андрій живе у Києві з мамою і татом. Йому (Андрію) 5 років. Він (Андрій) учиться в школі. Він ходить у перший клас. Минулого року Андрій ходив у дитячий садок.

Додаток В. Текстовий опис алгоритму функцій програми

1. Функція textworker:

1. Отримати текст (функція filename_input):
    1.1. Дізнатися від користувача назву файлу з текстом або повний шлях до нього;
    1.2. Спробувати відкрити файл з текстом:
        1.2.1. Прочитати файл у текстову змінну.
    1.3. У випадку, якщо такого файлу не існує, попросити спробувати ввести назву ще раз.
2. Поділити текст на речення (токенізувати):
    2.1. Власне токенізувати;
    2.2. Видалити повтори серед речень:
        2.2.1. Перетворити список на набір (тип даних, що не дозволяє повторень);
        2.2.2. Перетворити набір обратно на список.
3. Проаналізувати кожне речення (застосування функції textworker):
    3.1. Поділити речення на слова (токенізувати);
    3.2. Проаналізувати кожне слово:
        3.2.1. Визначити, чи є слово займенником для рівня A1:
            3.2.1.1. Так: утворити форми називного, родового, давального, знахідного та місцевого відмінків;
            3.2.1.2. Порівняти їх між собою та оригінальним словом, утворити список з оригінальних словоформ;
            3.2.1.3. Повернути список: індекс слова в реченні; речення, поділене на слова; коректе слово; список варіантів відповіді.
    3.3. Приєднати усі виокремлені з речення списки до спільного списку;
    3.4. Повернути список зі списками даних для кожної вправи, що утворилася з речення, обробленого програмою.

2. Функція db_worker:

1. Отримати текст файлу (функція db_chooser):
   1.1. Дізнатися від користувача назву файлу БД або повний шлях до нього;
   1.2. Якщо користувач нічого не вводить:
      1.2.1. Присвоїти змінній значення назви стандартного файлу з БД, що надається з програмою.
   1.3. У випадку, якщо такого файлу не існує, попросити спробувати ввести назву ще раз.
2. Спробувати прочитати файл, обраний користувачем:
   2.1. Якщо файл не пустий:
      2.1.1. Прочитати дані файлу в dataframe;
      2.1.2. Перевірити, чи збігаються нові дані з вже наявними:
         2.1.2.1. Якщо такого рядку ще нема у БД:
            2.1.2.1.1. Зберегти рядок.
      2.1.3. Створити новий dataframeіз збережених даних;
      2.1.4. Приєднати новий dataframeдо БД;
      2.1.5. Повернути об’єднані старі та нові дані для укладання вправ.
   2.2. Якшо файл пустий:
      2.2.1. Створити новий dataframeз наданих даних;
      2.2.2. Приєднати новий dataframeдо БД;
      2.2.3. Повернути нові дані для укладання вправ.

3. Функція task_grading_executor:

1. Отримати дані для формування вправ (функція data_choicemaker):
   1.1. Дізнатися в користувача, як він хоче працювати з програмою (функція choice_input);
   1.2. Якщо користувач обрав роботу із вже наявними текстами з БД:
      1.2.1. Прочитати дані з БД.
   1.3. Якщо користувач обрав роботу із вже наявними в БД текстами та додання своїх текстів:
      1.3.1. Запустити функцію dbworker з даними від textworker.

1.4. Якщо користувач обрав роботу із своїм новим текстом:
    1.4.1. Запустити функцію textworker.
1.5. У випадку, якщо користувач надав некоректну відповідь, попросити відповісти ще раз.
1.6. Повернути дані для формування вправ.
2. Дізнатися, скільки вправ користувач хоче отримати (функція quantity_choicemaker):
    2.1. Запитати в користувача, скільки вправ він хоче отримати;
    2.2. Якщо користувач ввів число, більше за наявну кількість вправ, сказати, що будуть надані усі вправи;
    2.3. У випадку, якщо користувач надав не ціле число, а щось інше, запитати ще раз;
    2.4. Повернути обрану користувачем кількість.
3. З усіх вправ випадково обрати визначену користувачем кількість перемішаних вправ;
4. Дізнатися, як саме користувач хоче отримувати свої результати (функція grading_choicemaker):
    4.1. Якщо користувач надав неприпустиму відповідь, запитати ще раз.
5. Якщо користувач обрав опцію з правильним варіантом одразу після відповіді та без оцінювання:
    5.1. Для кожної з випадково обраних вправ:
        5.1.1. Дати користувачу вправу для виконання (функція task_maker):
            5.1.1.1. Створити речення, у якому прибрано цільове слово (склеїти усі слова, замінивши те, чий номер було збережено, на прочерк);
            5.1.1.2. Утворити варіанти відповіді:
                5.1.1.2.1. Обрати два випадкових дистрактори;
                5.1.1.2.2. Приклеїти до дистракторів правильний варіант відповіді;
                5.1.1.2.3. Перемішати варіанти;

5.1.1.2.4. Визначити, якій літері (А/Б/В) відповідає правильний варіант.

