Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

з напряму підготовки 121 «Інженерія програмного забезпечення»

На тему: *Розробка програмного забезпечення ігрового тренажеру з вивчення англомовної комп'ютерної термінології*

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.
Студент гр. ІПЗ-20-1
_____ *Іосипов Р. К.*

Керівник кваліфікаційної роботи _____ */ Стрюк А. М. /*

Завідувач кафедри _____ */ Стрюк А. М. /*

Кривий Ріг
2024

Криворізький національний університет
Факультет: *Інформаційних технологій*
Кафедра: *Моделювання та програмного забезпечення*
Освітньо-кваліфікаційний рівень: *бакалавр*
Спеціальність: *121 "Інженерія програмного забезпечення"*

ЗАТВЕРДЖУЮ
Зав. кафедри
_____*Стрюк А. М.*
«___»_____ 2024__ р.

ЗАВДАННЯ
на кваліфікаційну роботу
студенту групи ІПЗ-20-1 Іосипову Роману Костянтиновичу

1. Тема: *Розробка програмного забезпечення ігрового тренажеру з вивчення англомовної комп'ютерної термінології* затверджено наказом по КНУ №___ від «__» лютого 2024 р.
2. Термін подання студентом закінченого проекту «10» червня 2024 р.
3. Вихідні дані по роботі: *Пояснювальна записка: 70 сторінок, 26 рисунків, 1 додаток, 30 використаних у роботі джерел*
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): .
5. Перелік графічного демонстраційного матеріалу: Приклади існуючих програм. Функціональна схема. Блок-схема основного алгоритму. Структура бази даних. Приклади роботи розробленої програми.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	*Формулювання мети та задач роботи*	*13.02.2024-20.02.2024*
2	*Аналіз інформаційних джерел*	*21.02.2024-09.03.2024*
3	*Визначення вимог до програмного забезпечення*	*10.03.2024-18.03.2024*
4	*Розробка функціональної схеми*	*19.03.2024-23.03.2024*
5	*Розробка алгоритмів*	*24.03.2024-31.03.2024*
6	*Розробка бази даних*	*01.04.2024-14.04.2024*
7	*Розробка програмного забезпечення*	*15.04.2024-13.05.2024*
8	*Тестування програмного забезпечення*	*14.05.2024-20.05.2024*
9	*Оформлення пояснювальної записки*	*21.05.2024-12.06.2024*
10	*Розробка демонстраційних матеріалів*	*13.06.2024-17.06.2024*

Дата видачі завдання: «_____»_____ 2024 р.
Студент _____ Іосипов Р.К.
Керівник роботи _____ Стрюк А. М.

РЕФЕРАТ

АНГЛІЙСЬКА МОВА, КОМП’ЮТЕРНА ТЕРМІНОЛОГІЯ, ІГРОВИЙ ТРЕНАЖЕР, ГЕЙМІФІКАЦІЯ, ПРОГРАМА, БАЗА ДАНИХ.

Пояснювальна записка: 86 с., 21 рис., 9 джерел.

В представленій кваліфікаційній роботі було спроєктовано та реалізовано ігровий тренажер, який призначений для ефективного вивчення та закріплення англомовних комп'ютерних термінів.

Метою роботи є створення програмного забезпечення ігрового тренажеру з вивчення англомовної комп'ютерної термінології.

У рамках кваліфікаційної роботи було успішно розроблено програмний продукт. Програмне забезпечення являє собою навчальну гру, що дозволяє користувачам в ігровій формі пізнавати нову лексику в сфері комп'ютерних технологій.

Розробка включала декілька основних етапів: визначення вимог до функціональності програми, проектування архітектури та інтерфейсу користувача, вибір технологічного стеку, програмування, тестування та налагодження. Ігровий тренажер містить різноманітні типи завдань, включаючи вправи на впізнавання термінів.

Для створення ігрового симулятора розробники вибрали інтегроване середовище розробки (IDE) RAD Studio.

Введення ігрового тренажера у навчальний процес може значно поліпшити зацікавленість студентів у вивченні іноземної мови та спеціалізованої термінології, підвищити мотивацію і ефективність засвоєння матеріалу завдяки інтерактивному та захоплюючому формату подачі інформації.

ABSTRACT

ENGLISH LANGUAGE, COMPUTER TERMINOLOGY, TRAINING SIMULATOR, GAMIFICATION, PROGRAM, DATABASE.

Explanatory note: 86 pages, 21 figures, 9 sources.

The presented qualification paper describes the design and implementation of a training simulator aimed at efficient learning and reinforcement of English computer terminology.

The purpose of the work is to create software for a training simulator in English computer terminology.

Within the qualification work, a software product was successfully developed. This software is an educational game that allows users to learn new vocabulary in the field of computer technology in a game format.

The development included several main stages: determining the software's functional requirements, designing the architecture and user interface, selecting a technology stack, programming, testing, and debugging. The training simulator contains various types of tasks, including exercises for recognizing terms.

For the creation of the training simulator, the developers chose the RAD Studio integrated development environment (IDE).

Introducing the training simulator into the educational process could significantly improve students' interest in learning a foreign language and specialized terminology, enhance motivation, and increase the efficiency of material assimilation thanks to an interactive and engaging format of information delivery.

ЗМІСТ

ВСТУП

Вивчення англійської мови, зокрема фахової термінології, є ключовим аспектом у професійному розвитку сучасного інженера-програміста. Це обумовлено тим, що багато програмного забезпечення, інструкцій, технічної літератури, а також спільноти розробників використовують англійську як основну мову спілкування. Тому глибоке розуміння англомовної термінології дозволяє інженерам-програмістам ефективно взаємодіяти з колегами по всьому світу, слідувати останнім тенденціям у сфері технологій та бути в курсі новітніх досягнень в області ІТ. Знання англійської допомагає також у написанні коду, оскільки більшість програмувальних мов використовують ключові слова англійською, а також полегшує створення та використання технічної документації і ресурсів. Враховуючи глобалізацію та міжнародний характер ІТ-індустрії, знання англійської мови є не просто перевагою, а необхідною умовою для досягнення успіху в даній області.

Використання гейміфікації та ігрових комп’ютерних програм може значно полегшити та зробити більш ефективним процес вивчення англомовної фахової термінології. Ігрові елементи, такі як бали, рівні, виклики та віртуальні нагороди, можуть заохочувати студентів до більш активної участі в навчальному процесі та підвищити їхню мотивацію. Особливо корисно це може бути в сферах, де необхідно засвоїти велику кількість спеціалізованої лексики, наприклад, у програмуванні. Ігрові комп’ютерні програми можуть надавати контекстуальне навчання, де терміни використовуються у сюжетно-орієнтованих сценаріях, допомагаючи студентам краще запам'ятовувати словниковий запас і розуміти його практичне застосування.

Підтвердження актуальності розробки програмного забезпечення ігрового тренажеру, спеціалізованого на вивченні англомовної комп'ютерної термінології, випливає із зростаючої потреби в освоєнні англійської мови як міжнародного засобу спілкування у сфері інформаційних технологій. Знання англомовної комп'ютерної термінології важливе не тільки для ІТ-спеціалістів,

але й для широкої аудиторії користувачів, яка використовує комп'ютерні програми та сервіси у своїй діяльності. Ігровий тренажер допоможе ефективно освоїти відповідну лексику в інтерактивному та мотивуючому форматі, таким чином сприяючи кращому розумінню фахових текстів і зміцненню комунікативних навичок у міжнародному професійному середовищі.

Тож метою нашої роботи є створення ігрового тренажеру з вивченні англомовної комп'ютерної термінології.

Для досягнення цієї мети необхідно виконати наступні завдання:

– проаналізувати досвід використання комп’ютерних програм у вивченні англомовної термінології;

– дослідити методи геймифікація вивчення іншомовних термінів;

– визначити вимоги до програмного забезпечення;

– змоделювати програмне забезпечення;

– обрати засоби реалізації;

– реалізувати та протестувати програмне забезпечення ігрового тренажеру з англомовної комп’ютерної термінології.

1 СУЧАСНІ ПІДХОДИ ДО ВИКОРИСТАННЯ КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ У ВИВЧЕНІ ІНОЗЕМНИХ МОВ

1.1 Використання комп'ютерних програм у вивчені англійської мови

Комп'ютери значно змінили процес навчання англійської мови, змінивши традиційні підходи до освіти. Ця технологічна революція змусила викладачів іноземних мов адаптуватися до нових умов та вирішувати проблеми, які не існували кілька десятків років тому, і про які навіть не могли задумуватися лінгвісти минулого. З появою інтернету, програм для віртуальних зустрічей та онлайн-ресурсів для самостійного вивчення мов, таких як мультимедійні платформи та навчальні додатки, сучасні викладачі мають зовсім інший набір інструментів та методів навчання, які вони мають освоїти, щоб бути ефективними в новому цифровому віці.

Використання ігрових елементів, або гейміфікація, у комп'ютерних програмах для вивчення англійської мови, значно підвищує ефективність та мотивацію у процесі навчання. Гейміфікація включає введення таких елементів, як ігрові рівні, набір балів, віртуальні нагороди, дошки лідерів та інші механіки, характерні для ігор, що робить навчальний процес більш захоплюючим і схожим на гру. Це дозволяє не тільки поліпшити залучення учнів до вивчення нової мови через веселий та конкурентний аспект, але й підвищує їх зацікавленість і прагнення досягти кращих результатів. Крім того, ці програми часто пропонують персоналізований досвід навчання, адаптуючи складність завдань до індивідуального темпу прогресу учня.

Гейміфікація в освіті — це використання елементів ігрового дизайну та принципів гейміфікації в освітньому процесі, з метою підвищення мотивації, залученості та активізації студентів. Підхід передбачає внесення елементів гри, таких як системи нагород, рівні складності, ігрові завдання та виклики, до контексту навчання.

Дослідження показують, що гейміфікація може сприяти підвищенню залучення студентів до навчання, оскільки ігрові елементи викликають підвищений інтерес та емоційне відгукання. Це може вести до більшої захопленості навчальним матеріалом, кращого засвоєння знань та розвитку додаткових навичок, таких як критичне мислення, комунікація та командна робота.

Введення гейміфікованих елементів може також посприяти самостійності студентів у навчанні, оскільки створює умови для самостійного встановлення цілей, пошуку ресурсів та вирішення проблемних ситуацій в контексті, наближеному до реального життя. В результаті студенти вчаться адаптуватися до нових викликів, використовувати логіку та стратегічне планування для досягнення поставлених цілей, що є важливими якостями в професійному та особистому житті.

Застосування гейміфікації в педагогіці вимагає від викладачів грунтовного розуміння як гейміфікаційних принципів, так і потреб своїх студентів. Ефективна гейміфікація має бути інтегрована з навчальними цілями та вміло адаптована до конкретного навчального середовища. Ознайомлення з найкращими практиками та постійне вдосконалення методів гейміфікації допоможе максимізувати її вплив на освіту та навчання студентів.

Гейміфікація – це процес використання елементів гейм-дизайну та принципів ігрового процесу у неігрових контекстах, як-от у сфері освіти, для стимулювання зацікавленості, взаємодії та мотивації учнів чи студентів. Вона відрізняється від таких методів, як едутейнмент (освітні ігри) або симуляційне навчання, які теж запозичують елементи з ігрової сфери, але гейміфікація цілеспрямовано використовує ігрові механіки та динаміку в освітньому процесі.

Центральним у гейміфікації є застосування комп'ютерних технологій та різноманітного інтерактивного устаткування, характерних для відеоігор, таких

як системи балів, дошки лідерів, віртуальні значки, рівні та досягнення. У процесі навчання такі інструменти допомагають створити сенс гри та конкуренції, спонукають студентів до активнішої участі та підтримують їхню увагу на високому рівні. Відповідна візуалізація успіхів та досягнень, а також використання елементів наративу і сторітелінгу можуть зацікавити і мотивувати учнів, роблячи навчальний матеріал більш зрозумілим та запам'ятовуваним.

Гейміфікація також зміцнює пізнавальну активність студентів шляхом включення елементів вибору та автономії, надаючи їм можливість впливати на свій навчальний процес та результати. Це створює ефект безпосередньої присутності та пов'язаності з матеріалом, що підвищує залученість та інтерес до навчання. Застосування гейміфікації може розглядатися як інноваційний підхід у сфері освіти, що дозволяє адаптувати традиційні методи навчання до потреб сучасного покоління студентів, зростаючих в оточенні цифрових технологій та інтерактивного контенту.

Гейміфікація, яка впроваджує елементи гри в освітній процес, виявляється надзвичайно ефективною у стимулюванні інтересу та мотивації студентів до навчання. Вона може значно покращити рівень володіння англійською мовою, так як ігрові елементи сприяють активному використанню мови в різних контекстах, що дозволяє практикувати та закріплювати лексику, граматику й вимову в природних комунікативних ситуаціях.

Крім того, гейміфікація сприяє кооперації між студентами, оскільки багато ігор передбачають командну взаємодію і співпрацю для досягнення спільної мети, що може позитивно вплинути на соціальне взаємодію і розвиток комунікативних навичок у студентів.

Також важливим аспектом впровадження гейміфікації є розвиток креативного мислення. Ігрові задачі часто вимагають нестандартного підходу та креативного рішення проблем, що спонукає студентів виходити за рамки

традиційного мислення і розвивати здатність до інноваційного мислення та креативності.

Однак, саме включення ігрових елементів у процес вивчення англійської мови ще не робить його повноцінною гейміфікацією. Для того щоб навчання перетворилося на захоплююче і ефективне заняття за принципом гейміфікації, викладач мусить індивідуалізувати ігровий контент з урахуванням його чотирьох важливих компонентів:

1. Взаємодія: Це включає широкий спектр технік, які стимулюють соціальну взаємодію між учасниками навчання. Приклади можуть включати роботу в парах чи групах, ділові ігри, рольові ігри, де кожен учасник може взаємодіяти, обмінюючись знаннями та досвідом з іншими.

2. Динаміка: Це стосується впровадження мотивуючого наративу або сюжету, який залучає користувачів і є стимулом для їхньої активності та залученості у навчальний процес у реальному часі. Наприклад, створення історії про подорож, де учасники мають вирішувати завдання на знання англійської для просування вперед.

3. Механіка: Це включення елементів, характерних для ігрового процесу, таких як системи віртуальних винагород, статусів, здобуття балів, просування по рівнях та досягнення результатів, які вимірюють прогрес учнів. Ці механізми допомагають підтримувати інтерес і надають зворотний зв'язок про успішність в навчанні.

4. Естетика: Це створення приємної візуальної та аудіальної атмосфери, яка сприяє естетичному і емоційному залученню учасників. За допомогою графічних елементів, анімації, звуків і музики формується оригінальний світ, в якому відбувається навчання, що дозволяє глибше занурити учнів у процес.

Поєднуючи ці чотири складники в процесі викладання англійської мови, викладач може розробляти надзвичайно мотивуючі і ефективні заняття, які залучають учнів і сприяють більш глибокому розумінню та закріпленню знань.

Використання ігрових методів у сфері освіти стає все більш популярним, адже воно забезпечує діяльнісний підхід до навчання, що сприяє глибшому

засвоєнню матеріалу та розвитку навичок критичного мислення і творчих здібностей учнів. Гейміфікація освіти - це процес впровадження елементів ігор у навчальні процеси, що спрямовані на підвищення мотивації та залученості студентів.

Основними напрямами гейміфікації є:

1. Освітні ігри - спеціально розроблені цифрові або фізичні ігри, які інтегровані у навчальний процес і мають ясно визначені навчальні цілі. Вони дозволяють учням застосовувати теоретичні знання на практиці в контексті гри, розширюючи та поглиблюючи розуміння предмету.

2. Застосування ігрових технік і методик під час звичайного навчального процесу - це може включати в себе елементи на кшталт накопичення балів, змагань, досягнень, віртуальних медалей, гейміфікованих тестів та вправ, які використовуються для підвищення зацікавленості і залученості учнів. такий підхід допомагає підтримувати мотивацію учнів, оскільки він вводить елементи змагання і візуального прогресу у набутті знань і навичок.

Враховуючи потенціал гейміфікації у сучасній освіті, багато навчальних закладів та освітні фахівці активно впроваджують такі методики у свою практику для підвищення ефективності навчання та задоволення потреб нового покоління учнів, які вирісши у цифрову епоху, мають високі вимоги до інтерактивності та динамічності навчального середовища.

В нашій роботі ми розглянемо приклади саме комп’ютерних засобів геймифікації вивчення англійської мови.

1.2 Огляд ігрових тренажерів з англійської мови

Розглянемо найпопулярніші комп'ютерні програми та мобільні застосунки, що реалізують ігрові елементи навчання англійської мови.

LingvLeo (рис. 1.1) – це мобільний застосунок, який вирізняється насамперед своєю персоналізацією. Програма навчання складається індивідуально для кожного користувача, з урахуванням його рівня володіння англійською мовою, навчальних цілей та особистих інтересів. Цей підхід дозволяє створити оптимізований та мотивуючий шлях навчання для студентів.

Рисунок 1.1 – LingvLeo

Крім того, LingvLeo пропонує багатий вибір безкоштовних навчальних матеріалів. Застосунок використовує сучасні методики навчання, зосереджуючись на теоретичних засадах англійської мови, що допомагає

учням поглиблено зрозуміти структуру мови та покращити свої знання. Окрім теоретичних аспектів, LingvLeo також пропонує практичні вправи, ігри та додаткові ресурси для розвитку навичок розмовної мови, слухання, читання та письма.

Англійський додаток LioDuo (рис. 1.2) поєднує візуальні та аудіальні методи навчання, пропонуючи приблизно 80 тематичних уроків з різноманітних предметів. Він надає фонетичні транскрипції як американською, так і британською вимовою, дозволяючи користувачам вибирати відповідно до їхніх уподобань або навчальних цілей. Додаток включає всебічний розділ словникового запасу, який категоризує слова залежно від їхнього значення та частоти використання, і включає складні терміни, відібраний лексичний запас, архаїчні терміни, а також неологізми. Крім того, надається статистичні дані про частоту використання кожного окремого слова в мові, що дозволяє учням розуміти практичне застосування їхнього словникового запасу в реальних контекстах.

Рисунок 1.2 – LioDuo

Lingvist (рис. 1.3) є передовим додатком для вивчення англійської мови, розробленим для того, щоб аналізувати індивідуальний підхід користувачів до навчання та адаптувати навчальний процес з урахуванням їхніх конкретних потреб. Програма пропонує велику варіативність навчальних матеріалів, охоплюючи різноманітні тематичні області, що дозволяє студентам зануритись у практику англійської мови через контекст, зацікавлений для них.

