Технічний коледж ТНТУ імені Івана Пулюя

Циклова комісія комп’ютерних систем і мереж

Курсова робота

з дисципліни :

СИСТЕМНЕ ПРОГРАМУВАННЯ

на тему:

« Розробка програми “Географічний довідник”»

Студента ­­­\_4\_\_курсу \_ОКС-406 групи

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­

(прізвище та ініціали)

напрям підготовки:

**6.050102 «Комп’ютерна інженерія»**

спеціальність:

**5.05010201 «Обслуговування ком’ютерних**

**систем та мереж»**

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка:

ECTS \_\_\_\_\_

Члени комісії\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**Технічний коледж ТНТУ  
 імені Івана Пулюя

**Факультет інформаційних технологій, менеджменту та туризму**

**Спеціальність:**

5.05010201 “Обслуговування комп’ютерних систем і мереж”

ЗАВДАННЯ

**на курсову роботу**

з дисципліни

**Системне програмування**

студент\_\_\_\_ курсу\_\_\_\_\_\_ групи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові)

Тема роботи:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Вихідні дані \_\_\_\_\_\_\_Середовище програмування \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зміст пояснювальної записки

Вступ

1 Загальний розділ

2 Розробка технічного та робочого проекту

3 Тестування та налагодження програм

Висновки

Перелік посилань

Додатки

Дата видачі завдання “\_\_02\_\_” \_\_\_11\_\_\_\_\_\_2016р.

Керівник курсового проекту \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Марціяш Г.Я.

Завдання до виконання отримав “\_02\_\_” \_\_\_\_11\_\_\_\_ 2016р.

Розглянуто на засіданні циклової комісії комп’ютерних систем та мереж

Протокол №3 від “\_26\_\_” \_\_10\_2016р.

Голова ЦК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Юзьків А.В.

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Назва етапів курсової роботи | Строк виконання етапів роботи | Примітка |
| 1 | Отримання і аналіз технічного завдання | 02.11.16р. |  |
| 2 | Збір і узагальнення інформації по роботі | 16.01.17р. |  |
|  | Написання загального розділу |  |  |
|  | Розробка технічного та робочого проекту |  |  |
|  | Тестування та налагодження програм |  |  |
|  | Написання спеціального розділу |  |  |
|  | Виконання графічної частини |  |  |
|  | Оформлення ПЗ |  |  |
|  | Попередній захист | 20.04.17р. |  |

7. Дата видачі завдання \_02.11.2016р.**\_**

Студент \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( підпис ) (прізвище та ініціали)

Керівник проекту \_\_\_\_\_\_\_\_\_\_\_\_\_ Марціяш Г.Я.

( підпис ) (прізвище та ініціали)

Зміст

[ВСТУП 5](#_Toc481538485)

[1 ЗАГАЛЬНИЙ РОЗДІЛ 6](#_Toc481538486)

[1.1 Аналітичний огляд існуючих рішень 6](#_Toc481538487)

[1.2 Технічне завдання 7](#_Toc481538488)

[1.2.1 Найменування та область застосування 8](#_Toc481538489)

[1.2.2 Призначення розробки 8](#_Toc481538490)

[1.2.3 Вимоги до програмного забезпечення 8](#_Toc481538491)

[1.2.4 Вимоги до програмної документації 10](#_Toc481538492)

[1.2.5 Стадії та етапи розробки 11](#_Toc481538493)

[1.2.6 Порядок контролю та прийому 11](#_Toc481538494)

[2 РОЗРОБКА ТЕХНІЧНОГО ТА РОБОЧОГО ПРОЕКТУ 12](#_Toc481538495)

[2.1 Постановка задачі на розробку програмного забезпечення 12](#_Toc481538496)

[2.2 Опис та обґрунтування вибору структури та методу організації вхідних та вихідних даних 12](#_Toc481538497)

[2.3 Опис методів реалізації функцій програми 14](#_Toc481538498)

[2.4 Визначення інформаційних зв'язків 15](#_Toc481538499)

[2.5 Написання текстів програм 17](#_Toc481538500)

[3 ТЕСТУВАННЯ ТА НАЛАГОДЖЕННЯ ПРОГРАМ 19](#_Toc481538501)

[ВИСНОВКИ 24](#_Toc481538502)

[ПЕРЕЛІК ПОСИЛАНЬ 25](#_Toc481538503)

[ДОДАТОК А БЛОК – СХЕМА АЛГОРИТМУ РОБОТИ ПРОГРАМИ 26](#_Toc481538504)

[ДОДАТОК Б ТЕКСТ ПРОГРАМНОГО ФАЙЛУ MAIN.CPP 27](#_Toc481538505)

[ДОДАТОК В ТЕКСТ ПРОГРАМНОГО ФАЙЛУ MAINFORM.H 28](#_Toc481538506)

# ВСТУП

Знання є невід’ємною частиною нашого життя. Ще перші люди використовували примітивні засоби для збереження та накопичення інформації, а зараз ми використовуємо електронні носії для зберігання, упорядкування, передачі, обробки інформації чи даних. З століттям цифрових технологій прийшло безліч речей які спрощують нам пошук інформації, в тому числі інтернет, різноманітні ресурси, довідники з будь-яких дисциплін та на будь-якій мові. Проте з розвитком транспорту та ще й особливостями безвізового режиму особливо популярними стають подорожі. А для того щоб вибрати куди поїхати чи знайти місце для відпочинку можна скористуватись географічним довідником.

Темою курсового проекту є програма «Географічний довідник».

Курсовий проект з навчальної дисципліни “Об’єктно-орієнтоване програмування» виконується з метою закріплення, поглиблення і узагальнення знань, одержаних студентами за час вивчення дисципліни та їх застосування до комплексного вирішення конкретного фахового завдання – розробка програмних продуктів з використанням сучасних інструментальних засобів їх розробки.

Метою цього курсового проекту є написання програмної системи «Географічний довідник» для операційної системи Microsoft Windows, для організації роботи з певною базою знань, використання даних з неї, подання у зручному для користувача вигляді. Програма повинна бути зрозуміла користувачу, надавати екранні підказки. Інформація повинна зберігатись локально, для роботи в режимі офлайн, для повсякденної роботи без інтернету.