5.1.1.3. Запитати в користувача питання;

5.1.1.4. Якщо користувач ввів відповідь у неприпустимому форматі (не А/Б/В/а/б/в), запитати ще раз;

5.1.1.5. Повернути правильне слово, відповідь користувача (літера), літеру правильної відповіді та питання.

5.1.2. Надрукувати правильну відповідь.

6. Якщо користувач обрав опцію з правильним варіантом одразу після відповіді та з оцінюванням наприкінці:

6.1. Для кожної з випадково обраних вправ:

6.1.1. Дати користувачу вправу для виконання (функція task_maker) (див. п. 5.1.1.);

6.1.2. Надрукувати правильну відповідь;

6.1.3. Якщо відповідь користувача та літера правильної відповіді співпадають – збільшити на 1 змінну з кількістю правильних відповідей.

6.2. Розрахувати процент правильних відповідей;

6.3. Надрукувати кількість правильних відповідей порівняно з кількістю питань та процент правильних відповідей.

7. Якщо користувач обрав опцію з неправильними відповідями та оцінюванням наприкінці:

7.1. Для кожної з випадково обраних вправ:

7.1.1. Дати користувачу вправу для виконання (функція task_maker) (див. п. 5.1.1.);

7.1.2. Якщо відповідь користувача та літера правильної відповіді співпадають – збільшити на 1 змінну з кількістю правильних відповідей;

7.1.3. Якщо відповідь користувача та літера правильної відповіді не співпадають – зберегти в змінну словника неправильних відповідей речення та правильну відповідь (слово).

7.2. Надрукувати словник неправильних відповідей.

7.3. Розрахувати процент правильних відповідей;

7.4. Надрукувати кількість правильних відповідей порівняно з кількістю питань та процент правильних відповідей.

8. Зупинити програму.

Додаток Г. Приклади помилок baseline програми

1. Помилки токенізації речень модулем NLTK

Нероз'єднані речення	Кількість речень	Причина
Моя сім'я — це мама, тато і я. Мою маму звати Олена.	2	слово з однієї графеми я
Це мама, тато та я. Ми живемо в Україні, у місті Київ.	2	слово з однієї графеми я
— Добрий день, я шукаю сорочку розміру S. — Я можу запропонувати чотири (4) сорочки.	2	слово з однієї графеми S
Мій розмір (є) L. Ви маєте мій розмір?	2	слово з однієї графеми L

2. Помилки токенізації речень модулем tokenize_uk

Нероз'єднані речення	Кількість речень	Причина
Як звати твоїх батьків? Ким працює твій тато? Ким працює твоя мама? Скільки років твоєму батькові? У тебе є дідусь та бабуся? У тебе є брат чи сестра? Скільки років твоєму братові? Де вчиться//навчається твоя сестра? У тебе є домашні улюбленці? Чим займається твій брат?	10	речення закінчується на незвичний знак – ?
— Ця дівчинка дуже схожа на тебе! — Дякую, це (є) моя молодша сестра.	2	речення закінчується на незвичний знак – ! , починається з незвичного знаку – —
Підкажіть, будь ласка, де сорок перша (41) аудиторія? Як пройти до кабінету директора?	2	речення закінчується на незвичний знак – ?