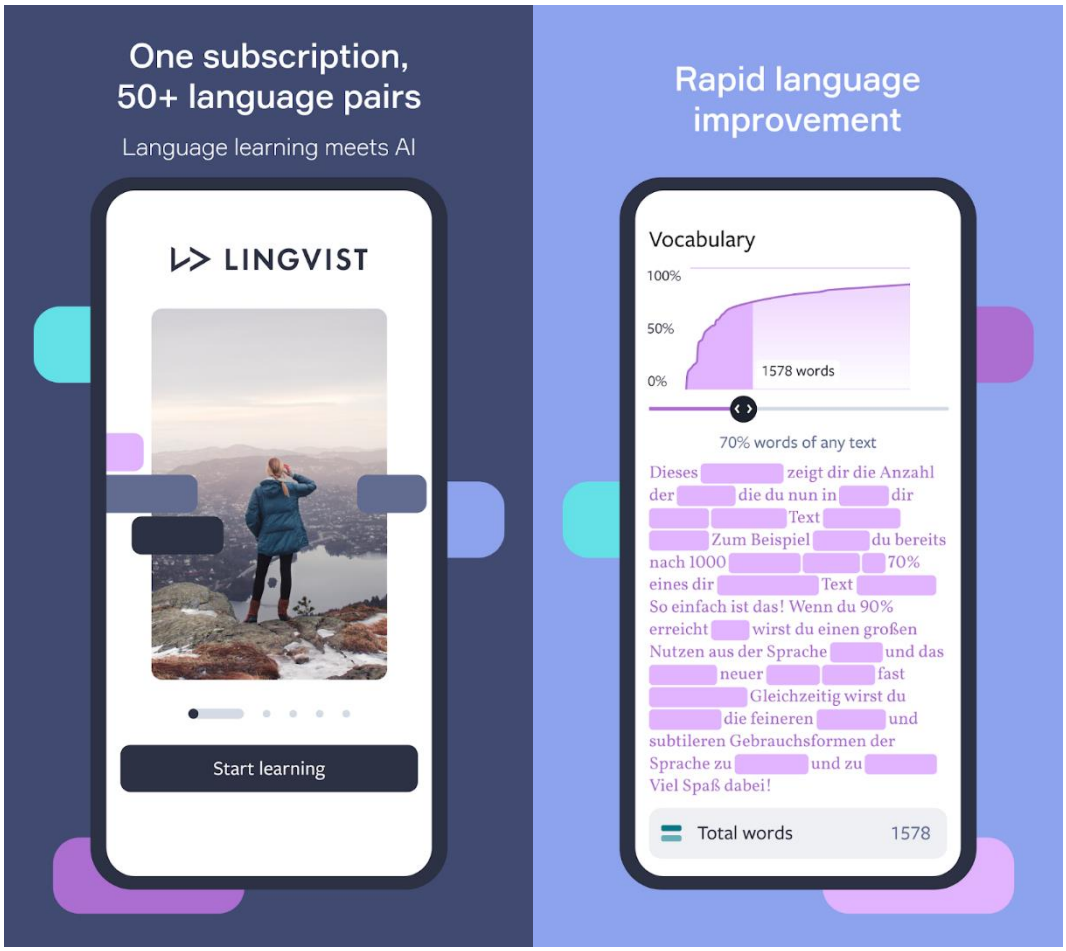

Рисунок 1.3 – Lingvist

Завдяки інтегрованому режиму розпізнавання мови, користувачі можуть покращувати своє вимову та розуміння усного мовлення, що є критичною частиною оволодіння будь-якою мовою. Також Lingvist дає можливість слідкувати за особистими досягненнями в навчанні за допомогою різних відстежувальних і аналітичних інструментів, які мотивують студентів і підтримують їх зацікавленість в процесі навчання. Ця функція дозволяє

користувачам бачити свій прогрес у реальному часі та налаштовувати свої навчальні плани відповідно до їхнього прогресу і особистих цілей.

EnglishDom (рис. 1.4) – це мобільний додаток, створений для ефективного засвоєння лексики англійської мови та для розширення словникового запасу користувачів. Він пропонує ряд вправ, які розроблені для поліпшення знань та запам'ятовування нових слів.

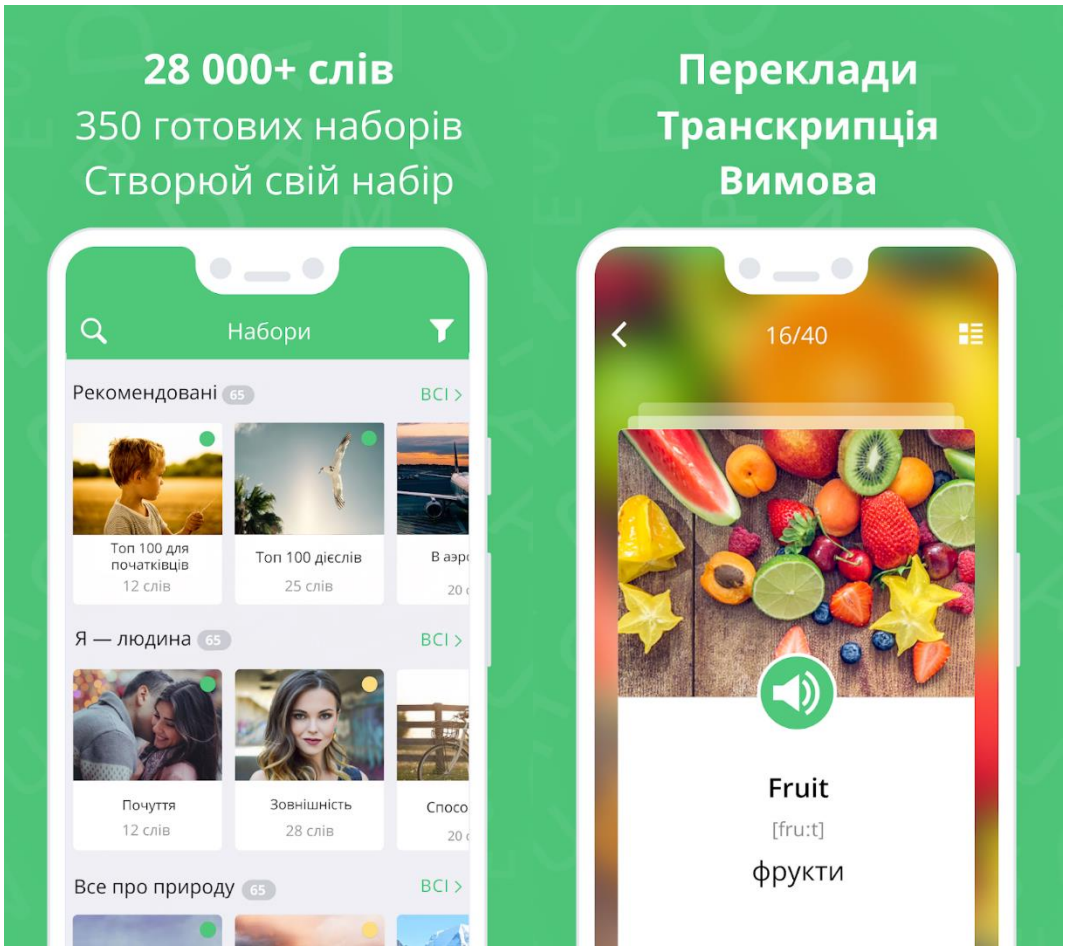

Рисунок 1.4 – EnglishDom

Користувачі мають можливість вибрати з чотирьох різних типів вправ, що включають:

1. Вибір правильного варіанта перекладу, де потрібно визначити вірний переклад запропонованого слова з декількох варіантів.

2. Аудіювання, яке дозволяє користувачам практикувати розуміння на слух, слухаючи слова, вимовлені носіями мови.

3. Складання слів з окремих літер, завдання, в якому користувачам належить правильно зібрати слово з набору літер.

4. Друкування слів, що допомагає практикувати правильне написання та орфографію слів.

EnglishDom також впроваджує графічні зображення до кожного слова, спрямовані на полегшення процесу запам'ятовування через асоціативне мислення. Ця особливість дозволяє користувачу створювати ментальні асоціації між слівцям та їх візуальним представленням.

Додатково, кожне слово у додатку супроводжується транскрипцією, перекладом та можливістю прослуховування правильної вимови слова носієм мови, що є надзвичайно корисним для вивчення правильної вимови та інтонації.

Прогрес користувача в отриманні нових знань відстежується через систему «контролю прогресу». Ця система дозволяє постійно аналізувати та оцінювати ефективність процесу навчання, надаючи зворотний зв'язок та можливість виявляти сильні та слабкі сторони студента у вивченні англійської мови. Завдяки цьому користувачам легше визначати, які аспекти мови їм потрібно покращити. EnglishDom є інструментом, який може ефективно допомогти у самостійному вивченні англійської мови, а також стати допоміжним ресурсом у класі з іншими методами навчання.

Додаток Ewa відкриває перед користувачами величезний простір для вивчення англійської, пропонуючи широкий спектр інтерактивних методик навчання. Завдяки йому, студенти можуть покращувати свої знання мови, читаючи велику колекцію книг на всілякі теми, слухаючи різноманітну улюблену музику та навіть дивлячись фільми або музичні відеокліпи. Для того щоб навчання було ефективним та цікавим, додаток оснащений інструментами, які дозволяють підібрати необхідний контент відповідно до індивідуальних переваг користувача.

Рисунок 1.5 – Ewa

Ewa забезпечує персоналізований підхід, дозволяючи кожному студенту визначити оптимальний для себе рівень складності. Користувачі можуть встановлювати свої цілі та швидкість навчання, щоб відповідати їх особистим навчальним цілям та рівням знань. Додаткові функції, такі як система повторення слів і виразів, ігрова форма навчання, а також різноманітні тестування і вправи, сприяють ефективному закріпленню матеріалу і роблять процес вивчення англійської не тільки корисним, але й захоплюючим і динамічним.

Використання різноманітних додатків для смартфонів є сучасним та ефективним засобом для самостійного вивчення англійської мови серед студентів. Завдяки цим додаткам, студенти мають можливість не лише збагатити свій лексичний запас, а й глибоко освоїти граматичні структури, регулярно практикуючи і вдосконалюючи свої знання. Крім того, інтерактивні

вправи, що спрямовані на розуміння текстів та аудіоматеріалів, сприяють поліпшенню розуміння на слух та читання, що є ключовими компонентами володіння мовою.

Такі додатки часто включають ігрові елементи, які знижують рівень стресу, пов'язаного з традиційними методами навчання, роблячи процес навчання більш привабливим і менш обтяжливим. Студенти, які користуються навчальними додатками, зазвичай знаходять у них джерело мотивації та заохочення для регулярного вдосконалення своїх мовних навичок. Це не тільки стимулює їх бажання до саморозвитку, а й відкриває нові горизонти у пізнанні англійської мови, що є особливо важливим в глобалізованому світі.

1.3 Визначення вимог ігрового тренажеру з вивчення англомовної комп'ютерної термінології

На сучасному ринку існує широкий спектр програм для вивчення англійської мови, що задовольняють різноманітні потреби та інтереси користувачів. Від інтерактивних онлайн-платформ і мобільних застосунків з іграми та квізами до комплексних освітніх курсів та віртуальних класів з носіями мови — можливості для покращення рівня англійської мови здаються нескінченними. Однак, попри вражаюче розмаїття цих ресурсів, існує ніша, яка поки що залишається незайнятою — спеціалізовані програми для вивчення комп'ютерної термінології на англійській мові.

Технологічна індустрія постійно розвивається, і знання англомовної комп’ютерної термінології стає все більш важливим для програмістів, айтішників, системних адміністраторів, та інших фахівців у цій галузі. Це не тільки полегшує доступ до найновішої літератури, обмін знаннями та спілкування з колегами з усього світу, але й є необхідною компетенцією у професійному зростанні.

Враховуючи існуючу потребу, ринок має потенціал для створення програм, які б орієнтувались спеціально на вивчення фахової лексики та дозволяли б розширити професійний словниковий запас у сфері інформаційних технологій. Такі програми могли б включати глосарії термінів, тематичні уроки, практичні завдання та симуляції реальних робочих ситуацій, що допомогли б користувачам ефективніше засвоювати та застосовувати спеціалізовану англомовну термінологію в IT галузі.

Вимоги до ігрового тренажера для вивчення англомовної комп’ютерної термінології повинні включати наступні аспекти:

1. Навчальний контент:

– Охоплення основних комп'ютерних тем, таких як апаратне забезпечення, програмування, мережі та Інтернет.

– Актуальність і точність термінології з вказівкою на джерела походження термінів.

– Різноманіття форм подання матеріалу: текст, ілюстрації, аудіо, відео.

2. Інтерактивність:

– Залучення користувача через застосування вправ, ігор, симуляцій.

– Можливість безпосередньої практики застосування термінології у контексті, наприклад, в розмовах або проектуванні.

3. Адаптивність:

– Персоналізація навчання з урахуванням рівня знань користувача.

– Включення системи налаштувань для адаптації складності завдань.

4. Оцінювання та зворотний зв'язок:

– Надання зворотного зв'язку для користувачів після виконання вправ.

– Відстеження прогресу користувача через систему рейтингів та досягнень.

5. Технічні аспекти:

– Сумісність з різноманітними пристроями та операційними системами.

– Мінімальні вимоги до апаратного забезпечення для забезпечення нормального функціонування тренажера.

– Швидка завантаженість і оптимізація для інтернет-з'єднань різної швидкості.

6. Доступність:

– Інтуїтивно зрозумілий інтерфейс для користувачів різного віку та досвіду.

– Допоміжні матеріали та керівництва для користувачів.

– При необхідності, доступність тренажера для людей з особливими потребами.

7. Інтеграція з іншими ресурсами:

– Відкриті API для інтеграції з іншими навчальними системами або елементами курсу.

– Можливість експорту даних про прогрес користувачів для аналізу або звітності.

8. Безпека та приватність:

– Захист особистих даних користувачів.

– Способи ідентифікації та аутентифікації, що забезпечують безпеку аккаунтів.

Ці вимоги формують основу для розробки ігрового тренажера, який буде ефективним інструментом для вивчення англомовної комп’ютерної термінології, привабливим для користувачів та гнучким під час адаптації до індивідуальних освітніх потреб.

Інтерфейс користувача, який вирізняється своєю інтуїтивною зрозумілістю та простотою використання, покликаний забезпечити ефективне та безпроблемне керування функціоналом. Ця система підтримує використання мультимедійних елементів, що дозволяє інтегрувати зображення, аудіофайли та відеоматеріали для більш динамічної та привабливої взаємодії з користувачем. Адаптивний дизайн гарантує, що інтерфейс буде ефективно працювати на різноманітних пристроях, включаючи настільні комп'ютери, ноутбуки, планшети та смартфони, тим самим

забезпечуючи високу якість відображення незалежно від роздільної здатності екрану або розміру пристрою.

Навчальний контент, що ми пропонуємо, розроблено з метою надати користувачам грунтовні знання в галузі сучасної комп'ютерної термінології. Він включає в себе обширний набір технічних термінів та понять, що охоплюють широкий спектр тем від основ інформаційних технологій до спеціалізованих областей як, наприклад, штучний інтелект, програмування, кібербезпека та мережеві технології.

Для кращого засвоєння матеріалу, наш контент включає інтеграцію з інтерактивними завданнями та вправами, що дозволяють користувачам активно застосовувати отримані знання на практиці. Це можуть бути онлайн-квізи, лабораторні роботи, програмування в реальному часі, тощо, завдяки чому процес навчання стає не тільки ефективнішим, а й цікавішим.

Крім того, ми передбачили можливість оновлення та розширення бази термінів у відповідності до постійного розвитку ІТ-галузі. Наші користувачі можуть бути впевнені, що вони мають доступ до найактуальнішої інформації, яка регулярно оновлюється з урахуванням найновіших технологічних тенденцій та винаходів.

Педагогічний підхід, який використовується, охоплює кілька ключових стратегій, спрямованих на залучення та підвищення мотивації користувачів. В першу чергу, це інтеграція принципів гейміфікації у навчальний процес, що передбачає введення елементів гри, таких як бали, рівні, віртуальні нагороди, що підвищує інтерес та відзначає досягнення учнів у ненав’язливій і захоплюючій формі.

Другим ключовим аспектом є адаптація навчальних програм до індивідуальних особливостей та рівня знань кожного користувача. Це включає персоналізацію завдань, регулювання темпу навчання та вибір контенту, який найбільше відповідає потребам учня, забезпечуючи ефективність навчання.

Останнім, але не менш важливим аспектом є система зворотного зв'язку та відстеження прогресу. Викладачі та навчальні платформи повинні бути обладнані засобами для моніторингу успіхів учнів, надавати їм конструктивну критику і підтримку, а також адекватно реагувати на їхні навчальні потреби. Це створює умови для самопізнання учнів та допомагає їм розвивати навички самостійного навчання.

Технічні вимоги до програмного забезпечення повинні забезпечувати:

1. Сумісність з основними операційними системами: програмне забезпечення має бути сумісним із широким спектром операційних систем, що включає Windows, macOS, Linux, а також можливо із різними дистрибутивами мобільних ОС, такими як Android та iOS. Це забезпечить доступність програми для користувачів на різноманітних пристроях.

2. Низькі системні вимоги: Програмне забезпечення повинно бути оптимізованим, аби забезпечувати працездатність навіть на старіших або менш потужних системах. Це включає оптимізацію щодо використання RAM, ЦПУ та простору на жорсткому диску, щоб зробити програму доступною для користувачів із обмеженими ресурсами.

Мовна підтримка системи включає в себе використання англійської мови як основної мови для інтерфейсу користувача та всього контенту, який він пропонує. Це означає, що всі меню, кнопки, інструкції та інші елементи інтерфейсу будуть представлені англійською мовою, а також весь доступний для користувача контент, включаючи документи, статті та інші матеріали.

Крім того, система передбачає можливість додавання додаткових мов за індивідуальним запитом користувача. Це означає, що користувачі можуть запитати інтеграцію підтримки інших мов, якщо є така потреба. В процесі розробки можуть бути додані іншомовні пакети для конкретних мов, які нададуть кінцевому користувачеві можливість переглядати та взаємодіяти з системою на обраних ним мовах, тим самим роблячи роботу з системою більше інтуїтивно зрозумілою і зручною для більшої кількості користувачів.