На сучасному етапі проект є все ще актуальним, тому що, ринок різноманітних довідників давно потребує конкуруючого рішення.

# 1 ЗАГАЛЬНИЙ РОЗДІЛ

## 1.1 Аналітичний огляд існуючих рішень

Наразі, через розвиток інформаційних технологій дуже багато інформації, в тому числі і про географію, було оцифровано і зберігається тепер в інтернеті. Проте серед кількох існуючих програмних систем можна відмітити кілька рішень, які надають інформацію про різні країни світу:

Довідник географічних координат міст СНД

GEOcoo.exe - довідник географічних координат міст СНД (рис. 1.1) - програма від УФОДОС, включає дані про геокоордінатах міст колишнього Радянського Союзу (4206 населених пунктів). Швидкий пошук, копіювання даних (значення північної широти і східної довготи) в буфер обміну.

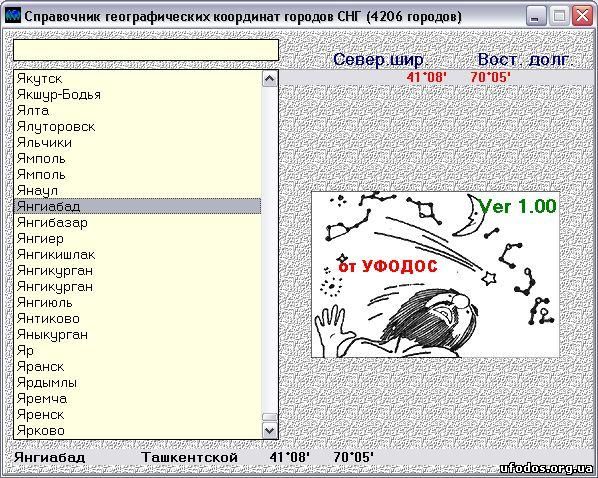


Рисунок 1.1 – довідник міст СНД

Географічний довідник від Igor Gavrilyuk

Інформація про континент, океан, море, країну, прапор, регіон, столицю і міста - все це доступно в нашому додатку Географічний довідник. Воно стане чудовим рішенням, як для любителів географії так і для людей які хочуть розширити свій кругозір. У базі даних міститься 87 морів, 250 країн та їх прапорів, більш ніж 3800 регіонів та 23000 міст усього світу. Цей довідник призначений для платформи Android (рис. 1.2).

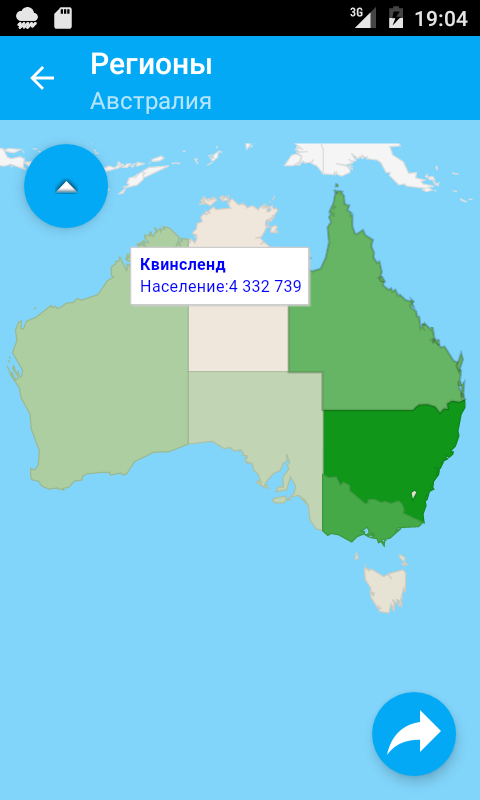


Рисунок 1.2 – Мобільний географічний довідник

## 1.2 Технічне завдання

Основна мета програмного додатку є надання інформації про країни в зручному вигляді.

Основні вимоги: реалізація мовою Visual C++, графічний користувацький інтерфейс за допомогою Windows Forms, інформація повинна зберігатись ззовні програми (для легкого редагування), ведення документації і часте коментування вихідного коду програми.

Програма буде спроектована і реалізована за допомогою мови програмування Visual C++. Графічний користувацький інтерфейс буде виконаний за допомогою технології Windows Forms. Весь процес реалізації буде вестись в інтегрованому середовищі розробки Microsoft Visual Studio 2015 Community Edition.

### 1.2.1 Найменування та область застосування

Повна назва: Географічний довідник.

Програмний виріб буде використовуватись людьми, які часто змушені подорожувати або просто люблять вивчати географію. Також програма знадобиться людям, які бажають вибрати місце відпочинку чи навіть майбутнє місце проживання.

### 1.2.2 Призначення розробки

Для нормального функціонування програмного продукту користувачеві лише необхідно мати встановлену операційну систему сімейства Windows. Ціль використання це отримання нової, цікавої, стислої інформації про певну країну світу. Для цього необхідне лише це програмне забезпечення та комп’ютер.

### 1.2.3 Вимоги до програмного забезпечення

В програмі потрібно реалізувати наступні функції:

* можливість вибору певної країни зі списку;
* можливість збереження інформації;
* зберігання інформації в базі даних;
* надання екранних підказок.

З метою подальшого вдосконалення програмного продукту, а також забезпечення зрозумілості програмного коду, програмний продукт повинен бути написаний на мові програмування C++, чітко до правил програмування даної мови.

#### 1.2.3.1 Вхідні дані

Вхідними даними програми є база даних, в якій зберігаються дані про держави разом із шляхами до зображень, які відповідають прапорам цих держав. Зображення зберігаються в форматі png підвантажуються в пам’ять з папки images.

#### 1.2.3.2 Вихідна інформація

Через те що прогрмама має інформаційно-пошуковий характер система то вихіднимі дані це вивід інформації про держави світу на екран комп’ютера інформації. Також програма може генерувати та виводити дані для друку на папері та текстовий документ txt.