Підкажіть, будь ласка, де розклад/їдальня/туалет/гардероб? Кабінет заступника директора/курилка/кафедра знаходиться на другому (2) поверсі?	2	речення закінчується на незвичний знак – ?
Яка наступна пара? В якій аудиторії вона пройде?	2	речення закінчується на незвичний знак – ?
Як пройти до аудиторії №56? Де я можу знайти Петрову Антоніну?	2	речення закінчується на незвичний знак – ?
Як мені отримати стипендію? Я маю пільги?	2	речення закінчується на незвичний знак – ?
Підкажіть, будь ласка, де тут каса? Я хочу придбати ці продукти.	2	речення закінчується на незвичний знак – ?
Хто крайній (останній) у черзі на експрес-касу? Я буду за вами.	2	речення закінчується на незвичний знак – ?
Які у вас є крупи? Мені потрібна гречка.	2	речення закінчується на незвичний знак – ?
Можна розрахуватись банківською картою? Дайте, будь ласка, пакет.	2	речення закінчується на незвичний знак – ?
Підкажіть, будь ласка, де автобусна зупинка? Там зупиняється автобус 214? Скільки коштує проїзд?	3	речення закінчується на незвичний знак – ?
Перепрошую, де зупиняється автобус до бульвару Тараса Шевченка? Як мені пройти до цієї зупинки?	2	речення закінчується на незвичний знак – ?
Вибачте, цей потяг зупиняється в Києві? Скільки коштує квиток на цей потяг? Коли приходить цей потяг? Коли відходить цей потяг?	4	речення закінчується на незвичний знак – ?
Це автобусна зупинка? Тут зупиняється автобус 331? Куди їде цей автобус? Скільки коштує проїзд на автобусі?	4	речення закінчується на незвичний знак – ?
На якому автобусі я можу туди дістатися? Скільки коштує проїзд?	2	речення закінчується на незвичний знак – ?
Перепрошую, як дістатися до метро? Яка	3	речення закінчується на

станція метро проходить поруч? Скільки коштує один проїзд в метро?		незвичний знак – ?
Підкажіть, будь ласка, де (є) примірочна? Яка з них вільна?	2	речення закінчується на незвичний знак – ?
Покажіть, будь ласка, усі червоні футболки у розмірі М? Я б хотіла приміряти всі.	2	речення закінчується на незвичний знак – ?
Ви продаєте зимові рукавиці та шарфи? Мені потрібен комплект червоного кольору.	2	речення закінчується на незвичний знак – ?
Мій розмір (є) S, а не XS. Принесіть, будь ласка сукню мого розміру.	2	речення закінчується незвичним словом – XS
Зараз я на станції Теремки? Мені потрібно доїхати у центр.	2	речення закінчується на незвичний знак – ?
Не підкажете, будь ласка, як мені дістатися до ТЦ OceanPlaza? Куди мені потім повернути? Чим я можу туди дістатися? А де це?	4	речення закінчується на незвичний знак – ?
Як дістатися до аеропорту? Покажіть, будь ласка, аеропорт на карті.	2	речення закінчується на незвичний знак – ?
Перепрошую, це станція Вокзальна? Як проїхати на Либідську?	2	речення закінчується на незвичний знак – ?
Де (є) найближчий Novus? Мені потрібно купити продукти.	2	речення закінчується на незвичний знак – ?
Ви не знаєте де він знаходиться? Ця будівля має дуже багато проходів.	2	речення закінчується на незвичний знак – ?
Привіт! Яке (є) твоє ім'я? Скільки тобі років? Звідки ти (є) родом?	4	речення закінчується на незвичний знак – ?
Ти теж живеш у цьому гуртожитку? На якому поверсі ти живеш? У якій кімнаті ти живеш?	3	речення закінчується на незвичний знак – ?
— Яке (є) ваше ім'я? Моє (є) Джон.	2	речення закінчується на незвичний знак – ?
Ти теж ходиш на курси англійської в університеті? Я бачив тебе там декілька	2	речення закінчується на незвичний знак – ?

разів.		
Це твій одногрупник? Я хочу з ним познайомитися.	2	речення закінчується на незвичний знак – ?
Де ти навчаєшся? На якому ти (є) курсі? Тобі подобається там навчатися? Ким ти хочеш працювати в майбутньому?	4	речення закінчується на незвичний знак – ?
А звідки (є) ти? Давай познайомимося ближче.	2	речення закінчується на незвичний знак – ?
Звідки ти приїхав до України? Як давно ти тут живеш? Які в тебе захоплення? Ти любиш слухати музику? Давай сходимо на концерт!	5	речення закінчується на незвичний знак – ?
Давай зустрінемося ще раз? Я залишу тобі свій номер.	2	речення закінчується на незвичний знак – ?
— Дякую вам за допомогу! Як вас звати?	2	речення закінчується на незвичний знак – !

3. Приклади аналізу займенників модулем pymorphy2 (зеленим виділені варіанти, які програма має обирати)

1.1. мене

1.1.1. Parse(word='мене', tag=OpencorporaTag('NOUN,masc,inan voct'), normal_form='мен', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мене', 15, 9),))

1.1.2. Parse(word='мене', tag=OpencorporaTag('NPRO,pers,sing,anim gent'), normal_form='я', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мене', 2878, 1),))

1.1.3. Parse(word='мене', tag=OpencorporaTag('NPRO,pers,sing,anim accs'), normal_form='я', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мене', 2878, 3),))

№	початкова	грамеми

	форма	частина мови	рід	відмінок
1	мен	іменник	чоловічий	кличний
2	я	займенник	–	родовий
3	я	займенник	–	знахідний