2 МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІГРОВОГО ТРЕНАЖЕРУ З ВИВЧЕННЯ АНГЛОМОВНОЇ КОМП'ЮТЕРНОЇ ТЕРМІНОЛОГІЇ

2.1 Загальне моделювання ігрового тренажеру з вивчення англомовної комп'ютерної термінології

Розробка загальної моделі ігрового тренажера для вивчення англомовної комп'ютерної термінології починається з побудови контекстної діаграми (рис. 2.1). Виконання цього етапу полягає у використанні нотації Гейн-Сарсона, яка дозволяє відобразити високорівневий огляд системи та взаємодії між зовнішніми акторами та самим тренажером.

В контекстній діаграмі за нотацією Гейн-Сарсона основний фокус ставиться на визначенні та представленні зовнішніх сутностей, як от користувачі (студенти та викладачі), інтерфейси взаємодії (наприклад, комп'ютери або мобільні пристрої), потрібні матеріали для навчання (дидактичні матеріали, набори термінів тощо) та будь-які інші системи, що пов’язані з тренажером (наприклад, бази даних або зовнішні освітні ресурси).

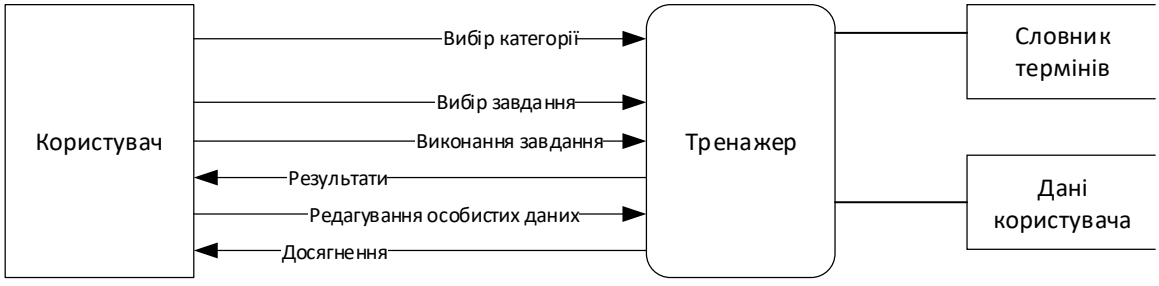

Рисунок 2.1 – Контекстна діаграма

Цю діаграму зазвичай зображують у вигляді ядро системи – в даному випадку, ігрового тренажера, де основні процеси та спосіб взаємодії з акторами ілюстровані за допомогою простих блоків та стрілок, що позначають

потоки даних. Контекстна діаграма служить керівництвом для подальшого деталізування системи, дозволяючи розробникам візуалізувати і зрозуміти, як користувачі будуть взаємодіяти з тренажером, які вхідні та вихідні дані будуть оброблятися, та як система інтегрується з іншими компонентами або зовнішніми ресурсами.

На наступному етапі нашого процесу проектування ми переходимо до деталізації системи шляхом побудови діаграми потоків даних (ДПД), яка також відома як діаграма потоку інформації (рис. 2.2). Це є важливим інструментом у процесі аналізу та моделювання систем, який дозволяє візуалізувати шляхи пересування інформації між процесами всередині системи.

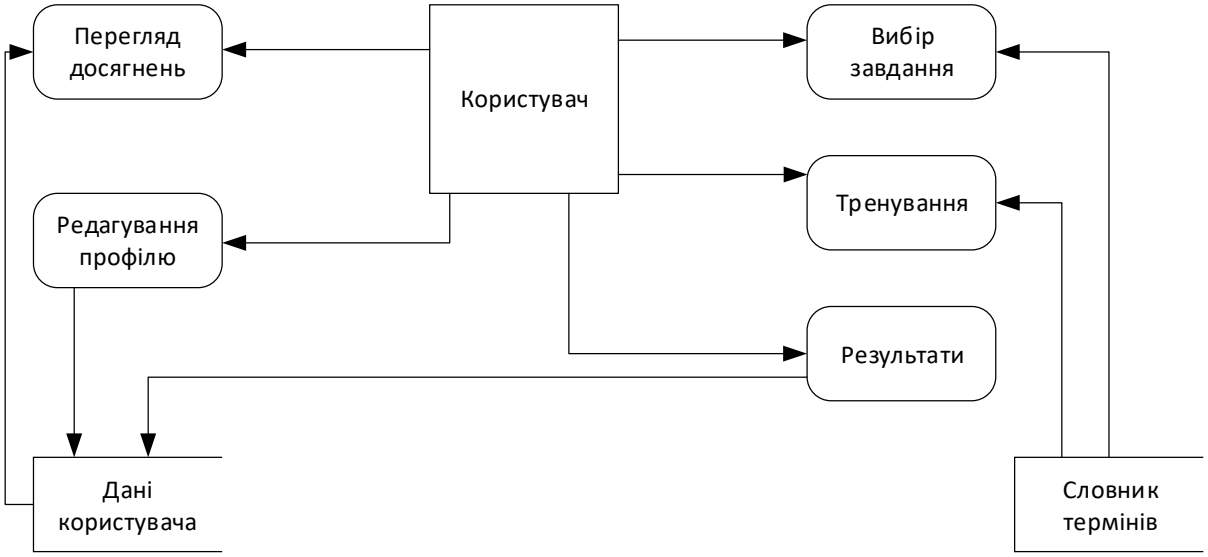

Рисунок 2.2 – Діаграма потоків даних

Для створення діаграми ми використаємо чотири основні елементи:

1. Процеси, представлені колами або прямокутниками з закругленими кутами, які визначають операції або трансформації, що відбуваються із вхідними даними для отримання результату.

2. Сховища даних, позначені прямокутниками або паралелограмами, що ілюструють місця зберігання даних у системі, де інформація може бути збережена для подальшого доступу.

3. Зовнішні суб'єкти або "термінатори", які є джерелами або призначенням інформації та можуть бути представлені особами, організаціями або іншими системами, що взаємодіють із системою.

4. Потоки даних, що представлені лініями зі стрілками, ідентифікують напрями обміну даними між процесами, сховищами даних та зовнішніми суб'єктами.

На наступному етапі проєкту передбачається створення діаграми використання для ігрового тренажеру, призначеного для вивчення англійської комп'ютерної термінології (рис. 2.3). Ця діаграма буде представляти взаємодію користувачів (акторів) із системою, визначаючи основні функціональні можливості, які ігровий тренажер надає. Акторами можуть бути студенти, викладачі та системні адміністратори. Вони використовуватимуть тренажер для таких сценаріїв, як вивчення нових термінів, проходження тестування для перевірки знань, перегляду статистики прогресу в навчанні, а також для налаштування та управління ігровим процесом та змістом. Діаграма буде містити варіанти використання, які представлятимуть основні можливості тренажеру, і лінії зв'язок, які показуватимуть, як актори взаємодіють з цими можливостями. Це забезпечить ясне візуальне представлення функціонування системи і допоможе команді програмістів розуміти, як різні частини ігрового тренажеру повинні взаємодіяти між собою.

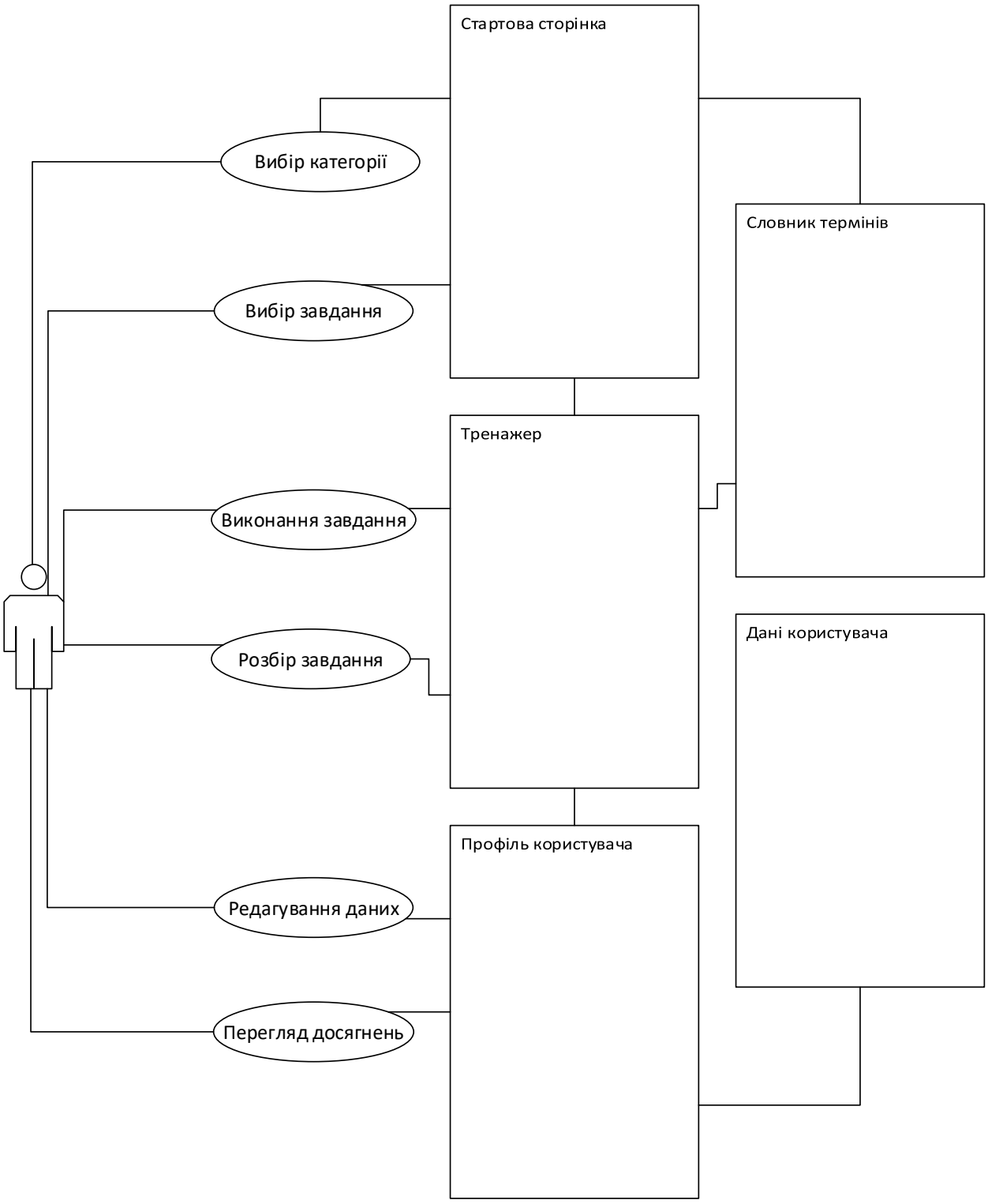

Рисунок 2.3 – Діаграма використання

2.2 Проєктування інтерфейсу ігрового тренажеру

Для кожного візуального елементу ігрового тренажеру з вивчення англійської комп'ютерної термінології спроєктуємо інтерфейс, представивши його схематичне зображення для розташування графічних елементів та елементів управління.

Проєкт дизайну стартової сторінки представлено на рисунку 2.4. Стартова сторінка складається з графічного тла, заголовку з назвою програми та кнопки старт.

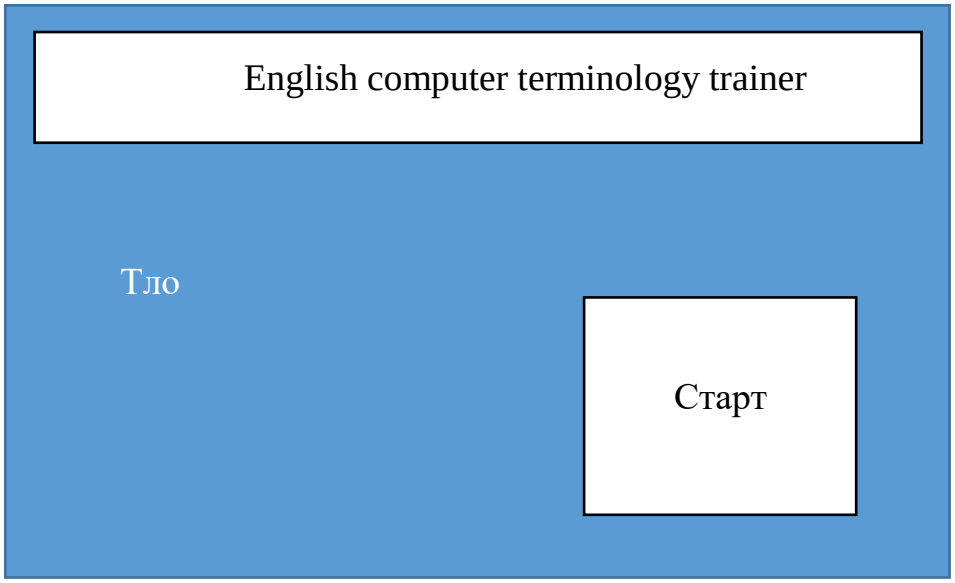

Рисунок 2.4 – Проєкт дизайну стартової сторінки

Проєкт дизайну сторінки вибору рівня представлено на рисунку 2.5. Цей дизайн передбачає відображення умовної мапи з позначками, що імітують певні локації на мапі. Кожна така позначка – рівень, що може обрати гравець.

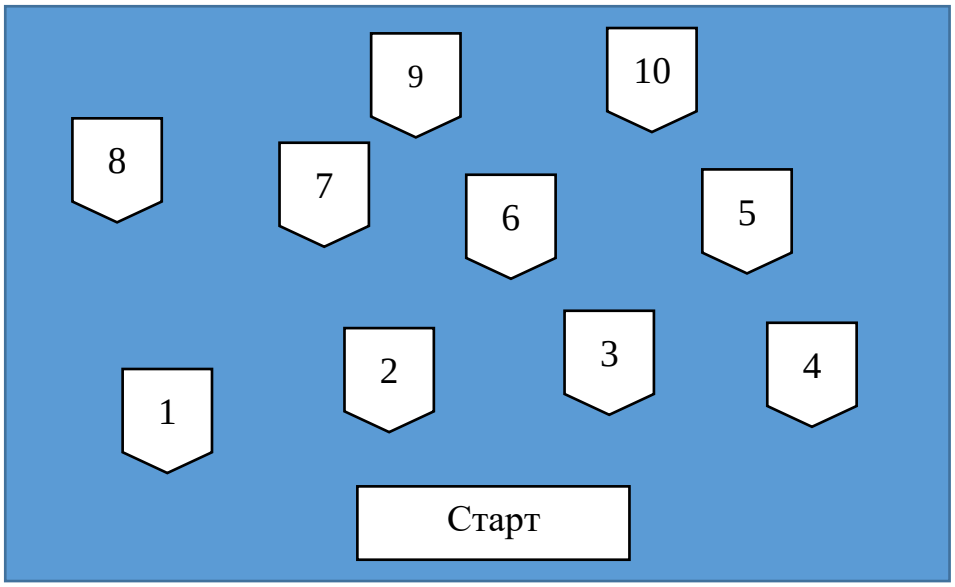

Рисунок 2.5 – Проєкт дизайну сторінки вибору рівня

Проєкт сторінки вивчення слів представлено на рисунку 2.6. На цій сторінці розміщено слова, натискання на які викликає вікно з перекладом та тлумаченням цього слова. Після вивчення слів можна перейти до гри, натиснувши кнопку Старт.

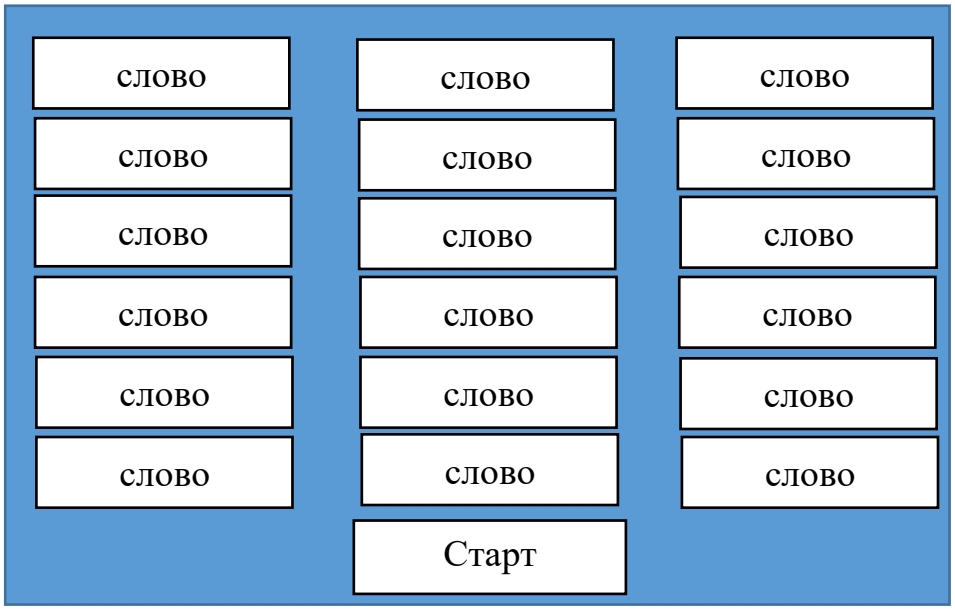

Рисунок 2.6 – Проєкт сторінки вивчення слів

Проєкт сторінки гри-вікторини представлено на рисунку 2.7.

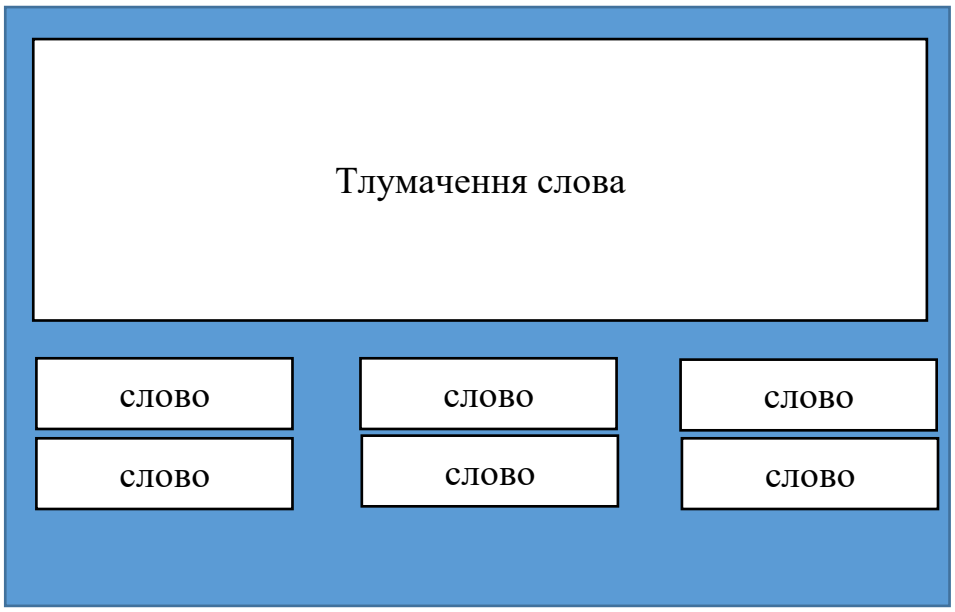

Рисунок 2.7 – Проєкт сторінки вікторини

У грі-вікторині гравцю пропонується тлумачення слова і він має обрати це слово з запропонованих варіантів.