#### 1.2.3.3 Опис функцій та обмежень

Основною функцією програми є вивід на екранну форму користувацького інтерфейсу докладної текстової та графічної інформації про визначену країну, для цього програма повинна підключатись до бази даних і зчитувати звідти дані.

#### 1.2.3.4 Часові характеристики

Програма буде виконана за допомогою мови програмування C++, тому за швидкодія програми буде високою, а час запуску – нетривалим. Нормальною вимогою буде якщо на запуск програми необхідно 1-5 секунд. Оскільки це інформаційно-пошукова система і дані беруться з бази даних то читання даних може становити до 10 сек. на відповідь на запит.

#### 1.2.3.5 Вимоги до надійності

З метою забезпечення надійного функціонування програми повинні бути передбачені:

* + ієрархічна структура комплексу;
  + блоки, які контролюють усунення збійних ситуацій в роботі системи (особливо під час під’єднання до бази даних);
  + контроль вводу даних, який забезпечує стійкість програми до помилок користувача (поле для користувацького вводу для здійснення пошуку). Повинна бути перевірка на порожню строку, цифри, та інші непередбачені значення.
  + обробка виняткових ситуацій;
  + видача чітких та корисних повідомлень про місцезнаходження помилки та її характер за допомогою виключень.

#### 1.2.3.6 Умови експлуатації

Програмне забезпечення вимагає від користувача базових навичок роботи з персональним комп’ютером та використовуваною операційною системою Windows. Інтерфейс повинен бути максимально простим і зрозумілим для кінцевого користувача.

Для повноцінної роботи з програмою вона повинна знаходитися на жорсткому диску комп’ютера, флеш-носії чи іншому носії інформації, який забезпечить програмі можливість запуску.

Оскільки програма несе інформаційно-пізнавальний характер то режим взаємодії користувача з програмним комплексом передбачається лише один – робота з екранною формою.

Оскільки реалізація програми максимально проста тому і системні вимоги до інформаційної та програмної сумісності будуть мінімальними – операційна система Microsoft Windows XP і сучасніше, а також мишка і екран з роздільною здатністю 800\*600 пікселів.

### 1.2.4 Вимоги до програмної документації

Програмна реалізація повинна супроводжуватись файлом ReadMe.txt з вказанням ключових моментів роботи програми.

Всі блоки вихідного коду повинні бути покриті коментарями, з чітким і стислим поясненням що і навіщо цей код виконує. Кожний програмний модуль або процедурний блок повинен мати початковий блок коментарів, який містить призначення, склад вхідних та вихідних даних, обмеження та умови використання.

Кожний модуль повинен мати одну точку входу та одну точку виходу.

### 1.2.5 Стадії та етапи розробки

Весь процес розробки можна поділити на кілька стадій:

1. Постановка завдання – формулювання функцій, вимог до програмної системи.
2. Проектування – тобто вибір середовища, мови програмування, вхідних даних, складання алгоритмів задач.
3. Створення проекту, написання вихідних кодів, оформлення графічного користувацького інтерфейсу.
4. Тестування і відладка роботи програми і графічного інтерфейсу.
5. Аналіз результатів.

### 1.2.6 Порядок контролю та прийому

Перш за все програма повинна повідомляти користувача, якщо при запуску програми база даних відсутня. Далі повинне перевірятись переключення між різними країнами. Також необхідною умовою є наявність красивого, читабельного графічного користувацького інтерфейсу.

# 2 РОЗРОБКА ТЕХНІЧНОГО ТА РОБОЧОГО ПРОЕКТУ

## 2.1 Постановка задачі на розробку програмного забезпечення

Перед нами ставиться задача розробити програмне забезпечення для операційної системи MS Windows, яке буде імітувати географічний довідник в цифровому вигляді та виконувати поставлені вище функції. Програмне забезпечення повинне відповідати вимогам визначеним вище в підрозділі «Вимоги до програмного забезпечення».

З специфічних особливостей програми можна відмітити лише взаємодію з базою даних, та необхідність побудови запитів мовою SQL.

## 2.2 Опис та обґрунтування вибору структури та методу організації вхідних та вихідних даних

Оскільки програма несе інформаційно-пізнавальний характер тому даних в ній доволі багато. Найкращий ступінь «структуризації» інформації на даному етапі інформатизації мають бази даних. Бази даних дозволяють створити пов’язані сутності, об’єднати схожі сутності, мінімізувати повтори інформації. Також вхідними даними можна вважати зображення, які представляють прапори країн.

Структура вхідних даних програмного забезпечення представлені базою даних створеною Microsoft Access. Назва файлу – database.mdb. MS Access - система управління базами даних, програма, що входить до складу пакету офісних програм Microsoft Office. Має широкий спектр функцій, включаючи зв'язані SQL запити, сортування по різних полях, зв'язок із зовнішніми таблицями і базами даних. З бази даних береться і гіперпосилання, яке містить шлях до зображення в вигляді прапору країни. Зображення зберігаються в папці «images» в форматі png.

Перевагою MS Acceess є використання SQL запитів що дуже спрощує взаємодію з даними в базі. SQL (англ. Structured query language — мова структурованих запитів) — декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування (C або Pascal), SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних.

Дані з бази даних будуть вибиратись за допомогою «сирих» запитів SQL які будуть звертатись безпосередньо до бази даних.

Структура бази даних, яка містить таблицю з вхідною інформацією про країни світу подана в таблиці 2.1.

Таблиця 2.1

Структура таблиці «table» бази даних

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Атрибути сутності | Обмеження атрибутів | | | | | Ключ |
| Тип даних | Межі або допустимі значення | Структура (формат) | Умова | Значення за замовчуванням |
| Номар | Автонумерація | - | Довге ціле число | Not NULL | - | P |
| Назва | Короткий текст | 255 | - | Not NULL | - | - |
| Коротка інформація | Довгий текст | - | - | Not NULL | - | - |
| Повна інформація | Довгий текст | - | - | Not NULL | - | - |
| Зображення | Гіперпосилання | - | - | - | - | - |

Вихідна інформація.