1.2. мені

1.2.1. Parse(word='мені', tag=OpencorporaTag('NOUN,masc,inanloct'), normal_form='мен', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 15, 6),))

1.2.2. Parse(word='мені', tag=OpencorporaTag('NOUN,masc,animloct'), normal_form='мень', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 33, 7),))

1.2.3. Parse(word='мені', tag=OpencorporaTag('NOUN,femn,inandatv'), normal_form='мена', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 61, 2),))

1.2.4. Parse(word='мені', tag=OpencorporaTag('NOUN,femn,inanloct'), normal_form='мена', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 61, 5),))

1.2.5. Parse(word='мені', tag=OpencorporaTag('NPRO,pers,sing,animdatv'), normal_form='я', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 2878, 2),))

1.2.6. Parse(word='мені', tag=OpencorporaTag('NPRO,pers,sing,animloct'), normal_form='я', score=1.0, methods_stack=((DictionaryAnalyzer(), 'мені', 2878, 5),))

№	початкова форма	граеми		
		частина мови	рід	відмінок
1	мен	іменник	чоловічий	місцевий
2	мень	іменник	чоловічий	місцевий
3	мена	іменник	жіночий	давальний

4	мена	іменник	жіночий	місцевий
5	я	займенник	–	давальний
6	я	займенник	–	місцевий

1.3. її

1.3.1. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,nomn'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 0),))

1.3.2. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,gent'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 1),))

1.3.3. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,datv'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 2),))

1.3.4. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,accs'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 3),))

1.3.5. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,ablt'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 4),))

1.3.6. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdmasc,loct'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 5),))

1.3.7. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,nomn'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 6),))

1.3.8. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,gent'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 7),))

1.3.9. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,datv'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 8),))

1.3.10. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,accs'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 9),))

1.3.11. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,ablt'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 10),))

1.3.12. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdfemn,loct'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 11),))

1.3.13. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,nomn'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 12),))

1.3.14. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,gent'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 13),))

1.3.15. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,datv'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 14),))

1.3.16. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,accs'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 15),))

1.3.17. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,ablt'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 16),))

1.3.18. Parse(word='її', tag=OpencorporaTag('NPRO,Fixdneut,loct'), normal_form='її', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1956, 17),))

1.3.19. Parse(word='її', tag=OpencorporaTag('NPRO,pers,femngent'), normal_form='вона', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1360, 1),))

1.3.20. Parse(word='її', tag=OpencorporaTag('NPRO,pers,femnaccs'), normal_form='вона', score=1.0, methods_stack=((DictionaryAnalyzer(), 'її', 1360, 4),))

№	початкова форма	грамеми		
		частина мови	рід	відмінок
1	її	займенник (фікс)	чоловічий	називний
2	її	займенник (фікс)	чоловічий	родовий
3	її	займенник (фікс)	чоловічий	давальний
4	її	займенник (фікс)	чоловічий	знахідний
5	її	займенник (фікс)	чоловічий	орудний
6	її	займенник (фікс)	чоловічий	місцевий
7	її	займенник (фікс)	жіночий	називний
8	її	займенник (фікс)	жіночий	родовий
9	її	займенник (фікс)	жіночий	давальний
10	її	займенник (фікс)	жіночий	знахідний
11	її	займенник (фікс)	жіночий	орудний
12	її	займенник (фікс)	жіночий	місцевий
13	її	займенник (фікс)	середній	називний
14	її	займенник (фікс)	середній	родовий
15	її	займенник (фікс)	середній	давальний
16	її	займенник (фікс)	середній	знахідний
17	її	займенник (фікс)	середній	орудний
18	її	займенник (фікс)	середній	місцевий
19	вона	займенник	жіночий	родовий
20	вона	займенник	жіночий	знахідний

1.4. його

1.4.1. Parse(word='його', tag=OpencorporaTag('NOUN,inanfemn,voct'), normal_form='йога', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 175, 6),))

1.4.2. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,nomn'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 0),))

1.4.3. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,gent'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 1),))

1.4.4. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,datv'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 2),))

1.4.5. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,accs'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 3),))

1.4.6. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,ablt'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 4),))

1.4.7. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdmasc,loct'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 5),))

1.4.8. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,nomn'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 6),))

1.4.9. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,gent'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 7),))

1.4.10. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,datv'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 8),))

1.4.11. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,accs'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 9),))

1.4.12. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,ablt'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 10),))

1.4.13. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdfemn,loct'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 11),))

1.4.14. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,nomn'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 12),))

1.4.15. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,gent'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 13),))