2.3 Проєктування бази даних ігрового тренажеру

Для ефективного формування рівнів та управління англомовними термінами, їх перекладами та тлумаченнями нам знадобиться база даних. ERD-діаграма бази даних ігрового тренажеру показана на рисунку 2.8.

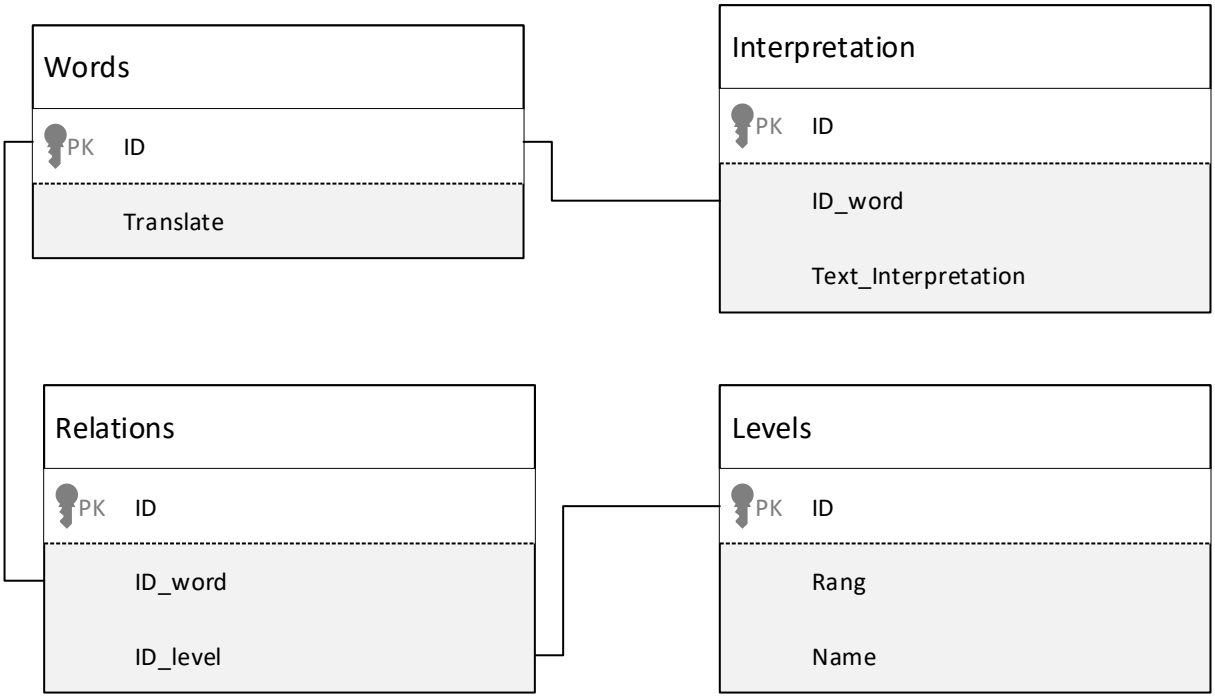

Рисунок 2.8 – ERD-діаграма бази даних ігрового тренажеру

База даних містить чотири таблиці:

– таблиця термінів (Words);

– таблиця тлумачень (Interpretation);

– таблиця рівнів (Levels);

– таблиця відношень (Relations).

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір засобів реалізації

Розробка ігрового тренажеру, який спрямований на вивчення англомовної комп'ютерної термінології, вимагає ретельного вибору інструментів розробки. У нашому випадку було вирішено скористатися інтегрованим середовищем розробки (Integrated Development Environment, IDE) RAD Studio, що надає розширені можливості для швидкої розробки крос-платформних додатків.

RAD Studio дає можливість працювати з різними мовами програмування, зокрема з Delphi та C++. У цьому проекті було обрано мову програмування C++, оскільки вона дає змогу ефективно використовувати об'єктно-орієнтовані підходи при створенні програмного забезпечення і має велику спільноту, що може допомогти з вирішенням потенційних проблем під час розробки.

C++ також є дуже потужною мовою, яка дозволяє розробникам впроваджувати складні алгоритми та ефективно керувати ресурсами системи, що є важливим для створення швидкодіючого та надійного ігрового тренажера.

Використання C++ разом з RAD Studio дозволяє команді розробників використовувати візуальні інструменти для розробки інтерфейсу користувача, а також користуватися вбудованими компонентами та бібліотеками, які полегшують інтеграцію різних функціональних бібліотек та модулів.

Цей вибір забезпечує не тільки ефективність процесу розробки, але й високу якість кінцевого продукту, що важливо для створення навчального застосунку, який може бути використаний різними групами користувачів для набуття та покращення знань англомовної комп'ютерної термінології.

3.2 Програмна реалізація

Робота ігрового тренажеру починається зі стартового екрану (рис. 3.1). Так як тренажер спрямовано на вивчення комп'ютерної термінології, ми вирішили зробити дизайн в стилі кіберпанку.

Для початку тренування потрібно натиснути кнопку «Start».

Рисунок 3.1 – Стартова сторінка ігрового тренажеру

Після натискання кнопки «Старт» відкривається вікно вибору рівня (рис. 3.2). Дизайн вибору рівня зроблено по аналогії з вибором локації на мапі футуристичного міста.

Рівні відкриваються поступово. Перейти до наступного рівня можна лише після того, як завершено попередній.

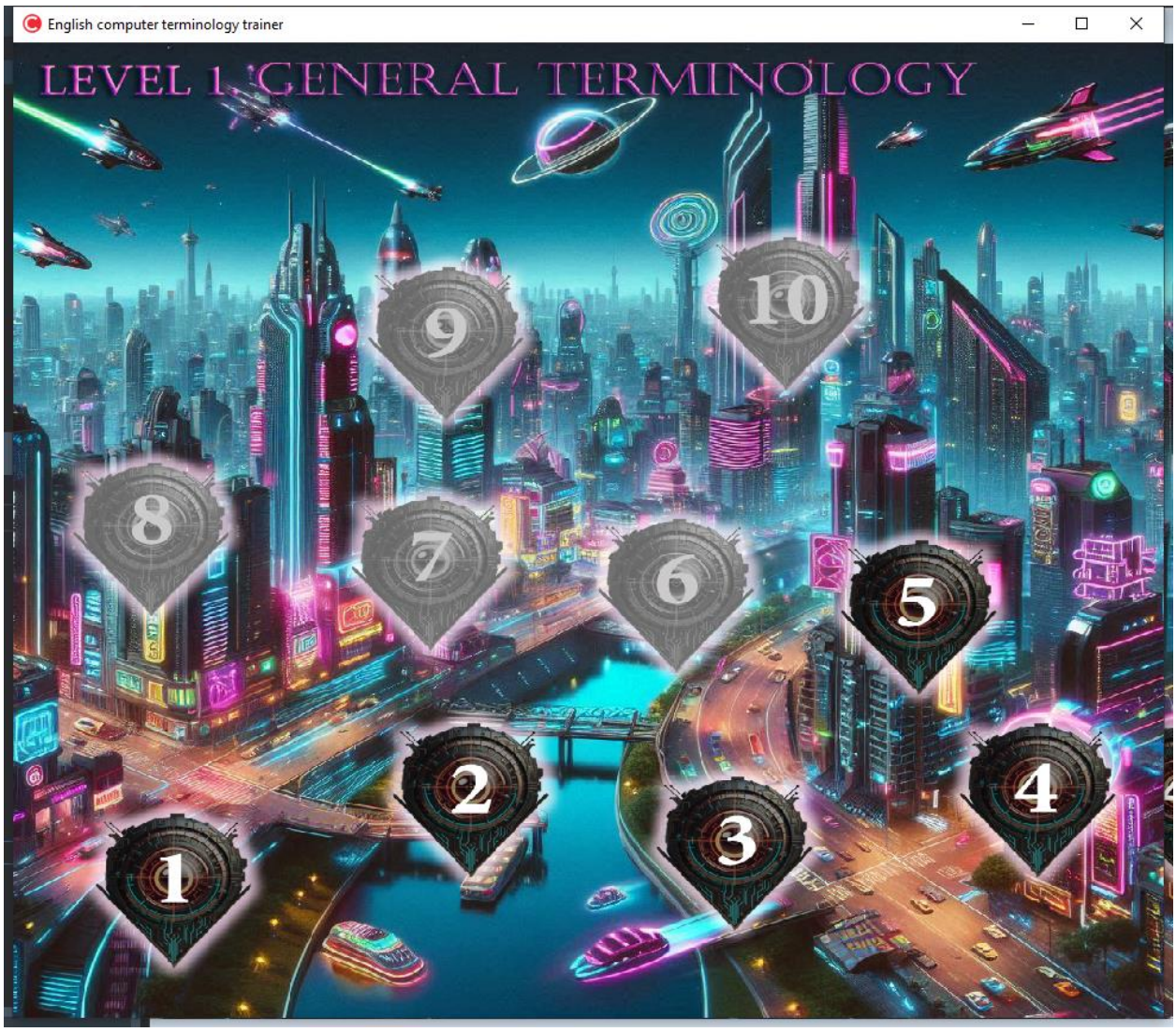

Рисунок 3.2 – Екран вибору рівня

Для вибору рівня необхідно натиснути на кнопку з номером відповідного рівня. Тоді почнеться тренування на відповідному рівні.

Тренування починається з ознайомлення з новими словами, що будуть вивчені на цьому рівні (рис 3.3).

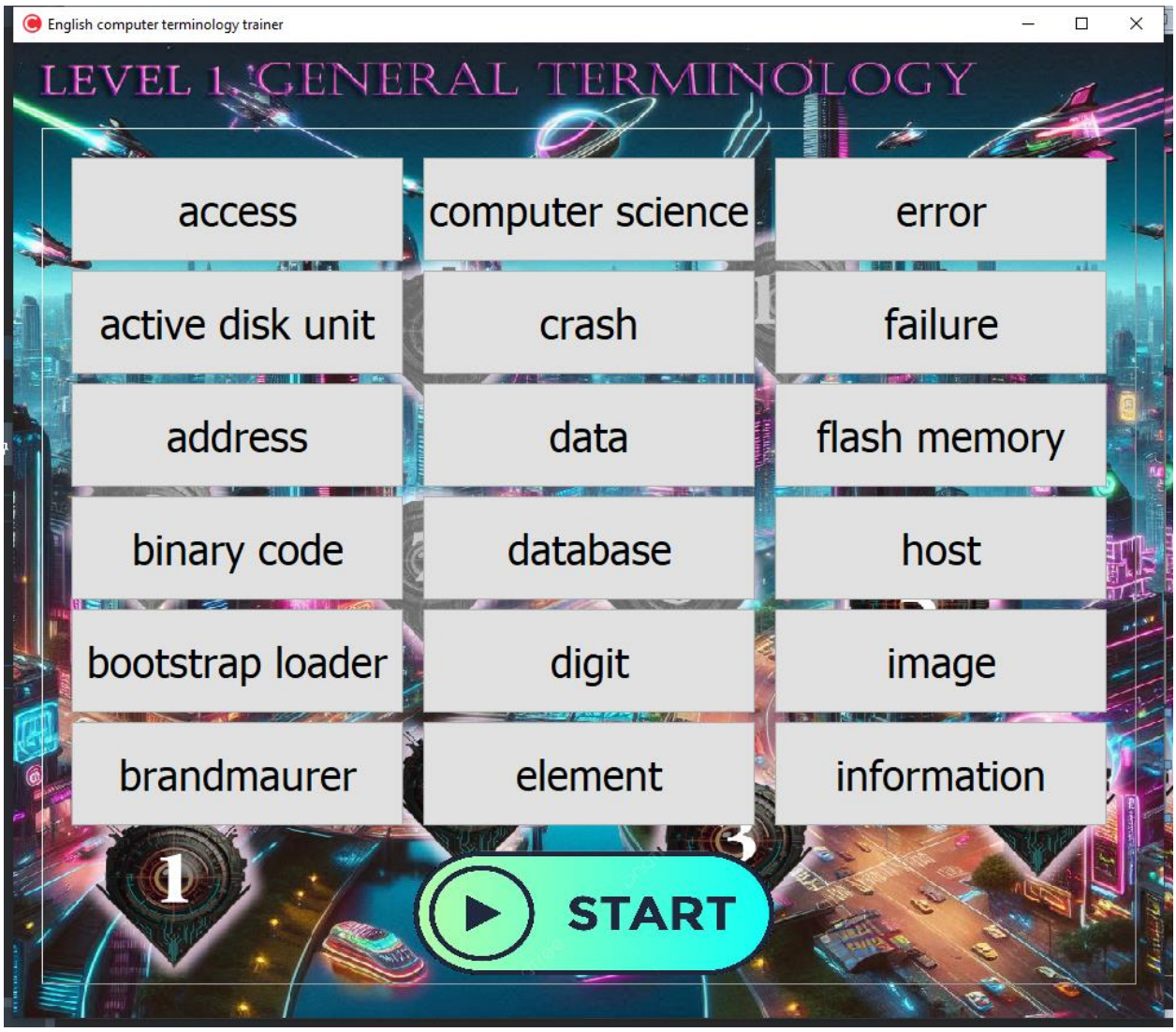

Рисунок 3.3 – Ознайомлення з новими словами

При натисканні на потрібне слово відкривається вікно з перекладом та детальним поясненням, що означає те чи інше слово в контексті інформаційних технологій (рис 3.4).

У користувача немає обмежень на час для вивчення нових слів. Після того, як студент вважає, що він засвоїв усі слова, він може перейти до перевірки знань у вигляді вікторини.

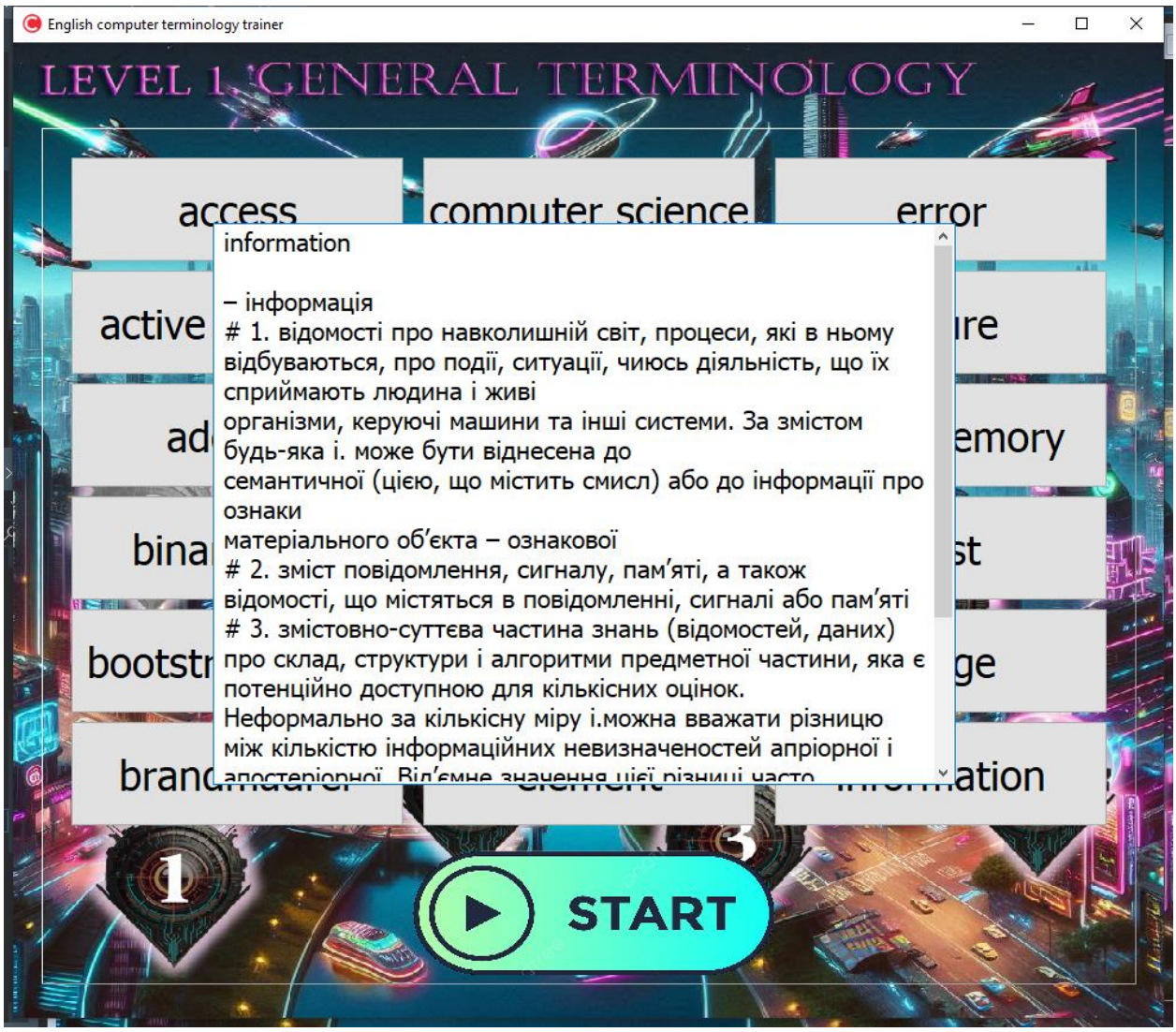

Рисунок 3.4 – Перегляд перекладу та пояснення нового слова

Для переходу до вікторини потрібно натиснути кнопку «Старт». Після цього відкриється відповідне вікно (рис. 3.5), на якому користувачу пропонується значення слова і потрібно знайти і обрати це слово з запропонованих варіантів.