Вихідна інформація представлена тільки у вигляді роздруківок інформації з програми про певну країну. Програма здатна генерувати сторінку, яку можна переглянути і автоматично передати на друк використовуючи принтер користувача.

## 2.3 Опис методів реалізації функцій програми

Програма запускається з головної функції і одразу ж створює новий екземпляр екранної форми MainForm.

Далі відбувається вивід головної форми програмного засобу, на ній відображаються всі елементи керування.

При натисканні на кнопку «Про програму» відбудеться запуск відповідної форми «Про програму», яка демонструє користувачу розробника програми.

Якщо в поле пошуку ввести назви держави то в списку відобразиться вона за наявності в базі даних.

Також на формі є кнопка друк, вона викликає код для формування та виведення відомостей про державу на аркуш паперу. Кнопка зберегти дозволяє зберегти цю інформацію в файл.

## 2.4 Визначення інформаційних зв'язків

Структурна схема програмного комплексу, яка демонструє визначення інформаційних зв’язків в програмі (рис. 2.1).

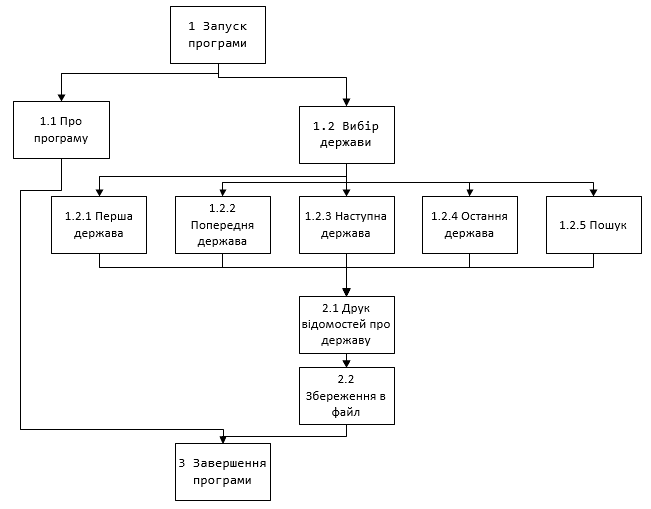


Рисунок 2.1 – Структурна схема програми.

|  |  |  |
| --- | --- | --- |
| № п/п | Призначення блоку | Назва програмного блоку |
| 1 | Вивід головної форми програмного засобу | Запуск програми |
| 1.1 | Вивід форми «Про програму» | Про програму |
| 1.2 | Вивід держави на екранну форму | Вибір держави |
| 1.2.1 | Натискання кнопки «Перша держава» | Перша держава |
| 1.2.2 | Натискання кнопки «Попередня держава» | Попередня держава |
| 1.2.3 | Натискання кнопки «Наступна держава» | Наступна держава |
| 1.2.4 | Натискання кнопки «Остання держава» | Остання держава |
| 1.2.5 | Введення назви держави | Пошук |
| 2.1 | Виведення відомостей про державу на аркуш паперу | Друк відомостей про державу |
| 2.2 | Збереження даних в текстовий документ | Збереження в файл |
| 3 | Завершення роботи програми | Завершення програми |

Функціональні специфікації повинні проектуватися й описуватись за рівнями (зверху-вниз). На верхньому рівні виділяються групи функцій , на наступних рівнях вони деталізуються. Верхній рівень специфікації програми подається у вигляді ієрархічної схеми зовнішніх функцій програмного виробу. Функціональні специфікації описують кожне можливе введення даних у програму, в тому числі і не передбачених, та реакцію програми на нього. До "даних" належать директиви , які вводяться користувачем; оперативні дані, які вводяться з клавіатури; дані, які зберігаються в базі даних.

На нижньому рівні, рівні функцій користувача, необхідно вказати такі види інформації: опис вводу, опис виводу, перетворення: опис змін стану системи, які відбуваються під час виконання функції. Слід пам’ятати, що ці перетворення повинні бути описані з точки зору користувача, і не торкатись питань внутрішньої побудови програмного забезпечення. Зв’язки по управлінню між функціями програмного комплексу і зображуються у вигляді структурної моделі програми, що може бути подана у графічному та текстовому вигляді . У таблиці змісту вказуються найменування функцій, ідентифікаційні номери блок-схем, в яких поданий їх алгоритм, та імена програмних модулів, що реалізують ці функції.

## 2.5 Написання текстів програм

Програма запускається з функції WinMain яка виконує початкові налаштування візуального стилю та створює новий екземпляр екранної форми MainForm.

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew MainForm);

return 0;

}

Для створення з’єднання з базою даних створюється елемент OleDbConnection.

//створюємо датасет, конекшн, і команду

oleConnection = gcnew OleDb::

OleDbConnection(

"Data Source=\"database.mdb\"; User ID=Admin;Provider=\"Microsoft.Jet.OLEDB.4.0\";");

dataSet = gcnew DataSet();

oleCommand = gcnew OleDb::OleDbCommand();

try

{

//під'єднуємося до бд

if (oleConnection->State == ConnectionState::Closed)

oleConnection->Open();

}

catch (Exception^ e)

{

MessageBox::Show("Проблема з базою даних\n", e->Message);

}

Для того щоб отримати дані від з’єднання використовуємо елемент OleDbDataAdapter.

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table]", oleConnection);

Тоді якщо необхідно вибрати дані створюємо елемент DataSet:

//створюємо датасет, конекшн, і команду

dataSet = gcnew DataSet();

…

//заносимо в кеш

oleDataAdapter->Fill(dataSet, "[table]");

Програмний код для кнопки «Друкувати» викликає діалогове вікно printPreviewDialog1.

printPreviewDialog1->ShowDialog();

Елемент printPreviewDialog1 має властивість «Document» в якому виставлено printDocument1. Останній елемент формує сторінку для друку. Виконується це кодом.