1.4.16. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,datv'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 14),))

1.4.17. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,accs'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 15),))

1.4.18. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,ablt'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 16),))

1.4.19. Parse(word='його', tag=OpencorporaTag('NPRO,Fixdneut,loct'), normal_form='його', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1956, 17),))

1.4.20. Parse(word='його', tag=OpencorporaTag('NPRO,pers,mascgent'), normal_form='він', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1240, 1),))

1.4.21. Parse(word='його', tag=OpencorporaTag('NPRO,pers,mascaccs'), normal_form='він', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1240, 4),))

1.4.22. Parse(word='його', tag=OpencorporaTag('NPRO,pers,neutgent'), normal_form='воно', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1362, 1),))

1.4.23. Parse(word='його', tag=OpencorporaTag('NPRO,pers,neutaccs'), normal_form='воно', score=1.0, methods_stack=((DictionaryAnalyzer(), 'його', 1362, 4),))

№	початкова форма	граеми		
		частина мови	рід	відмінок
1	йога	займенник (фікс)	жіночий	кличний
2	його	займенник (фікс)	чоловічий	називний
3	його	займенник (фікс)	чоловічий	родовий
4	його	займенник (фікс)	чоловічий	давальний
5	його	займенник (фікс)	чоловічий	знахідний
6	його	займенник (фікс)	чоловічий	орудний
7	його	займенник (фікс)	чоловічий	місцевий
8	його	займенник (фікс)	жіночий	називний
9	його	займенник (фікс)	жіночий	родовий
10	його	займенник (фікс)	жіночий	давальний
11	його	займенник (фікс)	жіночий	знахідний
12	його	займенник (фікс)	жіночий	орудний
13	його	займенник (фікс)	жіночий	місцевий
14	його	займенник (фікс)	середній	називний
15	його	займенник (фікс)	середній	родовий
16	його	займенник (фікс)	середній	давальний
17	його	займенник (фікс)	середній	знахідний
18	його	займенник (фікс)	середній	орудний
19	його	займенник (фікс)	середній	місцевий

20	він	займенник	чоловічий	родовий
21	він	займенник	чоловічий	знахідний
22	воно	займенник	середній	родовий
23	воно	займенник	середній	знахідний

Додаток Д. Повний програмний код

1. **Повний програмний код baseline solution**

   Переглянути всі файли проєкту можна за посиланням:

   https://github.com/your-lithium/task_generator/tree/2067b7b9485113d563d61d7a5d3a28b8d3dd9b6a

2. **Повний програмний код кінцевої версії розробки**

   Переглянути файл із програмним кодом кінцевої версії розробки можна за посиланням:

   https://github.com/your-lithium/task_generator/blob/51eb7ae9bcb807cc5e1dd92dc421e24105508e2b/main.py

Додаток Е. Скріншоти таблиць створеної БД

Відповідні скріншоти таблиць token, pronoun pronoun_level подано в основному тексті на малюнках 2.6, 2.7 і 2.8 відповідно.

1. Таблиця level

rowid	level_id	name
1	1	A1
2	2	A2
3	3	B1
4	4	B2
5	5	C1
6	6	C2

2. Таблиця noun (уривок)

rowid	noun_id	nom_s	gen_s	dat_s	acc_s	ins_s	loc_s	voc_s
1	1	мама	мами	мамі	маму	мамою	мамі	мамо
2	2	тато	тата	татові	тата	татом	таті	тату
3	3	брат	брата	братові	брата	братом	браті	брате
4	4	дідусь	дідуся	дідусеві	дідуся	дідусем	дідусеві	дідусю
5	5	син	сина	синові	сина	сином	сині	сину
6	6	племінник	племінника	племінникові	племінника	племінником	племінникові	племіннику
7	7	наречений	нареченого	нареченому	нареченого	нареченим	нареченім	наречений
8	8	улюбленець	улюбленця	улюбленцеві	улюбленця	улюбленцем	улюбленцеві	улюбленцю

nom_p	gen_p	dat_p	acc_p	ins_p	loc_p	voc_p	gender
мами	мам	мамам	мам	мамами	мамах	мами	feminine
тати	татів	татам	татів	татами	татах	тати	masculine
брати	братів	братам	братів	братами	братах	брати	masculine
дідусі	дідусів	дідусям	дідусів	дідусями	дідусях	дідусі	masculine
сини	синів	синам	синів	синами	синах	сини	masculine
племінники	племінників	племінникам	племінників	племінниками	племінниках	племінники	masculine
наречені	наречених	нареченим	наречених	нареченими	наречених	наречені	masculine
улюбленці	улюбленців	улюбленцям	улюбленців	улюбленцями	улюбленцях	улюбленці	masculine