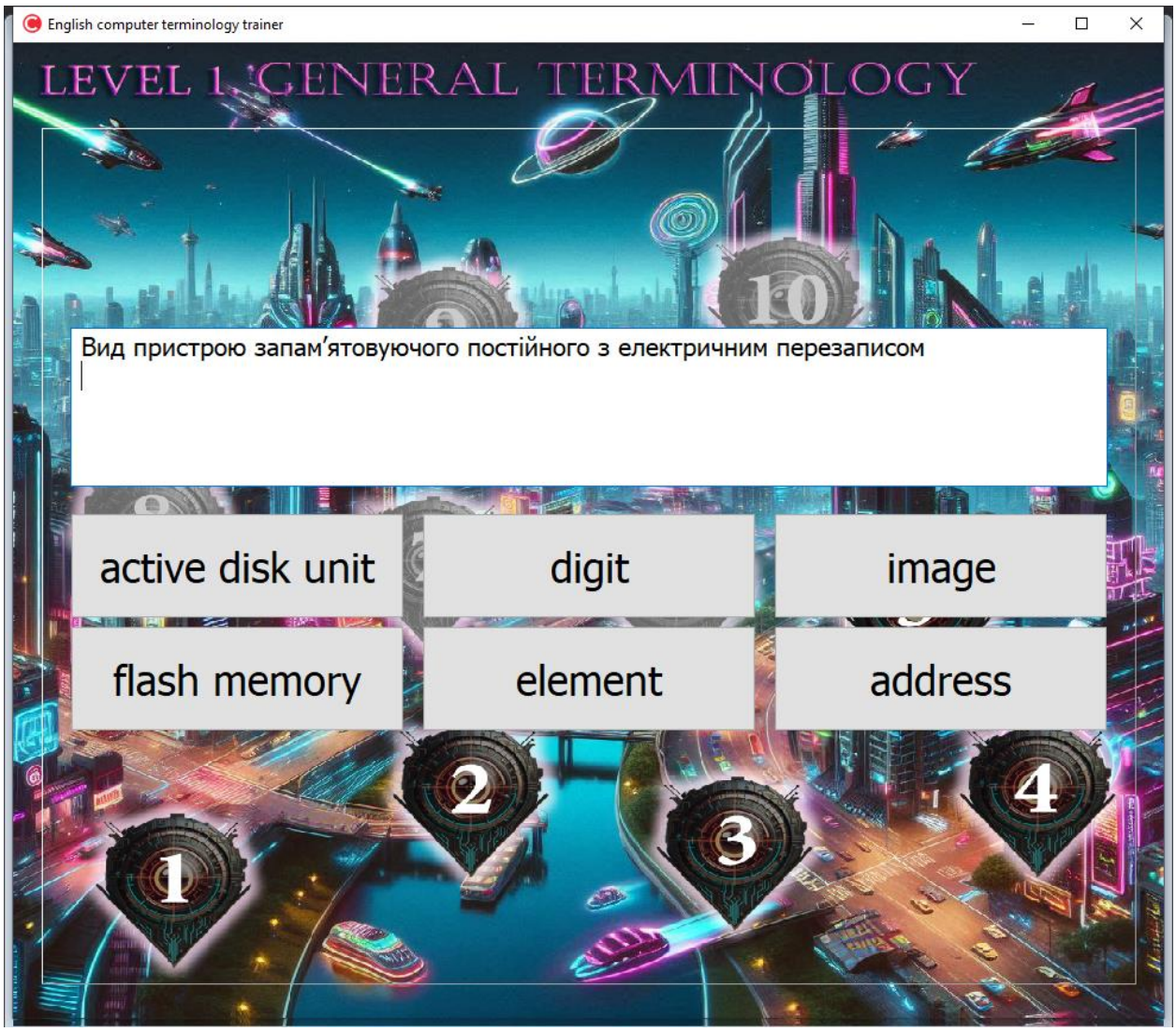

Рисунок 3.5 – Режим вікторини

Якщо слово обрано невірно, з'являється відповідне повідомлення (рис. 3.6) і відмінусовуються бали.

Якщо обрано правильне слово, відповідне повідомлення інформує про це студента (рис. 3.7) і бали додаються.

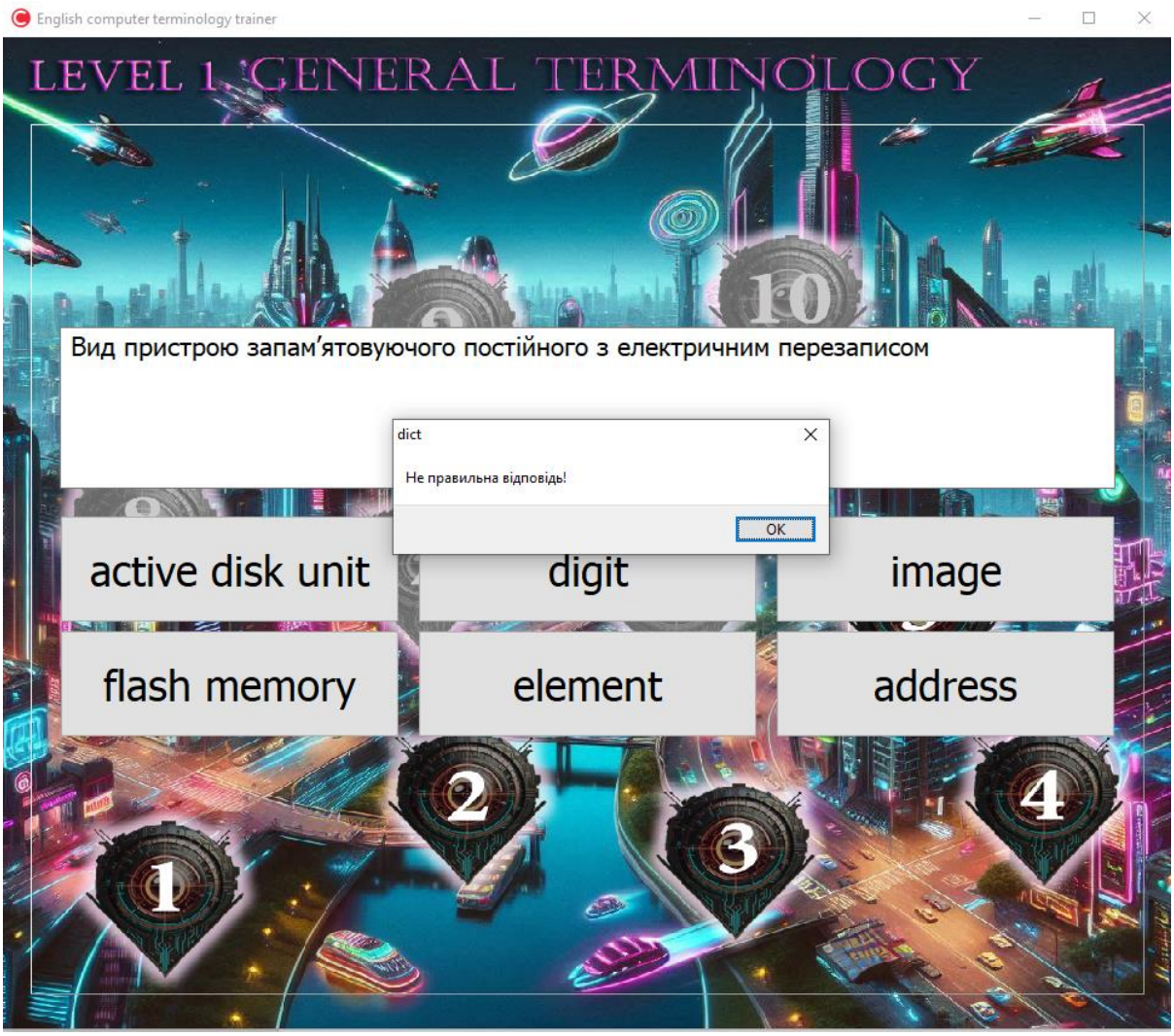

Рисунок 3.6 – Повідомлення про неправильну відповідь

Гра продовжується до тих пір, поки студент не визначить кожне слово правильно щонайменше тричі. Після цього рівень вважається пройденим.

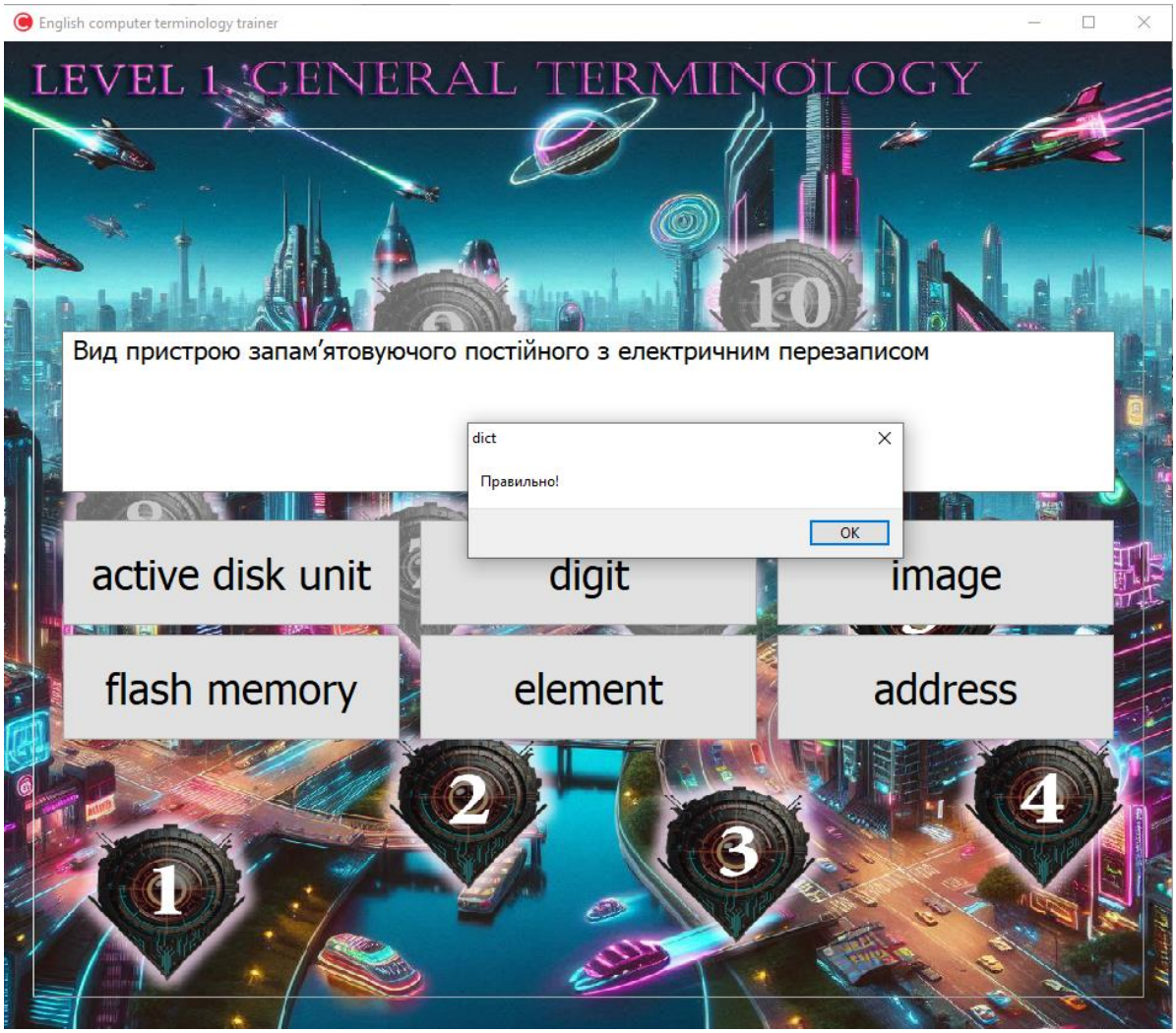

Рисунок 3.7 – Повідомлення про правильну відповідь

Користувач також може переглянути свою статистику в окремому вікні (рис. 3.8).

В цьому вікні відображається рівень гравця, кількість вивчених слів. Передбачено спеціальні нагороди, як то за проходження рівня без помилок, вивчення 1000 слів тощо.

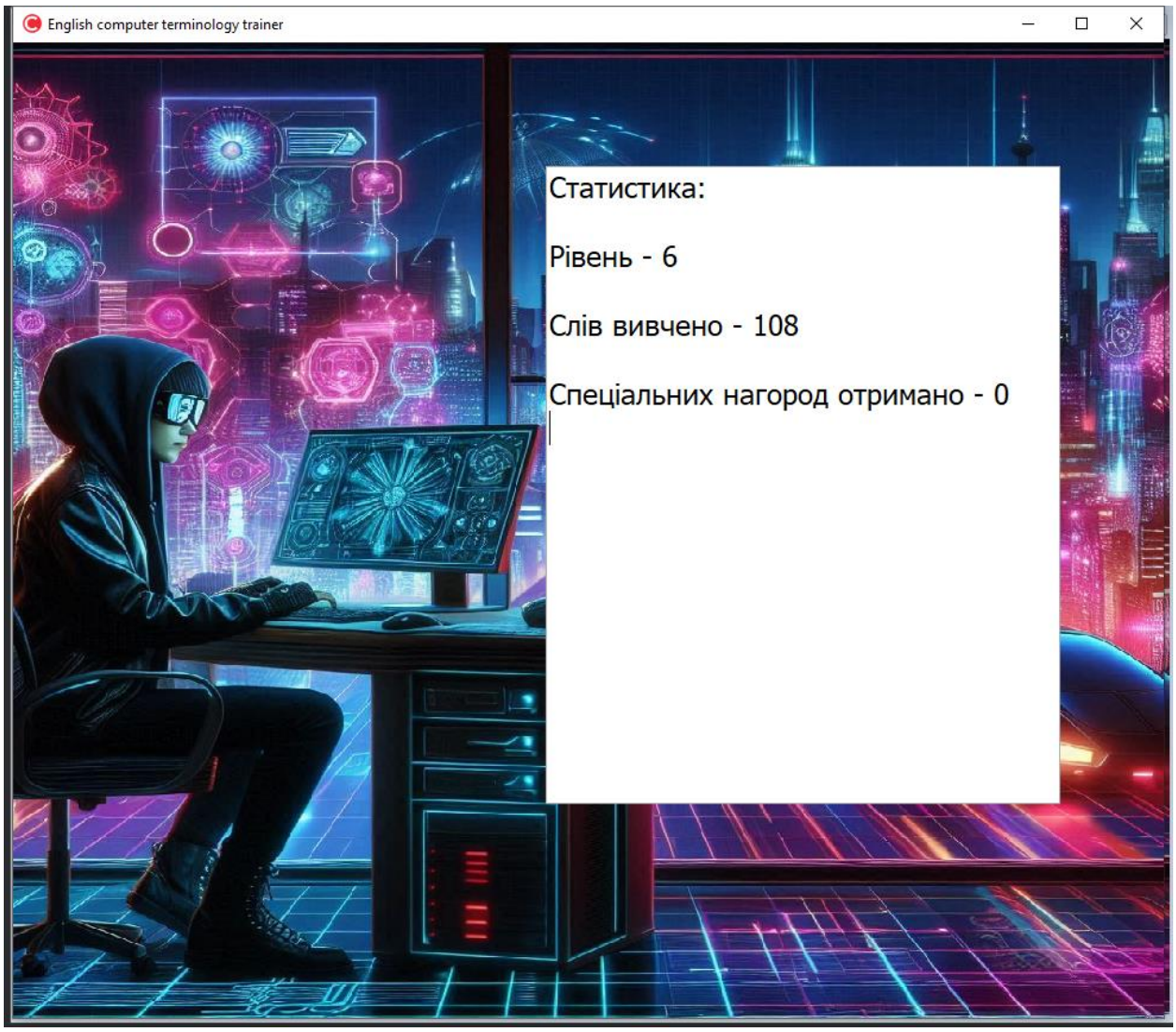

Рисунок 3.8 – Перегляд статистики

Надалі до статистики можна додавати інші дані, що будуть мотивувати гравця до активної гри.

ВИСНОВКИ

У рамках кваліфікаційної роботи було успішно розроблено програмний продукт – ігровий тренажер, який призначений для ефективного вивчення та закріплення англомовних комп'ютерних термінів. Програмне забезпечення являє собою навчальну гру, що дозволяє користувачам в ігровій формі пізнавати нову лексику, а також перевіряти та покращувати свої знання в сфері комп'ютерних технологій.

Розробка включала декілька основних етапів: визначення вимог до функціональності програми, проектування архітектури та інтерфейсу користувача, вибір технологічного стеку, програмування, тестування та налагодження. Ігровий тренажер містить різноманітні типи завдань, включаючи вправи на впізнавання термінів.

Для створення ігрового симулятора розробники вибрали інтегроване середовище розробки (IDE) RAD Studio, яке надає широкі можливості для розробки додатків для різних платформ, включаючи Windows, macOS, iOS та Android. Завдяки його потужним інструментам для візуального проектування, багатої бібліотеки компонентів та зручності використання, RAD Studio є популярним вибором серед розробників, особливо для швидкої розробки крос-платформних додатків.

Мова програмування, яку було обрано для цього проекту – C++, відома своєю високою продуктивністю та ефективністю, що є важливими аспектами при створенні ігрових тренажерів, які часто вимагають оптимальної швидкості обробки та графічного рендерингу у реальному часі. C++ також дозволяє розробникам застосовувати об'єктно-орієнтоване програмування (ООП), що сприяє створенню модульного та легко масштабованого коду, а також керуванні ресурсами через детальний контроль над використанням пам'яті і системними ресурсами.

Застосування такого союзу - потужного IDE та ефективної мови програмування, дозволило ефективно втілити необхідну функціональність,

забезпечити високий рівень продуктивності готового додатка та створити задовільний інтерфейс для кінцевих користувачів.

Введення ігрового тренажера у навчальний процес може значно поліпшити зацікавленість студентів у вивченні іноземної мови та спеціалізованої термінології, підвищити мотивацію і ефективність засвоєння матеріалу завдяки інтерактивному та захоплюючому формату подачі інформації.