//формуємо текст на друк

String^ fullText = txtBoxFullInfo->Text;

…

String^ printText = "Назва країни: " + lblCountryName->Text +

"\nКоротка інформація:\n" + txtBoxMainInfo->Text +

"\nПовна інформація:\n" + fullText;

//відображаємо аркуш

e->Graphics->DrawString(printText,

gcnew System::Drawing::Font("Arial", 12, FontStyle::Regular),

Brushes::Black, 20, 25);

При натисканні на кнопку «Про програму» появиться нове вікно представлення творця курсової роботи.

//запуск форми

AboutForm^ frm = gcnew AboutForm;

frm->Show(this);

Для збереження файлу створюється елемент saveFileDialog:

//відкриваємо діалог

saveFileDialog1->ShowDialog();

saveFileDialog1->DefaultExt = "\*.cpp";

saveFileDialog1->Filter = "Cpp files |\*.cpp";

…

//записуємо

file->WriteLine(strFile);

//закриваємо потік

file->Close();

# 3 ТЕСТУВАННЯ ТА НАЛАГОДЖЕННЯ ПРОГРАМ

Для початку тестування необхідно перевірити наявність та коректність вхідних даних які в нашому випадку представлені базою даних (рис. 3.1) та графічними зображеннями (рис. 3.2).

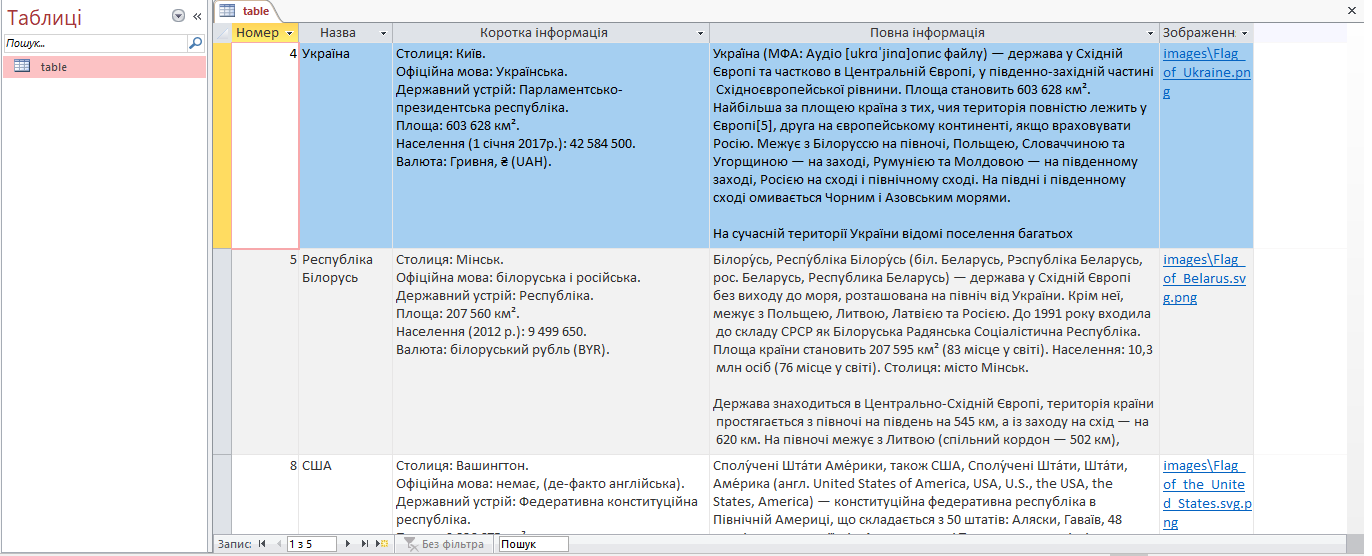


Рисунок 3.1 – Наявна інформація в базі даних.

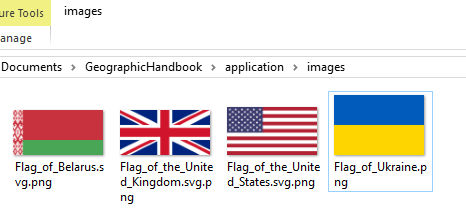


Рисунок 3.2 – Вхідні зображення в папці «images».

Оскільки вхідна інформація може бути відсутня ми повинні протестувати чи спрацює наш захист в вигляді обробки виключних ситуацій (рис. 3.3).

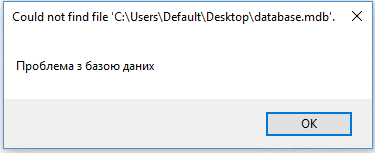


Рисунок 3.3 – запуск програми без бази даних.

Отже наша програма запустилася (рис. 3.4) і все добре.

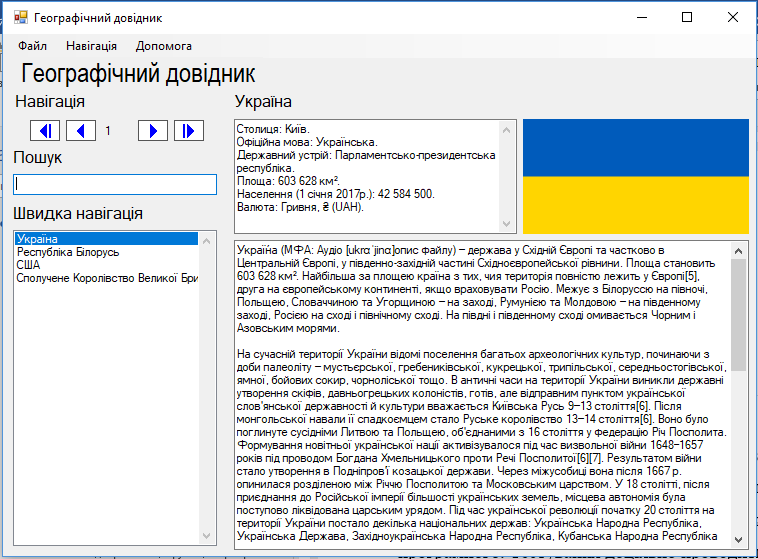


Рисунок 3.4 – Головна форма програми.