3. Таблиця noun_level

rowid	level_id	nom_s	gen_s	dat_s	acc_s	ins_s	loc_s	voc_s	nom_p	gen_p	dat_p	acc_p	ins_p	loc_p	voc_p
1	1	1	1	0	1	0	1	1	1	1	0	1	0	1	0
2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	6	1	1	1	1	1	1	1	1	1	1	1	1	1	1

4. Таблиця sentence (уривок)

rowid	sentence_id	text
	Click here to define a filter	
1	1	Моя сім'я складається з мене, мами, тата, сестри та бра
2	2	У мене є мама, тато, сестра та брат.
3	3	Також у мене є бабуся та дідусь.
4	4	Мій брат має сина, тому я ще маю племінника.
5	5	Я заручена.
6	6	Скоро я та мій наречений створимо свою сім'ю.
7	7	Також ми маємо домашнього улюбленця.
8	8	Ми любимо нашого кота.
9	9	Його ім'я – Барс.

5. Таблиця sentence_set (уривок)

rowid	sentence_id	set_id
	Click here to define a filter	
1	1	1
2	2	1
3	3	1
4	4	1
5	5	1
6	6	1
7	7	1
8	8	1
9	9	1

6. Таблиця set

rowid	set_id	name	description
	Click here to define a filter		
1	1	corpus	Вправи, укладені на основі корпусу текстів Еліни Півкіної для рівня

Додаток Є. Список словоформ, для яких не було знайдено відповідності серед лем

Словоформа	Речення
сім’я	Моя сім’я складається з мене, мами, тата, сестри та брата.
сестри	Моя сім’я складається з мене, мами, тата, сестри та брата.
сестра	У мене є мама, тато, сестра та брат.
бабуся	Також у мене є бабуся та дідусь.
сім’ю	Скоро я та мій наречений створимо свою сім’ю.
ім’я	Його ім’я – Барс.
сім’я	У мене є велика сім’я.
сестру	Також я маю брата та сестру.
сестра	Мої брат та сестра є молодшими за мене.
сім’ю	Я люблю свою сім’ю.
сім’я	У мене невелика сім’я.
сім’я	Моя сім’я – це мама, тато і я.
бабусю	Я також маю бабусю.
бабуся	Моя бабуся живе за містом.
Бабуся	Бабуся Ольга проживає у приватному будинку.
бабусю	Я люблю відвідувати свою бабусю.
сім’я	Моя сім’я живе у Львові.
сестру	Я маю сестру та брата.
сестру	Мою сестру звати Ірина.
сестра	Моя сестра навчається в університеті.
повітрі	Ми часто гуляємо в парку або робимо пікніки на свіжому повітрі.
бабусею	Я люблю проводити час з бабусею та дідусем.
Бабуся	Бабуся та дідусь вже на пенсії.
сім’я	У мене велика сім’я.
бабуся	У мене також є бабуся та дідусь.
сестру	Моя мама має брата та сестру.
сім’я	Зараз моя сім’я живе у Києві.
дитинство	Тут пройшло моє дитинство.
бабуся	Мої бабуся та дідусь продовжують жити у Вінниці.

ім’я	Його ім’я – Роман.
сестрою	Я навчаюсь у цьому університеті разом із молодшою сестрою Марією.
сім’я	Моя сім’я – це мама та тато.
домі	Ми живемо разом у затишному домі.
сестра	У мене також є старша сестра і молодший брат.
бабуся	Мої бабуся та дідусь – щасливі пенсіонери, які люблять проводити час з нами та розповідати цікаві історії зі свого життя.
їжу	Найбільше ми любимо робити спільні прогулянки, готувати смачну їжу та подорожувати.
сестру	Я маю рідну сестру.
бабуся	У тебе є дідусь та бабуся?
сестра	У тебе є брат чи сестра?
сестра	Де навчається твоя сестра?
бабуся	У нього є тато, мама, бабуся та сестра.
сестра	У нього є тато, мама, бабуся та сестра.
Ім’я	Ім’я бабусі – Марія Іванівна.
бабусі	Ім’я бабусі – Марія Іванівна.
Сестру	Сестру Сергія звати Марина.
ім’я	Моє ім’я – Джон.
спорт	Я люблю спорт.
ім’я	Моє ім’я – Алла.
дитинства	Це є мої хобі з дитинства.
майбутньому	Ким ти хочеш працювати у майбутньому?
спортом	Також я займаюсь спортом.
крилі	У якому крилі будівлі знаходиться їдальня?
їжею	Там я можу пообідати смачною їжею.
дворі	На задньому дворі університету знаходиться «курилка«.
людині	За виконану роботу людині платять зарплату.
кар’єрі	Досвід роботи дає тобі знання та вміння, які можна використовувати в подальшій кар’єрі.
молоком	Я люблю снідати вівсяною кашею з молоком.
англійської	Двічі на тиждень я ходжу на курси англійської.
їжу	На ринку продають їжу, напої та інші товари.