ПЕРЕЛІК ПОСИЛАНЬ

1. Англо-український СЛОВНИК ТЕРМІНІВ з інформаційних технологій та кібербезпеки [Електронний ресурс]. – Режим доступу: https://ela.kpi.ua/server/api/core/bitstreams/034d3344-9f8a-4697-a679-944144bd95de/content
2. Тлумачний словник з інформатики [Електронний ресурс]. – Режим доступу: http://www.programmer.dp.ua/download/tlumachniy-slovnik-z-informatiki.pdf
3. Тлумачний словник основних понять і термінів програмування [Електронний ресурс]. – Режим доступу: https://financial.lnu.edu.ua/wp-content/uploads/2015/10/%D0%A2%D0%B5%D1%80%D0%BC%D1%96%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%87%D0%BD%D0%B8%D0%B9-%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%B8%D0%BA-%D0%B7-%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%81%D0%BA%D0%BE%D1%80%D0%BE%D1%87%D0%B5%D0%BD%D0%BE.pdf
4. Гейміфікація як засіб мотивації освітнього процесу [Електронний ресурс]. – Режим доступу: http://eprints.zu.edu.ua/39737/1/%D0%90%D0%BD%D1%82%D0%BE%D0%BD%D0%BE%D0%B2_%D0%BC%D0%BE%D0%BD%D0%BE_1_%D0%A0%D0%BE%D0%B7%D0%B4%D1%96%D0%BB.pdf
5. Програмний додаток для вивчення іноземних слів «English For You (E4u)» [Електронний ресурс]. – Режим доступу:

http://ir.lib.vntu.edu.ua/bitstream/handle/123456789/17274/2798.pdf?sequence=3

6. 15 найкращих програм для вивчення англійської мови: письмо, усне мовлення та багато іншого (2024) [Електронний ресурс]. – Режим доступу: https://preply.com/ua/blog/dodatky-dlya-vyvchennya-angliyskoyi/
7. ТОП 10 додатків для вивчення англійської [Електронний ресурс]. – Режим доступу: https://www.moyo.ua/ua/news/top_10_prilozheniy_dlya_izucheniya_angliyskogo.html
8. Топ 10 мобільних додатків для вивчення англійської [Електронний ресурс]. – Режим доступу: https://www.vivavox.com.ua/amp/english-2
9. ТОП 20 кращих прикладних програм для вивчення англійської для iOS та Андроід [Електронний ресурс]. – Режим доступу: https://enguide.ua/magazine/20-luchshih-prilozheniy-dlya-izucheniya-angliyskogo-dlya-ios-i-android