Одразу ж можемо перевірити правильність роботи функції «Про програму» (рис. 3.5).

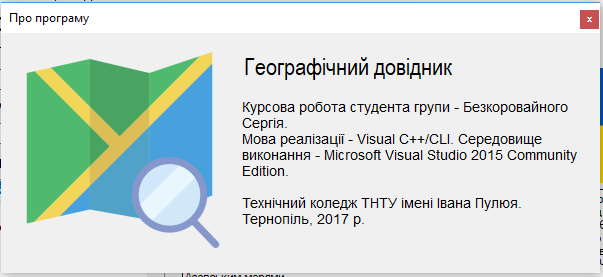


Рисунок 3.5 – Форма «Про програму».

Також можемо переконатися що екранні підказки ToolTip працюють (рис. 3.6).

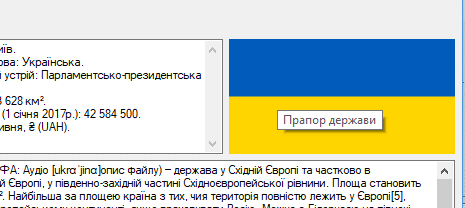


Рисунок 3.6 – Демонстрація екранної підказки.

Для тестування пошуку потрібно ввести в поле пошуку необхідну назву держави (рис. 3.7) і тоді ж програма покаже чи є вона в наявності (рис. 3.8).

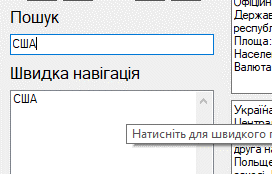


Рисунок 3.7 – Введення назви держави в полі пошуку.

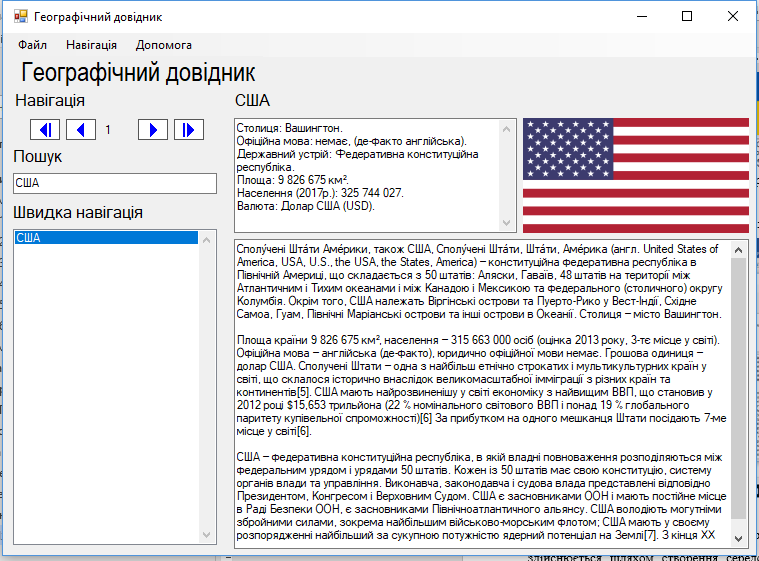


Рисунок 3.8 – Результати пошуку.

На кінець можна протестувати функцію друку (рис. 3.9). Для цього достатньо натиснути кнопку «Друк» в вкладці «Файл» (рис. 3.10).

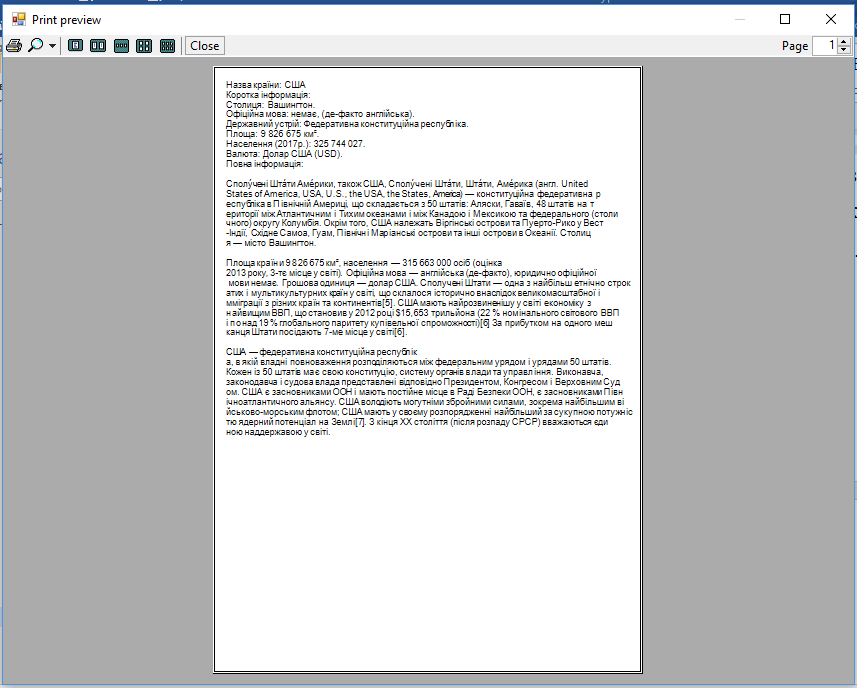


Рисунок 3.9 – Згенерована сторінка для друку в режимі перегляду.

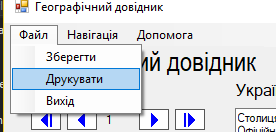


Рисунок 3.10 – Вкладка «Файл».

Також можна перевірити функцію збереження файла, вона знаходить в тій же вкладці «Файл».