гречка	Мені потрібна гречка.
гречки	Я хочу придбати три кілограми гречки.
картоплю	Моя подруга буде картоплю фрі та бургер.
їжа	Напевно, тут дуже смачна їжа.
сметаною	Мій батько віддає перевагу варениками зі сметаною.
фаст	Ми зазвичай йдемо в місця, де готують фаст-фуд.
фуд	Ми зазвичай йдемо в місця, де готують фаст-фуд.
ім’я	Яке ваше ім’я?
свинини	Я б хотіла замовити закуску із печеними баклажанами, стейк зі свинини, овочевий салат та склянку газованої води.
їжу	Він хоче купити їжу та напої.
молоко	Йому потрібно купити молоко, йогурт, сир та вершкове масло.
крупу	Сергій додає до свого кошика яйця, гречану крупу та макарони.
джинси	Я в магазині одягу і шукаю нові джинси.
взуття	– Мені підходить це взуття.
взуття	Я ношу взуття тридцять дев’ятого розміру, допоможіть мені знайти кросівки в моєму розмірі, будь ласка.
взуття	Це взуття мені не подобається.
взуття	Я спробую знайти інше взуття.
взуття	Я шукаю нове зимове взуття на каблуку.
№	Вам потрібен автобус №333.
№	Це – тролейбус №4.
Філології	– Як мені дістатися до Інституту Філології?
№	– Вам потрібно проїхати на маршрутці №825 дві зупинки звідси.
№	– На автобусі №322 «Sky Bus«.
сумку	Тримайте сумку чи рюкзак перед собою.
філології	Я не знаю як звідси дістатися до Інституту філології.
Філології	Олена вирушає з гуртожитку до Інституту Філології.
Філології	Олена запитує перехожих, яким автобусом найкраще дістатися до Інституту Філології.
№	Вона дізнається, що їй потрібно сісти на автобус №18 або №23, які зупиняються на вулиці Шевченка.
№	Вона дізнається, що їй потрібно сісти на автобус №18 або №23, які

	зупиняються на вулиці Шевченка.
№	Олена дочекалася автобуса №18 і тепер прямує до Інституту Філології.
Філології	Олена дочекалася автобуса №18 і тепер прямує до Інституту Філології.
Філології	Олена вийшла на потрібній зупинці та пішла до Інституту Філології.
тепло	Сьогодні тепло та гарна погода.
повітрі	Узиму українці проводять багато часу на свіжому повітрі.
сім'єю	Відповідно до українських традицій, на Новий рік відбувається обмін подарунками із сім'єю та близькими.
Різдво	У деяких регіонах України на Різдво хлопці ходять із зірками, виконуючи колядки.
щедрощі	Вони просять благословення та щедрощі від господарів.
Різдво	Різдво - одне з найважливіших свят в Україні.
церкви	Люди ходять до церкви, де відбуваються різдвяні богослужіння.
сім'єю	Екскурсія – це відмінний спосіб провести час з сім'єю або друзями, насолоджуючись новими враженнями і споглядаючи красу навколишнього світу.
красу	Екскурсія – це відмінний спосіб провести час з сім'єю або друзями, насолоджуючись новими враженнями і споглядаючи красу навколишнього світу.
року	Зима - моя улюблена пора року.
роком	За традицією, ми прощаємося зі старим роком та зустрічаємо новий опівночі.
опівночі	За традицією, ми прощаємося зі старим роком та зустрічаємо новий опівночі.
Різдво	Третє свято - Різдво Христове.
сім'я	Моя сім'я цього дня ходить на святкову молитву до церкви та запрошує до себе в гості родичів.
церкви	Моя сім'я цього дня ходить на святкову молитву до церкви та запрошує до себе в гості родичів.
крові	У лікарні проводять різні обстеження для виявлення хвороб, наприклад, аналіз крові чи ультразвукове дослідження..
Лікаре	– Лікаре, які ліки мені потрібно приймати?
годині	Запишу вас на понеділок наступного тижня о 10 годині ранку.
здоров'я	Також в аптеці можна придбати вітаміни та дієтичні добавки для підтримки