Додаток А

Текст програми

```
#include <fmx.h>
#pragma hdrstop

#include "MEIV_Unit1.h"
//---------------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm1 *Form1;
//---------------------------------------------------------------------
-------------
__fastcall TForm1::TForm1(TComponent* Owner)
     : TForm(Owner)
{
}
#if defined(__ANDROID__)
  #include <WideCharUtils.h>
#endif

#define _STR_CAST(type, arg) const_cast<type>(arg)

namespace System
{

  template <typename CH>
  int WideCharLen(const CH* src)
  {
    int len = 0;
    if (src)
    {
      while (*src++)
        ++len;
    }
    return len;
  }


  static int UTF32ToUTF16(const char32_t* src, WideChar* dst, int
len)
  {
    int count = 0;
    while ((len != 0) && *src) {
      char32_t ch = *src++;
      if (ch < 0x0000FFFF) {
        count++;
        if (dst)
          *dst++ = (ch & 0x0000FFFF);
      }
```

```
      else if (ch < 0x0010FFFF) {
        count += 2;
        if (dst) {
          ch -= 0x10000;
          *dst++ = (ch >> 10)  | 0xD800;
          *dst++ = (ch & 0x3FF)| 0xDC00;
        }
      } else {
        count++;
        if (dst)
          *dst++ = 0x0000FFFD;
      }
      if (len > 0)
        len--;
    }
    return count;
  }

#if defined(WIDECHAR_IS_CHAR16)
  #if defined(__ANDROID__)
  // NOTE: vswprintf does not work on Android/bionic currently
  //                                                          See
https://code.google.com/p/android/issues/detail?id=20567
  //             https://groups.google.com/forum/#!topic/android-
ndk/fBqGZDOIn7E
  static int vsnwprintf(UnicodeString& dst, const wchar_t* fmt,
va_list paramList)
  {
    UTF8String _fmt8(fmt);
    const char* pFmt = _fmt8.c_str();

    UTF8String _buf;
    int _size = 16;
    _buf.SetLength(_size);
    char                          *pBuf                          =
static_cast<char*>(const_cast<void*>(_buf.data()));

    int count = vsnprintf(pBuf, _size+1, pFmt, paramList);
    if (!count)
    {
      dst.SetLength(0);
      return 0;
    }

    if (count < 0)
      return count;

    if (count > _size)
    {
      _size = count;
      _buf.SetLength(_size);
      pBuf = static_cast<char*>(const_cast<void*>(_buf.data()));
      count = vsnprintf(pBuf, _size+1, pFmt, paramList);
```

```
    }
    else if (count < _size)
    {
      _buf.SetLength(count);
    }

    if (!count)
    {
      dst.SetLength(0);
    }
    else if (count > 0)
    {
      dst = _buf;
      count = dst.Length();
    }
    return count;
  }
  #else
  static int vsnwprintf(TAPtr<wchar_t>& _buf, const wchar_t* fmt,
va_list paramList)
  {
    int _size = 16;
    _buf = new wchar_t[_size];
    int count = vswprintf(&_buf[0], _size, fmt, paramList);
    while (count < 0)
    {
      _size <<= 1;
      _buf = new wchar_t[_size];
      count = vswprintf(&_buf[0], _size, fmt, paramList);
    }
    return count;
  }
  #endif
#endif

  static void dump(const UnicodeString& str, const char* f, int
line)
  {
  #ifdef WIDECHAR_IS_WCHAR  // %ls is not valid for char16_t*
    printf("%s:%d     ->     Data(%p),     Str='%ls',     len(%d),
refcount(%d)\n", f, line, str.data(), str.c_str(), str.Length(),
str.RefCount());
  #endif
  }
  #define DUMP(s) dump(s, #s, __LINE__)

  void  __cdecl UnicodeString::ThrowIfOutOfRange(int idx) const
  {
    if (idx < 1 || idx > Length())    // NOTE: UnicodeString is 1-
based !!
      throw
System::Sysutils::ERangeError(System_Sysconst_SRangeError);
  }
```

```
  UnicodeString::UnicodeString(const char* src): Data(0)
  {
#ifdef __ANDROID__
    // The Android startup sequence is such that global C++ objects
cannot use
    // Delphi code as the .ctor section is executed before
"System.main_app_entry"
    // is invoked, which means that "System.DelphiActivity" is not
initialized yet.
    // So, to allow global UnicodeString instances, such as ...
    //
    //  System::String str("I am a global variable");
    //
    //  ... we will go directly to ICU. Ultimately the Delphi Unit
Initialization
    // sequence should be updated to tolerate calls from C++ global
objects. Or the
    // C++ startup code should initialize Delphi units early enough
so that C++
    // global objects can safely access Delphi code.
    //
    InitICU();
    WideBuffer wbuff(src);
    *this = reinterpret_cast<WideChar*>(wbuff.c_str());
#else
    System::Internal::Strhlpr::UnicodeFromPChar(*this,
const_cast<char*>(src));
#endif
  }

  UnicodeString::UnicodeString(const UnicodeString& src)
  {
    // UStrAddRef
    if (this == &src) {
      if   (Data   &&   reinterpret_cast<const   StrRec*>(Data)[-
1].refCnt > 0) {

System::Syncobjs::TInterlocked::Increment((reinterpret_cast<StrR
ec*>(Data)[-1].refCnt));
      }
      return;
    }
    Data = 0;
    System::Internal::Strhlpr::UnicodeAssign(*this,
const_cast<UnicodeString&>(src));
  }

#if !defined(_DELPHI_NEXTGEN)
  UnicodeString::UnicodeString(const WideString& src): Data(0)
  {
    System::Internal::Strhlpr::UnicodeFromWide(*this,
_STR_CAST(WideString&, src));
```

```
  }
#endif

  UnicodeString::UnicodeString(const WideChar* src, int len) :
Data(0)
  {
    if (len == -1) {
      len = WideCharLen(src);
    }
    if (src && (len > 0))
    {
      SetLength(len);
      memcpy(Data, src, len*sizeof(WideChar));
    }
  }

  UnicodeString::UnicodeString(const WideChar* src) : Data(0)
  {
    if (src && *src)
    {
      int len = WideCharLen(src);
      SetLength(len);
      memcpy(Data, src, len*sizeof(WideChar));
    }
  }

#if defined(WIDECHAR_IS_WCHAR)
  UnicodeString::UnicodeString(const    char16_t*    src,    int
numChar16/*= -1*/) : Data(0)
  {
    if (src && (numChar16 != 0))
    {
      int len = (numChar16 == -1) ? WideCharLen(src) : numChar16;
      SetLength(len);
      memcpy(Data, src, len*sizeof(char16_t));
    }
  }
#endif

#if defined(WIDECHAR_IS_CHAR16)
  UnicodeString::UnicodeString(const wchar_t* src, int numWChar
/*= -1*/) : Data(0)
  {
    if (src && (numWChar != 0))
    {
      int len = UTF32ToUTF16((const char32_t*)src, 0, numWChar);
      SetLength(len);
      UTF32ToUTF16((const char32_t*)src, Data, numWChar);
    }
  }
#endif
```

```
  UnicodeString::UnicodeString(const    char32_t*    src,    int
numChar32/*= -1*/) : Data(0)
  {
    if (src && (numChar32 != 0))
    {
      int len = UTF32ToUTF16(src, 0, numChar32);
      SetLength(len);
      UTF32ToUTF16(src, Data, numChar32);
    }
  }

  UnicodeString::UnicodeString(const char* src, int len) : Data(0)
  {
#if !defined(_DELPHI_NEXTGEN)
    AnsiString str(src, len);
    System::Internal::Strhlpr::UnicodeFromAnsi(*this,
*PRawByteString(&str));
#else
  #ifdef __ANDROID__
    // See comment in UnicodeString(const char*) about Android and
global objects
    //
    InitICU();
    WideBuffer wbuff(src, len);
    *this                                                          =
UnicodeString(reinterpret_cast<WideChar*>(wbuff.c_str()),
wbuff.getLen());
  #else
    System::Internal::Strhlpr::UnicodeFromAnsi(*this,          0,
const_cast<char*>(src), len);
  #endif
#endif
  }

  UnicodeString::UnicodeString(double src) : Data(0)
  {
    *this = System::Sysutils::FloatToStr(src);
  }

  UnicodeString::~UnicodeString()
  {
    System::Internal::Strhlpr::UnicodeFree(*this);
  }

  UnicodeString&  UnicodeString::operator=(const  UnicodeString&
src)
  {
    System::Internal::Strhlpr::UnicodeAssign(*this,
const_cast<UnicodeString&>(src));
    return *this;
  }
```

```
  UnicodeString&  UnicodeString::operator+=(const  UnicodeString&
src)
  {
    System::Internal::Strhlpr::UnicodeAppend(*this,
_STR_CAST(UnicodeString&, src));
    return *this;
  }

  UnicodeString   UnicodeString::operator+(const   UnicodeString&
rhs) const
  {
    UnicodeString tmp(*this);
    System::Internal::Strhlpr::UnicodeAppend(tmp,
_STR_CAST(UnicodeString&, rhs));
    return tmp;
  }

  bool   UnicodeString::operator==(const   UnicodeString&   other)
const
  {
    return
System::Internal::Strhlpr::UnicodeEqual(_STR_CAST(UnicodeString&
, *this), _STR_CAST(UnicodeString&, other));
  }

  bool   UnicodeString::operator!=(const   UnicodeString&   other)
const
  {
    return
!System::Internal::Strhlpr::UnicodeEqual(_STR_CAST(UnicodeString
&, *this), _STR_CAST(UnicodeString&, other));
  }

  bool UnicodeString::operator<(const UnicodeString& other) const
  {
    return
System::Internal::Strhlpr::UnicodeLess(_STR_CAST(UnicodeString&,
*this), _STR_CAST(UnicodeString&, other));
  }

  bool UnicodeString::operator>(const UnicodeString& other) const
  {
    return
System::Internal::Strhlpr::UnicodeGreater(_STR_CAST(UnicodeStrin
g&, *this), _STR_CAST(UnicodeString&, other));
  }

  bool UnicodeString::operator<=(const UnicodeString& rhs) const
  {
    return !operator>(rhs);
  }

  bool UnicodeString::operator >=(const UnicodeString& rhs) const
```

```
  {
    return !operator<(rhs);
  }

  int UnicodeString::Compare(const UnicodeString& rhs) const
  {
    if (rhs.Data == Data) {
      return 0;
    }
    if (!Data || !rhs.Data) {
      return Data ? 1 : -1;
    }
#if defined(_Windows)
    return    ::CompareStringW(LOCALE_USER_DEFAULT,    0,    Data,
Length(),
                               rhs.Data, rhs.Length()) - CSTR_EQUAL;
#else
    return System::Sysutils::AnsiCompareStr(*this, rhs);
#endif
  }

  int UnicodeString::CompareIC(const UnicodeString& rhs) const
  {
    if (rhs.Data == Data) {
      return 0;
    }
    if (!Data || !rhs.Data) {
      return Data ? 1 : -1;
    }
#if defined(_Windows)
    return ::CompareStringW(LOCALE_USER_DEFAULT, NORM_IGNORECASE,
Data, Length(),
                               rhs.Data, rhs.Length()) - CSTR_EQUAL;
#else
    return System::Sysutils::AnsiCompareText(*this, rhs);
#endif
  }

  UnicodeString  UnicodeString::StringOfChar(WideChar  ch,  int
count)
  {
    UnicodeString tmp;
    tmp.SetLength(count);
    WideChar* p = tmp.Data;
    while (count--)
      *p++ = ch;
    return tmp;
  }

#if !defined(_DELPHI_NEXTGEN)
  UnicodeString UnicodeString::LoadStr(int ident)
  {
    return System::Sysutils::LoadStr(ident);
```

```
  }

#if defined(_Windows)
  UnicodeString UnicodeString::LoadStr(HINSTANCE hInstance, int
ident)
  {
    UnicodeString str;
    str.LoadString(hInstance, ident);
    return str;
  }

  UnicodeString& UnicodeString::LoadString(HINSTANCE hInstance,
int id)
  {
    HRSRC       resHdl       =       ::FindResource(hInstance,
MAKEINTRESOURCE(id/16+1), RT_STRING);
    if (resHdl)
    {
      HGLOBAL gblHdl = ::LoadResource(hInstance, resHdl);
      if (gblHdl)
      {
        WCHAR* resData = (WCHAR*)::LockResource(gblHdl);
        if (resData)
        {
          unsigned int len;
          for (int cnt = id % 16; len = *resData++, cnt--; resData
+= len)
            ;
          if (len != 0)
          {
            SetLength(len);
            wchar_t* p = Data;
            int i = len;
            while (i--)
              *p++ = *resData++;
            if (len > 0)
              Data[len] = 0;
          }
          /* Unnecessary in Win32
          ::UnlockResource(gblHdl);
          */
        }
        ::FreeResource(resHdl);
      }
    }
    return *this;
  }
#endif

  UnicodeString   UnicodeString::FmtLoadStr(int   ident,   const
TVarRec* args, int size)
  {
#if defined(_DELPHI_STRING_UNICODE)
```

```
    return System::Sysutils::FmtLoadStr(ident, args, size);
#else
    // In ANSI mode the Delphi RTL does not expose this
    // this functionality for UnicodeStrings.
    // Hence, potential data loss!
    return System::Sysutils::FmtLoadStr(ident, args, size);
#endif
  }
#endif

  UnicodeString    UnicodeString::Format(const    UnicodeString&
format, const TVarRec *args, int size)
  {
#if defined(_DELPHI_STRING_UNICODE)
    return System::Sysutils::Format(format, args, size);
#else
    // In ANSI mode the Delphi RTL does not expose this
    // this functionality for UnicodeStrings.
    // We use WideFormat instead
    return System::Sysutils::WideFormat(WideString(format), args,
size);
#endif
  }

  UnicodeString  UnicodeString::FormatFloat(const  UnicodeString&
format, const long double& value)
  {
#if defined(_DELPHI_STRING_UNICODE)
    return System::Sysutils::FormatFloat(format, value);
#else
    // In ANSI mode the Delphi RTL does not expose this
    // this functionality for UnicodeStrings.
    return System::Sysutils::FormatFloat(format, value);
#endif
  }


#undef vprintf
#undef printf
#undef sprintf

  int   __cdecl   UnicodeString::vprintf(const   wchar_t*   format,
va_list paramList)
  {
#if !defined(WIDECHAR_IS_CHAR16)
    int size = vsnwprintf(NULL, 0, format, paramList);
    SetLength(size);
    return size ? vsnwprintf(Data, size + 1, format, paramList) :
0;
#else
  #if defined(__ANDROID__)
     return vsnwprintf(*this, format, paramList);
  #else
```

```
    TAPtr<wchar_t> _buf;
    int size = vsnwprintf(_buf, format, paramList);
    if (size >= 0)
    {
      UnicodeString temp(&_buf[0], size);
      temp.swap(*this);
    }
    return size;
  #endif
#endif
  }

  int __cdecl UnicodeString::cat_vprintf(const wchar_t* format,
va_list paramList)
  {
#if !defined(WIDECHAR_IS_CHAR16)
    int size = vsnwprintf(NULL, 0, format, paramList);

    if (!size)
      return 0;

    int len = Length();
    SetLength(len + size);
    return vsnwprintf(Data + len, size + 1, format, paramList);
#else
  #if defined(__ANDROID__)
    UnicodeString temp;
    int size = vsnwprintf(temp, format, paramList);
    if (size > 0)
      this->operator+=(temp);
    return size;
  #else
    TAPtr<wchar_t> _buf;

    int size = vsnwprintf(_buf, format, paramList);
    if (size >= 0)
    {
      UnicodeString temp(&_buf[0], size);
      this->operator+=(temp);
    }
    return size;
  #endif
#endif
  }

  int __cdecl UnicodeString::printf(const wchar_t* format, ...)
  {
    int rc;
    va_list paramList;
    va_start(paramList, format);
    rc = vprintf(format, paramList);
    va_end(paramList);
    return rc;
```

```
  }

  UnicodeString&  __cdecl  UnicodeString::sprintf(const  wchar_t*
format, ...)
  {
    va_list paramList;
    va_start(paramList, format);
    vprintf(format, paramList);
    va_end(paramList);
    return *this;
  }

  int  __cdecl  UnicodeString::cat_printf(const  wchar_t*  format,
...)
  {
    int rc;
    va_list paramList;
    va_start(paramList, format);
    rc = cat_vprintf(format, paramList);
    va_end(paramList);
    return rc;
  }

  UnicodeString&     __cdecl     UnicodeString::cat_sprintf(const
wchar_t* format, ...)
  {
    va_list paramList;
    va_start(paramList, format);
    cat_vprintf(format, paramList);
    va_end(paramList);
    return *this;
  }

  UnicodeString  UnicodeString::FloatToStrF(long  double  value,
TStringFloatFormat format, int precision, int digits)
  {
    return                    System::Sysutils::FloatToStrF(value,
System::Sysutils::TFloatFormat(format), precision, digits);
  }

  UnicodeString UnicodeString::IntToHex(int value, int digits)
  {
    return System::Sysutils::IntToHex(value, digits);
  }

  UnicodeString UnicodeString::CurrToStr(Currency value)
  {
    return System::Sysutils::CurrToStr(value);
  }

  UnicodeString     UnicodeString::CurrToStrF(Currency     value,
TStringFloatFormat format, int digits)
  {
```

```
    return                    System::Sysutils::CurrToStrF(value,
System::Sysutils::TFloatFormat(format), digits);
  }

  UnicodeString& UnicodeString::Unique()
  {
    System::UniqueString(*this);
    return *this;
  }

  UnicodeString&   UnicodeString::Insert1(const   UnicodeString&
source, int index)
  {
    System::Internal::Strhlpr::UnicodeInsert(*this,
_STR_CAST(UnicodeString&, source), index);
    return *this;
  }

  UnicodeString&   UnicodeString::Insert0(const   UnicodeString&
source, int index)
  {
    return Insert1(source, index+1);
  }

  UnicodeString& UnicodeString::Delete1(int index, int count)
  {
    System::Internal::Strhlpr::UnicodeDelete(*this,        index,
count);
    return *this;
  }

  UnicodeString& UnicodeString::Delete0(int index, int count)
  {
    return Delete1(index+1, count);
  }

  UnicodeString& UnicodeString::SetLength(int newLength)
  {
    System::Internal::Strhlpr::UnicodeSetLength(*this,
newLength);
    return *this;
  }

  int UnicodeString::Pos1(const UnicodeString& subStr) const
  {
    return
System::Internal::Strhlpr::UnicodePos(_STR_CAST(UnicodeString&,
*this), _STR_CAST(UnicodeString&, subStr));
  }

  int UnicodeString::Pos0(const UnicodeString& subStr) const
  {
    return Pos1(subStr) - 1;
```

```
  }

  UnicodeString UnicodeString::LowerCase() const
  {
    return System::Sysutils::AnsiLowerCase(*this);
  }

  UnicodeString UnicodeString::UpperCase() const
  {
    return System::Sysutils::AnsiUpperCase(*this);
  }

  UnicodeString UnicodeString::Trim() const
  {
    return System::Sysutils::Trim(*this);
  }

  UnicodeString UnicodeString::TrimLeft() const
  {
    return System::Sysutils::TrimLeft(*this);
  }

  UnicodeString UnicodeString::TrimRight() const
  {
    return System::Sysutils::TrimRight(*this);
  }

  UnicodeString UnicodeString::SubString1(int index, int count)
const
  {
    // This method is intended to be compatible with Delphi's
Copy().
    // Be careful when reordering parameter validation to maintain
the
    // semantics!
    const int len = Length();
    if (index > len || count < 1)
      return UnicodeString();
    if (index < 1)
      index = 1;
    int n = len - index + 1;
    if (n > count)
      n = count;
    return UnicodeString(Data + index - 1, n);
  }

  UnicodeString UnicodeString::SubString0(int index, int count)
const
  {
    // This method is intended to be compatible with Delphi's
Copy().
    // Be careful when reordering parameter validation to maintain
the
```

```
    // semantics!
    const int len = Length();
    if (index >= len || count < 1)
      return UnicodeString();
    if (index < 0)
      index = 0;
    int n = len - index;
    if (n > count)
      n = count;
    return UnicodeString(Data + index, n);
  }

  int UnicodeString::ToInt() const
  {
    return System::Sysutils::StrToInt(*this);
  }

  int UnicodeString::ToIntDef(int defaultValue) const
  {
    return System::Sysutils::StrToIntDef(*this, defaultValue);
  }

  double UnicodeString::ToDouble() const
  {
    return System::Sysutils::StrToFloat(*this);
  }

  bool      UnicodeString::IsDelimiter1(const      UnicodeString&
delimiters, int index) const
  {
    return    System::Sysutils::IsDelimiter(delimiters,    *this,
index);
  }

  bool      UnicodeString::IsDelimiter0(const      UnicodeString&
delimiters, int index) const
  {
    return IsDelimiter1(delimiters, index+1);
  }

  bool UnicodeString::IsPathDelimiter1(int index) const
  {
    return System::Sysutils::IsPathDelimiter(*this, index);
  }

  bool UnicodeString::IsPathDelimiter0(int index) const
  {
    return IsPathDelimiter1(index+1);
  }

  int     UnicodeString::LastDelimiter1(const     UnicodeString&
delimiters) const
  {
```

```
    return System::Sysutils::LastDelimiter(delimiters, *this);
  }

  int     UnicodeString::LastDelimiter0(const     UnicodeString&
delimiters) const
  {
    return LastDelimiter1(delimiters)-1;
  }

  UnicodeString::TStringLeadCharType UnicodeString::ByteType1(int
index) const
  {
    return
UnicodeString::TStringLeadCharType(Sysutils::ByteType(*this,
index));
  }

  UnicodeString::TStringLeadCharType UnicodeString::ByteType0(int
index) const
  {
    return ByteType(index+1);
  }

  bool UnicodeString::IsLeadSurrogate1(int index) const
  {
    return ByteType(index) == ctbLeadSurrogate;
  }

  bool UnicodeString::IsLeadSurrogate0(int index) const
  {
    return IsLeadSurrogate1(index+1);
  }

  bool UnicodeString::IsTrailSurrogate1(int index) const
  {
    return ByteType(index) == ctTrailSurrogate;
  }

  bool UnicodeString::IsTrailSurrogate0(int index) const
  {
    return IsTrailSurrogate1(index+1);
  }

  DynamicArray<System::Byte> UnicodeString::BytesOf() const
  {
    return System::Sysutils::BytesOf(*this);
  }

  UnicodeString operator+(const char* lhs, const UnicodeString&
rhs)
  {
    UnicodeString tmp(lhs);
```

```
    return             System::Internal::Strhlpr::UnicodeCat(tmp,
_STR_CAST(UnicodeString&, rhs));
  }

  UnicodeString operator+(const wchar_t* lhs, const UnicodeString&
rhs)
  {
    UnicodeString tmp(lhs);
    return             System::Internal::Strhlpr::UnicodeCat(tmp,
_STR_CAST(UnicodeString&, rhs));
  }

  UnicodeString    operator+(const    char16_t*    lhs,    const
UnicodeString& rhs)
  {
    UnicodeString tmp(lhs);
    return             System::Internal::Strhlpr::UnicodeCat(tmp,
_STR_CAST(UnicodeString&, rhs));
  }

  UnicodeString    operator+(const    char32_t*    lhs,    const
UnicodeString& rhs)
  {
    UnicodeString tmp(lhs);
    return             System::Internal::Strhlpr::UnicodeCat(tmp,
_STR_CAST(UnicodeString&, rhs));
  }

  WideChar* UnicodeString::LastChar() const
  {
    return System::Sysutils::AnsiLastChar(*PUnicodeString(this));
  }

  UnicodeString& UnicodeString::swap(UnicodeString& other)
  {
    Data = reinterpret_cast<System::WideChar*>(

System::Syncobjs::TInterlocked::Exchange(reinterpret_cast<void*&
>(other.Data),

reinterpret_cast<void*>(Data)));
    return *this;
  }

//---------------------------------------------------------------
-------------


//---------------------------------------------------------------
-------------

#include <fmx.h>
#pragma hdrstop
```

```
#include "MEIV_Unit2.h"
//---------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm2 *Form2;
//---------------------------------------------------------------
-------------
__fastcall TForm2::TForm2(TComponent* Owner)
     : TForm(Owner)
{
}
TMtsDll::TMtsDll()
{
  bool init = init_com();
  if (!init) { init = init_mts(); }
  if                (!init)               {                throw
System::Sysutils::Exception(System::Sysutils::LoadStr(sInitMTSSe
rvicesError)); }
}

TMtsDll::~TMtsDll()
{
  if (Library) delete Library;
}

bool TMtsDll::init_com()
{
  TDll* comservices = new TDll("comsvcs.dll"); /* do not localize
*/
  HINSTANCE success = comservices->operator HINSTANCE ();
  if (success)
  {
    delete comservices;
    comservices = new TDll("ole32.dll"); /* do not localize */
    success = comservices->operator HINSTANCE();
  }
  if (success)
  {
    Library = comservices;
    Type = COMPLUS;
    return true;
  } else {
    delete comservices;
    return false;
  }
}

bool TMtsDll::init_mts()
{
  TDll* mtsservices = new TDll("mtxex.dll"); /* do not localize */
  HINSTANCE success = mtsservices->operator HINSTANCE();
```

```
  if (success)
  {
    Library = mtsservices;
    Type = MTS;
    return true;
  } else {
    delete mtsservices;
    return false;
  }
}

HRESULT               TMtsDll::Get_ObjectContext(IObjectContext**
ppInstanceContext)
{
  switch(Type)
  {
    case COMPLUS :
    {
      TDllStdProc2<REFIID,                      IObjectContext**>
BGetObjectContext(*Library, _T("CoGetObjectContext")); /* do not
localize */
      HRESULT hr = E_FAIL;
      if (BGetObjectContext)
        hr        =        BGetObjectContext(IID_IObjectContext,
ppInstanceContext);
      return hr;
    }
    case MTS :
    {
      TDllProc1<IObjectContext**,                         HRESULT>
BGetObjectContext(*Library,  _T("GetObjectContext"));  /*  do  not
localize */
      HRESULT hr = E_FAIL;
      if (BGetObjectContext)
        hr = BGetObjectContext(ppInstanceContext);
      return hr;
    }
    default :
    {
      throw
System::Sysutils::Exception(System::Sysutils::LoadStr(sLoadMTSSe
rvicesError));
    }
  }
}

void* TMtsDll::SafeRef(REFIID rid, IUnknown* pUnknown)
{
  switch(Type)
  {
    case COMPLUS:
    {
      pUnknown->AddRef();
```

```
      return pUnknown;
    }
    case MTS:
    {
      TDllProc2<REFIID,   IUnknown*,   void*>   BSafeRef(*Library,
_T("SafeRef")); /* do not localize */
      if (BSafeRef)
        return BSafeRef(rid, pUnknown);
      else
        return 0;
    }
    default :
    {
      throw
System::Sysutils::Exception(System::Sysutils::LoadStr(sLocateMTS
ServicesError));
    }
  }
}
//---------------------------------------------------------------
-------------

//---------------------------------------------------------------
-------------

#include <fmx.h>
#pragma hdrstop

#include "MEIV_Unit3.h"
//---------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm3 *Form3;
//---------------------------------------------------------------
-------------
__fastcall TForm3::TForm3(TComponent* Owner)
     : TForm(Owner)
{
}
static inline void InitVariant(TVarData &vd)
{
  System::Varutils::VariantInit(vd);
  vd.VInt64 = 0;
}

namespace System
{
#if defined(VARIANT_AUTOMATION_SUPPORT)
  Variant __fastcall Variant::CreateObject(const System::String&
ProgID)
  {
```

```
    return
Variant(System::Win::Comobj::CreateOleObject(const_cast<System::
String&>(ProgID)));
  }

  Variant        __fastcall        Variant::GetActiveObject(const
System::String& ProgID)
  {
    return
Variant(System::Win::Comobj::GetActiveOleObject(const_cast<Syste
m::String&>(ProgID)));
  }

  Variant& Variant::NoParam()
  {
    VariantError err;
    #pragma option push -w-8104
    static Variant _noParam(err);
    #pragma option pop
    _ASSERTE(_noParam.VType == VT_ERROR);
    _ASSERTE(V_ERROR(&_noParam) == DISP_E_PARAMNOTFOUND);
    return _noParam;
  }
#endif // VARIANT_AUTOMATION_SUPPORT

  Variant& Variant::Empty()
  {
    #pragma option push -w-8104
    static Variant _empty;
    #pragma option pop
    _ASSERTE(_empty.VType == varEmpty);
    return _empty;
  }

  // Ctr - From Variants
  __fastcall Variant::Variant()
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));
  }

  __fastcall Variant::Variant(const Variant& rhs)
  {
    // Handle case where this == &rhs (_VarAddRef)