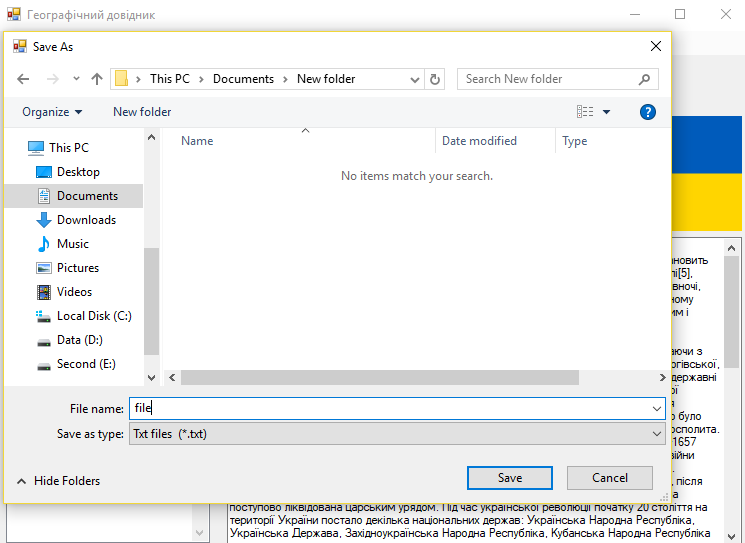


Рисунок 3.11 – Діалог збереження файла.

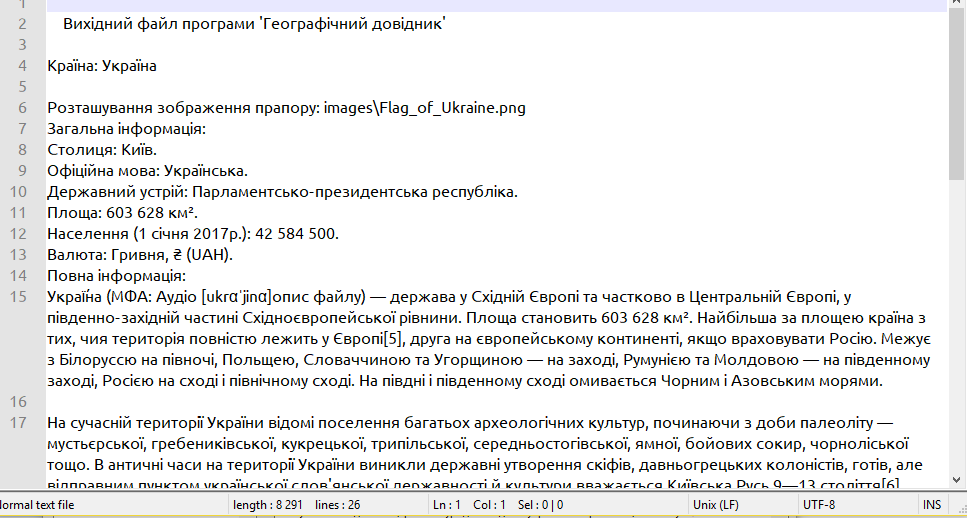


Рисунок 3.12 - Вміст збереженого файла.

З цього можна вивести висновок, що тестування та налагоджування програм є дуже важливими діями. Тестування здійснюється шляхом умов використання, схожими до умов використання кінцевого користувача. Створена програма працездатна, виконує всі необхідні функції, відповідає визначеним вимогам, проходить всі тести по швидкодії та є зручною в користуванні.

# ВИСНОВКИ

В результаті виконання курсового проекту було розроблено програму «Географічний довідник», яка дозволяє користувачу отримувати інформацію про різні країни світу. Дана програма володіє зручним та доволі простим у користуванні та зрозумілим для користувача інтерфейсом. Пошук дозволить користувачам швидше знайти необхідну їм державу.

Перевагами даного програмного продукту є легкий доступ до усіх країн світу які містяться в базі даних. Також плюсом є те, що програма адаптована для широкого кола користувачів, які бажають дізнатись більше про географію. Програма досить проста у використанні, перекладена на українську і містить екранні підказки та можливість простого і швидкого друку на принтері. Можна відмітити і доволі економне використання ресурсів: проект не займе багато місця на жорсткому диску і використовує мало ресурсів системи.

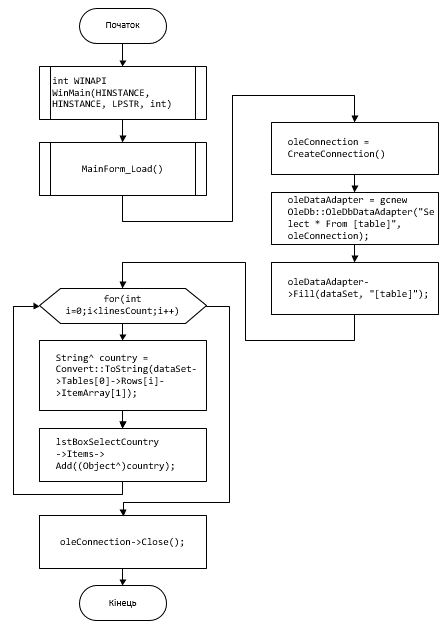
До недоліків даної програми можна віднести те що програмний продукт мало функціональний та містить тільки список визначених країн, без інших географічних елементів, таких як розташування на інтерактивній карті.

Для вдосконалення даного продукту можна добавити функції: додавання та редагування країни прямо в програмі за допомогою панелі адміністратора або розширений пошук.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Основи алгоритмізації та програмування. 777 задач з рекомендаціями та прикладами / Т.П. Караванова. – К.: Генеза, 2006. – 288с.
2. Изучаем программирование на C / Дэвид Гриффитс / Дон Гриффитс – К: – Ексмо, 2013. - 625с.
3. Microsoft Visual Studio 2008 / Л. Пауэрс, М. Снелл: Пер. С англ. – К.: БХВ-Петербург, 2009 – 1200 с.
4. Пахомов Б.И. С/С++ и MS Visual C++ 2010 для начинающих. – К.: БХВ-Петербург, 2011. – 736 с.
5. Программирование: принципы и практика использования С++.: Пер. с англ. / Страуструп Бьярне – К.: Вильямс, 2011. – 1248 с.
6. C++. Базовий курс / Герберт Шилдт – К.: Вільямс, 2015. - 624с.
7. Programming with C++ / B. L. Juneja, Anita Seth – К.: New Age International, 2009. – 673с.
8. C++ для професіоналів / Ніколас А. Солтер, Майкл Л. Клепер – К.: Вільямс, 2006. – 912с.
9. Exceptional C++ / Herb Sutter – К.: Вільямс, 2015. – 400.
10. Visual C++ 2008: базовый курс. : Пер. с англ. / Хортон, Айвор – К. : Вильямс, 2009. – 1152 с.
11. MS Visual C++ 2010 в среде .net / В. Зиборов – К.: Пітер, 2012. – 320 c.