	здоров’я.
контролю	В аптеці можна купити термометр, тонометр та інші прилади для контролю за здоров’ям.
здоров’ям	В аптеці можна купити термометр, тонометр та інші прилади для контролю за здоров’ям.
догляду	Крім ліків, ти можеш придбати там засоби для догляду за зубами, такі як зубні пасти та зубні щітки.
парацетамолу	– Рекомендую вам придбати препарат на основі парацетамолу.
догляду	Порекомендуйте мені засоби для догляду за шкірою, будь ласка.
догляду	– Ми маємо різні засоби для догляду за шкірою обличчя та тіла.
пил	У мене алергія на пил.
парацетамолу	Вам потрібно щось на основі парацетамолу чи ібупрофену?
ібупрофену	Вам потрібно щось на основі парацетамолу чи ібупрофену?
парацетамолу	– Я оберу препарат на основі парацетамолу.
вподоби	– Мені більше до вподоби жувальні таблетки.
парацетамол	– Доброго дня, я купив парацетамол, але не розумію, як його вживати.
здоров’я	Він занепокоєний своїм станом здоров’я.
здоров’я	Лікар також пропонує зробити певні зміни у житті Івана для поліпшення стану здоров’я.
їжу	Він починає регулярно вживати здорову їжу та робити фізичні навантаження, як рекомендував йому лікар, для збереження здоров’я та профілактики захворювань.
здоров’я	Він починає регулярно вживати здорову їжу та робити фізичні навантаження, як рекомендував йому лікар, для збереження здоров’я та профілактики захворювань.
їжу	Тут подають дуже смачну та різноманітну їжу.
Інтернетом	Я можу легко з’єднатися з Інтернетом і використовувати свої пристрої.
замок	Я відвідав Львів і побачив величний Львівський замок.
церкву	Мені подобається відвідувати церкву.
повітрям	Я також дихав свіжим повітрям і насолоджувався спокоєм природи.
Удома	Удома я маю велике вікно, через яке надходить багато світла у квартиру і відкривається прекрасний вид на місто.
вікно	Удома я маю велике вікно, через яке надходить багато світла у квартиру і відкривається прекрасний вид на місто.

року	У моїй квартирі є система опалення, яка допомагає мені підігріває приміщення в холодну пору року.
повітрі	Я маю великий сад та місце для відпочинку на свіжому повітрі.
сім’єю	Моя квартира має простору вітальню, де я люблю приймати гостей, проводити час із сім’єю та друзями.
повітрям	Моя квартира має маленьку балконна терасу, де я можу насолоджуватися свіжим повітрям і читати книжку.
вікна	Моя квартира знаходиться на останньому поверсі багатоповерхового будинку, тому я кожного дня бачу прекрасний краєвид з вікна.
року	У моєму будинку є камін, який створює затишну атмосферу і гріє мою родину в холодну пору року.
їжу	Я часто проводжу час удома, читаючи книги, готуючи їжу та відпочиваючи з родиною.
повітря	Температура повітря становить близько +25 градусів Цельсія.
Цельсія	Температура повітря становить близько +25 градусів Цельсія.
небі	На небі немає хмар, а сонце світить яскраво.
Небо	Небо буде вкрите хмарами, а температура повітря буде нижчою, ніж сьогодні.
повітря	Небо буде вкрите хмарами, а температура повітря буде нижчою, ніж сьогодні.
Видимість	Видимість була дуже низькою.
повітря	Влітку температура повітря піднімається, а люди можуть насолоджуватися теплими днями та прохолодною водою у річці або морі.
листя	Восени листя на деревах поступово змінює свій колір на жовтий, червоний та коричневий.
року	Існує чотири пори року: весна, літо, осінь та зима.
повітря	Я люблю ранкові прогулянки, коли повітря ще прохолодне, а сонце тільки зійшло та ще не встигло нагріти землю.
землю	Я люблю ранкові прогулянки, коли повітря ще прохолодне, а сонце тільки зійшло та ще не встигло нагріти землю.
року	Моя улюблена пора року – це літо.
годині	О 12 годині буде обідня перерва, під час якої я збираюся зустрітися зі своїми друзями в кафе.
небо	Сонце світило яскраво, а небо було безхмарним.

повітрям	Я прокинувся о 7:00 і вийшов на балкон, щоб насолодитися свіжим повітрям.
року	Проходячи повз квітковий магазин, я помітив, що квіти рясно розквітають у цю пору року.
Небо	Небо похмурилося, а потім почався дощ.
щастя	На щастя, дощ не тривав довго.
повітрям	Я вийшов на прогулянку і насолоджувався свіжим повітрям.
метеорологією	Під час прогулянки я зустрів свого друга, який займається метеорологією.