    // NOTE: This copy *MUST* happen before the VARIANTINIT call
    //       because if the compiler calls with this=&rhs, we'll
    //       be wiping out our source!!
    char copy[sizeof(Variant)];
    memcpy(&copy, &rhs, sizeof(copy));

    InitVariant(VARIANT_AS_TVARDATA(*this));

    // Use Delphi assignment operator
```

```
System::Internal::Varhlpr::VariantCpy(*(reinterpret_cast<Variant
*>(&copy)), *this);
  }

  __fastcall Variant::Variant(const TVarData& rhs)
  {
    char copy[sizeof(TVarData)];
    memcpy(&copy, &rhs, sizeof(copy));

    InitVariant(VARIANT_AS_TVARDATA(*this));


System::Internal::Varhlpr::VariantCpy(*(reinterpret_cast<Variant
*>(&copy)), *this);
  }

#if defined(VARIANT_TVARIANT_SUPPORT)
  __fastcall Variant::Variant(const TVariant& rhs)
  {
    ::VariantInit(reinterpret_cast<VARIANTARG*>(this));
    OLECHECK(::VariantCopy(reinterpret_cast<VARIANTARG  *>(this),
const_cast<TVariant*>(&rhs)));
  }
#endif

  inline void __fastcall Variant::SetType(uint16_t vt, bool init)
  {
    if (init)
      InitVariant(VARIANT_AS_TVARDATA(*this));
    else
      Clear();
    VType = vt;
  }

  inline void __fastcall Variant::SetRefType(uint16_t vt, void *p,
bool init)
  {
    SetType(vt|varByRef, init);
    VPointer = p;
  }

  // Ctr - From basic C++ types
  __fastcall Variant::Variant(const bool src)
  {
    SetType(varBoolean, true);
    VBoolean = src ? VARIANT_TRUE : VARIANT_FALSE;
  }

  __fastcall Variant::Variant(const char src)
  {
    SetType(varShortInt, true);
    VShortInt = src;
```

```
}

__fastcall Variant::Variant(const signed char src)
{
  SetType(varShortInt, true);
  VShortInt = src;
}

__fastcall Variant::Variant(const unsigned char src)
{
  SetType(varByte, true);
  VByte = src;
}

__fastcall Variant::Variant(const short src)
{
  SetType(varSmallint, true);
  VSmallint = src;
}

__fastcall Variant::Variant(const unsigned short src)
{
  SetType(varWord, true);
  VWord = src;
}

__fastcall Variant::Variant(const int src)
{
  SetType(varInteger, true);
  VInteger = src;
}

__fastcall Variant::Variant(const unsigned int src)
{
  SetType(varLongWord, true);
  VLongWord = src;
}

__fastcall Variant::Variant(const long src)
{
  SetType(Variant_varLong, true);
  Variant_VLong = src;
}

__fastcall Variant::Variant(const unsigned long src)
{
  SetType(Variant_varULong, true);
  Variant_VLong = src;
}

__fastcall Variant::Variant(const float src)
{
  SetType(varSingle, true);
```

```
    VSingle = src;
  }

  __fastcall Variant::Variant(const double src)
  {
    SetType(varDouble, true);
    VDouble = src;
  }

  __fastcall Variant::Variant(const long double src)
  {
    SetType(varDouble, true);
    VDouble = src;
  }

  __fastcall Variant::Variant(const __int64 src)
  {
    SetType(varInt64, true);
    VInt64 = src;
  }

  __fastcall Variant::Variant(const unsigned __int64 src)
  {
    SetType(varUInt64, true);
    VUInt64 = src;
  }

#if defined(VARIANT_NATIVE_SUPPORT)
  // Ctr - From OLE Structures
  __fastcall Variant::Variant(const VARIANT& src)
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));
    OLECHECK(::VariantCopy(reinterpret_cast<VARIANT*>(this),
const_cast<VARIANT*>(&src)));
  }

  __fastcall Variant::Variant(const CURRENCY& src)
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));
    SET_VTYPE_AND_VAR(CY, src);
  }

  __fastcall Variant::Variant(const DECIMAL& src)
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));
    V_DECIMAL(reinterpret_cast<VARIANT*>(this)) = src;
    V_VT(this) = VT_DECIMAL;
  }

  __fastcall Variant::Variant(const SAFEARRAY& src)
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));
```

```
    VARTYPE _vt;
    ::SafeArrayGetVartype(const_cast<SAFEARRAY*>(&src), &_vt);
    VType = VT_ARRAY|_vt;

    V_ARRAY(this) = const_cast<SAFEARRAY*>(&src);
  }

  __fastcall Variant::Variant(SAFEARRAY* src)
  {
    InitVariant(VARIANT_AS_TVARDATA(*this));

    VARTYPE _vt;
    ::SafeArrayGetVartype(src, &_vt);
    VType = VT_ARRAY|_vt;

    V_ARRAY(this) = src;
  }

  __fastcall Variant::Variant(VARIANT* src)
  {
    ::VariantInit(reinterpret_cast<VARIANTARG*>(this));

    if (src && (src->vt != VT_VARIANT) && (src->vt !=
(VT_VARIANT|VT_BYREF))) {
      SET_VTYPE_AND_VARREF(VARIANT, src);
    } else {
      //
      OLECHECK(::VariantCopy(reinterpret_cast<VARIANTARG
*>(this), src));
    }
  }
//---------------------------------------------------------------
-------------

//---------------------------------------------------------------
-------------

#include <fmx.h>
#pragma hdrstop

#include "MEIV_Unit4.h"
//---------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm4 *Form4;
//---------------------------------------------------------------
-------------
__fastcall TForm4::TForm4(TComponent* Owner)
    : TForm(Owner)
{
}
```

```
//---------------------------------------------------------------
-------------

//---------------------------------------------------------------
-------------

#include <fmx.h>
#pragma hdrstop

#include "MEIV_Unit5.h"
//---------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm5 *Form5;
//---------------------------------------------------------------
-------------
__fastcall TForm5::TForm5(TComponent* Owner)
     : TForm(Owner)
{
}
#if defined(WIDESTRING_IS_BSTR)
  WideString::WideString(const UnicodeString& src)
  {
    Data               =               src.data()               ?
::SysAllocStringLen(reinterpret_cast<const wchar_t*>(src.data()),
                                          src.Length()) : 0;
  }
#else
  WideString::WideString(const UnicodeString& src) : Data(0)
  {
    System::Internal::Strhlpr::WideFromUnicode(*this,
const_cast<UnicodeString&>(src));
  }
#endif

  WideString::WideString(const WideChar* src, int len) : Data(0)
  {
    if (src && (len > 0))
    {
#if defined(WIDESTRING_IS_BSTR)
      Data = ::SysAllocStringLen(src, len);
#else
      SetLength(len);
      memcpy(Data, src, len * sizeof(WideChar));
#endif
    }
  }

#if !defined(WIDECHAR_IS_CHAR16)
  WideString::WideString(const char16_t* src, int numChar16/*= -
1*/) : Data(0)
  {
```

```
    if (src)
    {
#if defined(WIDESTRING_IS_BSTR)
      if (numChar16 > 0)
        Data = ::SysAllocStringLen((wchar_t*)src, numChar16);
      else
        Data = ::SysAllocString((wchar_t*)src);
#else
      if (numChar16 < 0)
        numChar16 = WideCharLen(src);
      SetLength(numChar16);
      memcpy(Data, src, numChar16 * sizeof(WideChar));
#endif
    }
  }
#endif

  WideString::WideString(const char32_t* src, int numChar32/*= -
1*/) : Data(0)
  {
    if (src && (numChar32 != 0))
    {
      int len = UTF32ToUTF16(src, 0, numChar32);
      SetLength(len);
      UTF32ToUTF16(src, Data, numChar32);
    }
  }

  WideString::WideString(const char* src, int len) : Data(0)
  {
    AnsiString str(src, len);
    System::Internal::Strhlpr::WideFromAnsi(*this,
*PRawByteString(&str));
  }


  WideString::WideString(const WideChar* src) : Data(0)
  {
    if (src && *src)
    {
#if defined(WIDESTRING_IS_BSTR)
      Data = ::SysAllocString(src);
#else
      int len = WideCharLen(src);
      SetLength(len);
      memcpy(Data, src, len * sizeof(WideChar));
#endif
    }
  }

#if !defined(WIDECHAR_IS_WCHAR)
  WideString::WideString(const wchar_t* src) : Data(0)
  {
```

```
    // Note: we assume the following:
    // 1) wchar_t uses UCS4 encoding
    // 2) WideChar uses UCS2 encoding
    // 3) it's permissible to "squish" character values that are
too
    // large.
    //
    //    These    are    the    same    assumptions    made    by
System::UCS4StringToWideString
    int len = wcslen(src);
    SetLength(len);
    for (int i = 0; i < len; ++i)
      Data[i] = static_cast<WideChar>(src[i]);
  }

  WideString::WideString(const wchar_t* src, int len) : Data(0)
  {
    // Note: See 'note' above
    SetLength(len);
    for (int i = 0; i < len; ++i)
      Data[i] = static_cast<WideChar>(src[i]);
  }

  WideString::WideString(const wchar_t  src) : Data(0)
  {
    // Note: See 'note' above
    SetLength(1);
    Data[0] = static_cast<WideChar>(src);
  }
#endif

  static WideString IntFmt("%d");
  static WideString UintFmt("%u");
  static WideString FltFmt("%g");

  WideString::WideString(const WideChar src)
    : Data(0)
  {
    WideChar buf[2];
    buf[0] = src;
    buf[1] = 0;
    *this = buf;
  }

  WideString::WideString(char src)
    : Data(0)
  {
    char buf[2];
    buf[0] = src;
    buf[1] = 0;
    *this = buf;
  }
```

```
WideString::WideString(short src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, IntFmt, &r, 0);
}

WideString::WideString(unsigned short src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, UintFmt, &r, 0);
}

WideString::WideString(int src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, IntFmt, &r, 0);
}

WideString::WideString(unsigned int src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, UintFmt, &r, 0);
}

WideString::WideString(long src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, IntFmt, &r, 0);
}

WideString::WideString(unsigned long src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, UintFmt, &r, 0);
}

WideString::WideString(__int64 src)
  : Data(0)
{
  TVarRec r(src);
  Sysutils::WideFmtStr(*this, IntFmt, &r, 0);
}

WideString::WideString(unsigned __int64 src)
  : Data(0)
{
  *this = UnicodeString(src);
```

```
  }

  WideString::WideString(float src)
    : Data(0)
  {
#if defined(__clang__)
    TVarRec r(static_cast<long double>(src));
#else
    TVarRec r(src);
#endif
    Sysutils::WideFmtStr(*this, FltFmt, &r, 0);
  }

  WideString::WideString(double src)
    : Data(0)
  {
#if defined(__clang__)
    TVarRec r(static_cast<long double>(src));
#else
    TVarRec r(src);
#endif
    Sysutils::WideFmtStr(*this, FltFmt, &r, 0);
  }

  WideString::WideString(long double src)
    : Data(0)
  {
    TVarRec r(src);
    Sysutils::WideFmtStr(*this, FltFmt, &r, 0);
  }

  WideString::~WideString()
  {
    if (Data)
    {
      System::Internal::Strhlpr::WideFree(*this);
    }
  }


  WideString& WideString::operator=(const WideString& src)
  {
    System::Internal::Strhlpr::WideAssign(*this,
const_cast<WideString&>(src));
    return *this;
  }

#ifdef WIDESTRING_IS_BSTR
  WideString& WideString::operator=(BSTR src)
  {
    if (Data) {
      ::SysFreeString(Data);
    }
```

```
    Data = (src && *src) ? ::SysAllocString(src) : 0;
    return *this;
  }

  void WideString::Attach(BSTR src)
  {
    // Must Detach what's there before Attaching new BSTR
    //
    _ASSERTE(Data == 0);
    Data = src;
  }

  BSTR WideString::Detach()
  {
    BSTR bstr = Data;
    Data = 0;
    return bstr;
  }

  void WideString::Empty()
  {
    if (Data)
    {
      ::SysFreeString(Data);
      Data = 0;
    }
  }
#else
  WideString& WideString::operator =(const char16_t* rhs)
  {
    WideString temp(rhs);
    this->swap(temp);
  }
#endif

  WideString& WideString::operator+=(const WideString& src)
  {
    System::Internal::Strhlpr::WideAppend(*this,
_STR_CAST(WideString&, src));
    return *this;
  }
//---------------------------------------------------------------------
-------------


//---------------------------------------------------------------------
-------------

#include <fmx.h>
#pragma hdrstop

#include "MEIV_Unit6.h"
```

```
//---------------------------------------------------------------
-------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm6 *Form6;
//---------------------------------------------------------------
-------------
__fastcall TForm6::TForm6(TComponent* Owner)
     : TForm(Owner)
{
}
static inline TPerformanceGraph *ValidCtrCheck()
{
     return new TPerformanceGraph(NULL);
}
//---------------------------------------------------------------
-------------
__fastcall       TPerformanceGraph::TPerformanceGraph(TComponent*
Owner)
     : TGraphicControl(Owner)
{
     ControlStyle << csFramed << csOpaque;

     // default values
     FKind        = pgLine;
     FForeColor = clGreen;
     FBackColor = clBlack;
     FGridSize  = 15;
     FStepSize  = 3;
     FGradient  = 100;
     FGridlines = true;
     FPenWidth  = 2;
     FScale       = 1000;
     Width        = 100;
     Height       = 100;

     Canvas->Pen->Width = FPenWidth;

     // create objects used by class.  Bands is incremented by two
in
     // order to have one TDataPoints object for the oldest,
     //  non-visible  data  (so  that  PaintLine  can  do  its  job
correctly),
     // and one for the current "in-progress" data, which has not
been
     // displayed yet.
     Allocated  = Bands + 2;
     History      = new TDataPoints[Allocated];
     BeginY       = 1;
     CurrentY   = 0;

     Hidden     = 0;
     SaveArea   = new Graphics::TBitmap;
```

```
     Occupied   = false;

     // simulate a history of empty data
     for  (int i = 0; i < Allocated; i++)
          History[i].used = 0;

     // initialize the save area used for painting
     Initialize(GridSize - 1);
}
//---------------------------------------------------------------
-------------
__fastcall TPerformanceGraph::~TPerformanceGraph()
{
     delete SaveArea;
     delete[] History;
}
//---------------------------------------------------------------
-------------


<Project
xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
    <Import
Condition="Exists('$(BDS)\bin\CodeGear.Deployment.targets')"
Project="$(BDS)\bin\CodeGear.Deployment.targets"/>
    <ProjectExtensions>
        <ProjectFileVersion>12</ProjectFileVersion>
    </ProjectExtensions>
    <ItemGroup Condition="'$(Platform)'=='iOSDevice'"/>
    <ItemGroup Condition="'$(Platform)'=='Android'">
        <DeployFile Include=".\Android\Debug\AndroidManifest.xml"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>AndroidManifest.xml</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile           Include="c:\program           files
(x86)\embarcadero\studio\14.0\lib\android\debug\mips\libnative-
activity.so" Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\library\lib\mips\</RemoteDir>
            <RemoteName>libIV.so</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\bin\Artwork\Android\FM_LauncherIcon_72x72.png"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\res\drawable-hdpi\</RemoteDir>
```

```
            <RemoteName>ic_launcher.png</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile           Include="c:\program           files
(x86)\embarcadero\studio\14.0\lib\android\debug\armeabi\libnativ
e-activity.so" Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\library\lib\armeabi\</RemoteDir>
            <RemoteName>libIV.so</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="C:\Users\Public\Documents\Embarcadero\Studio\14.0\Platf
ormSDKs\android-ndk-r9c\prebuilt\android-
arm\gdbserver\gdbserver" Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\library\lib\armeabi-v7a\</RemoteDir>
            <RemoteName>gdbserver</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile            Include=".\Android\Debug\libIV.so"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\library\lib\armeabi-v7a\</RemoteDir>
            <RemoteName>libIV.so</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
            <Required>True</Required>
        </DeployFile>
        <DeployFile           Include="c:\program           files
(x86)\embarcadero\studio\14.0\lib\android\debug\classes.dex"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\classes\</RemoteDir>
            <RemoteName>classes.dex</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile           Include="c:\program           files
(x86)\embarcadero\studio\14.0\lib\android\debug\x86\libnative-
activity.so" Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\library\lib\x86\</RemoteDir>
            <RemoteName>libIV.so</RemoteName>
            <Operation>1</Operation>
```

```
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\bin\Artwork\Android\FM_LauncherIcon_144x144.png"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\res\drawable-xxhdpi\</RemoteDir>
            <RemoteName>ic_launcher.png</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\bin\Artwork\Android\FM_LauncherIcon_96x96.png"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\res\drawable-xhdpi\</RemoteDir>
            <RemoteName>ic_launcher.png</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\bin\Artwork\Android\FM_LauncherIcon_36x36.png"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\res\drawable-ldpi\</RemoteDir>
            <RemoteName>ic_launcher.png</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\bin\Artwork\Android\FM_LauncherIcon_48x48.png"
Condition="'$(Config)'=='Debug'">
            <RemoteDir>IV\res\drawable-mdpi\</RemoteDir>
            <RemoteName>ic_launcher.png</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
    </ItemGroup>
    <ItemGroup Condition="'$(Platform)'=='Win32'">
        <DeployFile            Include="$(BDS)\bin\cc32150mt.dll"
Condition="'$(DynamicRTL)'=='true'                              And
'$(Multithreaded)'=='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>cc32150mt.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
```

```
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile              Include="$(BDS)\bin\cc32150.dll"
Condition="'$(DynamicRTL)'=='true'                              And
'$(Multithreaded)'!='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>cc32150.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile             Include="$(BDS)\bin\borlndmm.dll"
Condition="'$(UsingDelphiRTL)'=='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>borlndmm.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
    </ItemGroup>
    <ItemGroup Condition="'$(Platform)'=='OSX32'">
        <DeployFile Include="$(BDS)\Redist\osx32\libcgcrtl.dylib"
Condition="'$(DynamicRTL)'=='true'">
            <RemoteDir>IV.app\Contents\MacOS\</RemoteDir>
            <RemoteName>libcgcrtl.dylib</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile
Include="$(BDS)\Redist\osx32\libcgunwind.1.0.dylib">
            <RemoteDir>IV.app\Contents\MacOS\</RemoteDir>
            <RemoteName>libcgunwind.1.0.dylib</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile  Include="$(BDS)\Redist\osx32\libcgstl.dylib"
Condition="'$(DynamicRTL)'=='true'">
            <RemoteDir>IV.app\Contents\MacOS\</RemoteDir>
            <RemoteName>libcgstl.dylib</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
    </ItemGroup>
    <ItemGroup Condition="'$(Platform)'=='Win64'">
```

```
        <DeployFile           Include="$(BDS)\bin64\borlndmm.dll"
Condition="'$(UsingDelphiRTL)'=='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>borlndmm.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile          Include="$(BDS)\bin64\cc64150mt.dll"
Condition="'$(DynamicRTL)'=='true'                              And
'$(Multithreaded)'=='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>cc64150mt.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
        <DeployFile            Include="$(BDS)\bin64\cc64150.dll"
Condition="'$(DynamicRTL)'=='true'                              And
'$(Multithreaded)'!='true'">
            <RemoteDir>IV\</RemoteDir>
            <RemoteName>cc64150.dll</RemoteName>
            <Operation>0</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
    </ItemGroup>
    <ItemGroup Condition="'$(Platform)'=='iOSSimulator'">
        <DeployFile
Include="$(BDS)\Redist\osx32\libcgunwind.1.0.dylib">
            <RemoteDir>IV.app\</RemoteDir>
            <RemoteName>libcgunwind.1.0.dylib</RemoteName>
            <Operation>1</Operation>
            <LocalCommand/>
            <RemoteCommand/>
            <Overwrite>True</Overwrite>
        </DeployFile>
    </ItemGroup>
</Project>

#include <fmx.h>
#ifdef _WIN32
#include <tchar.h>
#endif


//---------------------------------------------------------------
-------------

#ifndef MEIV_Unit1H
```

```
#define MEIV_Unit1H
//---------------------------------------------------------------------
-------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.Ani.hpp>
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
//---------------------------------------------------------------------
-------------
class TForm1 : public TForm
{
__published:   // IDE-managed Components
     TBitmapAnimation *BitmapAnimation1;
     TLabel *Label1;
     TLabel *Label2;
     TLabel *Label3;
     TLabel *Label4;
private:  // User declarations
public:        // User declarations
     __fastcall TForm1(TComponent* Owner);
};
//---------------------------------------------------------------------
-------------
extern PACKAGE TForm1 *Form1;
//---------------------------------------------------------------------
-------------
#endif

//---------------------------------------------------------------------
-------------

#ifndef MEIV_Unit2H
#define MEIV_Unit2H
//---------------------------------------------------------------------
-------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.Layouts.hpp>
#include <FMX.Memo.hpp>
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
//---------------------------------------------------------------------
-------------
class TForm2 : public TForm
{
__published:   // IDE-managed Components
     TLabel *Label1;
     TMemo *Memo1;
     TLabel *Label2;
     TLabel *Label3;
```

```
private:  // User declarations
public:         // User declarations
     __fastcall TForm2(TComponent* Owner);
};
//---------------------------------------------------------------
-------------
extern PACKAGE TForm2 *Form2;
//---------------------------------------------------------------
-------------
#endif


//---------------------------------------------------------------
-------------

#ifndef MEIV_Unit3H
#define MEIV_Unit3H
//---------------------------------------------------------------
-------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
//---------------------------------------------------------------
-------------
class TForm3 : public TForm
{
__published:   // IDE-managed Components
     TLabel *Label1;
     TLabel *Label2;
     TLabel *Label3;
     TLabel *Label4;
     TLabel *Label5;
     TRadioButton *RadioButton1;
     TRadioButton *RadioButton2;
     TRadioButton *RadioButton3;
     TRadioButton *RadioButton4;
     TRadioButton *RadioButton5;
     TImageControl *ImageControl1;
     TImageControl *ImageControl2;
     TImageControl *ImageControl3;
     TImageControl *ImageControl4;
     TImageControl *ImageControl5;
private:  // User declarations
public:         // User declarations
     __fastcall TForm3(TComponent* Owner);
};
//---------------------------------------------------------------
-------------
extern PACKAGE TForm3 *Form3;
//---------------------------------------------------------------
-------------
```

```
#endif

//---------------------------------------------------------------------------

#ifndef MEIV_Unit4H
#define MEIV_Unit4H
//---------------------------------------------------------------------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.Edit.hpp>
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
//---------------------------------------------------------------------------
class TForm4 : public TForm
{
__published:   // IDE-managed Components
     TLabel *Label1;
     TLabel *Label2;
     TLabel *Label3;
     TLabel *Label4;
     TLabel *Label5;
     TLabel *Label6;
     TLabel *Label7;
     TLabel *Label8;
     TEdit *Edit1;
     TLabel *Label9;
     TLabel *Label10;
     TLabel *Label11;
private:  // User declarations
public:        // User declarations
     __fastcall TForm4(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TForm4 *Form4;
//---------------------------------------------------------------------------
#endif

//---------------------------------------------------------------------------
#ifndef MEIV_Unit5H
#define MEIV_Unit5H
//---------------------------------------------------------------------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.Edit.hpp>
```

```
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
//---------------------------------------------------------------
-------------
class TForm5 : public TForm
{
__published:   // IDE-managed Components
     TLabel *Label1;
     TLabel *Label2;
     TLabel *Label3;
     TLabel *Label8;
     TLabel *Label11;
     TLabel *Label4;
     TLabel *Label5;
     TEdit *Edit1;
private:  // User declarations
public:        // User declarations
     __fastcall TForm5(TComponent* Owner);
};
//---------------------------------------------------------------
-------------
extern PACKAGE TForm5 *Form5;
//---------------------------------------------------------------
-------------
#endif
//---------------------------------------------------------------
-------------
#ifndef MEIV_Unit6H
#define MEIV_Unit6H
//---------------------------------------------------------------
-------------
#include <System.Classes.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <FMX.Grid.hpp>
#include <FMX.Layouts.hpp>
#include <FMX.StdCtrls.hpp>
#include <FMX.Types.hpp>
#include <System.Rtti.hpp>
//---------------------------------------------------------------
-------------
class TForm6 : public TForm
{
__published:   // IDE-managed Components
     TLabel *Label1;
     TLabel *Label2;
     TLabel *Label3;
     TStringGrid *StringGrid1;
     TStringColumn *StringColumn1;
     TStringColumn *StringColumn2;
     TStringColumn *StringColumn3;
private:  // User declarations
public:        // User declarations
```

```
    __fastcall TForm6(TComponent* Owner);
};
//---------------------------------------------------------------------
-------------
extern PACKAGE TForm6 *Form6;
//---------------------------------------------------------------------
-------------
#endif
```