# ДОДАТОК А БЛОК – СХЕМА АЛГОРИТМУ РОБОТИ ПРОГРАМИ



# ДОДАТОК Б ТЕКСТ ПРОГРАМНОГО ФАЙЛУ MAIN.CPP

#include "MainForm.h"

#include <Windows.h>

//текст головного файлу

using namespace DB;

[STAThread]

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew MainForm);

return 0;

}

# ДОДАТОК В ТЕКСТ ПРОГРАМНОГО ФАЙЛУ MAINFORM.H

#include "AboutForm.h"

#include <iostream>

#include <fstream>

#include <string>

#pragma once

namespace DB {

using namespace std;

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

public ref class MainForm : public System::Windows::Forms::Form

{

public:

MainForm(void)

{

InitializeComponent();

}

protected:

~MainForm()

{

if (components)

{

delete components;

}

}

…

}

#pragma endregion

//глобальні змінні для операцій з бд

DataSet ^ dataSet;

OleDb::OleDbDataAdapter ^ oleDataAdapter;

OleDb::OleDbConnection^ oleConnection;

OleDb::OleDbCommand ^ oleCommand;

//глобальні змінні

int sel = 0;

int linesCount = 0;

int selCountry = 0;

private: OleDb::OleDbConnection^ CreateConnection()

{

//створюємо датасет, конекшн, і команду

oleConnection = gcnew OleDb::

OleDbConnection(

"Data Source=\"database.mdb\"; User ID=Admin;Provider=\"Microsoft.Jet.OLEDB.4.0\";");

dataSet = gcnew DataSet();

oleCommand = gcnew OleDb::OleDbCommand();

try

{

//під'єднуємося

if (oleConnection->State == ConnectionState::Closed)

oleConnection->Open();

}

catch (Exception^ e)

{

MessageBox::Show("Проблема з базою даних\n", e->Message);

}

return oleConnection;

}

private: System::

Void MainForm\_Load(System::Object^ sender, System::EventArgs^ e)

{

//запуск форми

//створюємо зв'язок

oleConnection = CreateConnection();

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table]", oleConnection);

//заносимо в кеш

oleDataAdapter->Fill(dataSet, "[table]");

//отримуємо кількість заповнених рядків

linesCount = (dataSet->Tables[0]->Rows->Count) - 1;

lstBoxSelectCountry->Items->Clear();

for(int i=0;i<linesCount;i++)

{

//заповнюємо комбобокс

String^ country = Convert::ToString(dataSet->Tables[0]->Rows[i]->ItemArray[1]);

lstBoxSelectCountry->Items->Add((Object^)country);

}

//початкова ініціалізація

selCountry = 1;

lblNum->Text = Convert::ToString(selCountry);

lstBoxSelectCountry->SelectedIndex = selCountry-1;

//від'єднуємося від бази

oleConnection->Close();

}

private: System::Void lstBoxSelectCountry\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e)

{

//вибір країни в лістбокс

//створюємо зв'язок

oleConnection = CreateConnection();

//отримуємо індекс вибраного елементу

selCountry = lstBoxSelectCountry->SelectedIndex;

lblNum->Text = Convert::ToString(selCountry + 1);

//якщо поле пошуку порожнє

if (txtBoxSearchCountry->Text == "")

{

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table]", oleConnection);

}

else

{

//якщо в полі пошуку є текст

String^ search = txtBoxSearchCountry->Text;

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table] where [Назва] like ('%" + search + "%');", oleConnection);

}

//заносимо в кеш

oleDataAdapter->Fill(dataSet, "[table]");

//заповнюємо поля форми

lblCountryName->Text = Convert::ToString(dataSet->Tables[0]->Rows[selCountry]->ItemArray[1]);

txtBoxMainInfo->Text = Convert::ToString(dataSet->Tables[0]->Rows[selCountry]->ItemArray[2]);

txtBoxFullInfo->Text = Convert::ToString(dataSet->Tables[0]->Rows[selCountry]->ItemArray[3]);

//отримуємо малюнок і запихаємо в picturebox

String^ imgLocation = Convert::ToString(dataSet->Tables[0]->Rows[selCountry]->ItemArray[4]);

if (imgLocation != "")

{

imgLocation = imgLocation->Remove(imgLocation->IndexOf("#"));

pictCountryFlag->ImageLocation = imgLocation;

}

//вираховуємо кількість рядків таблиці

linesCount = (dataSet->Tables[0]->Rows->Count) - 1;

//від'єднуємося від бази

oleConnection->Close();

}

private: System::Void btnPrev\_Click(System::Object^ sender, System::EventArgs^ e) {

//перехід до попереднього елемента

if (selCountry >= 1)

{

lblNum->Text = Convert::ToString(Convert::ToInt32(lblNum->Text) - 1);

selCountry--;

lstBoxSelectCountry->SelectedIndex -= 1;

}

}

private: System::Void btnNext\_Click(System::Object^ sender, System::EventArgs^ e) {

//перехід до наступного елемента

if (selCountry < linesCount - 1)

{

lblNum->Text = Convert::ToString(Convert::ToInt32(lblNum->Text) + 1);

selCountry++;

lstBoxSelectCountry->SelectedIndex += 1;

}

}

private: System::Void btnFirst\_Click(System::Object^ sender, System::EventArgs^ e) {

//перехід до першого елемента

if (lstBoxSelectCountry->Items->Count != 1)

{

lblNum->Text = Convert::ToString(1);

selCountry = 1;

lstBoxSelectCountry->SelectedIndex = 0;

}

}

private: System::Void btnLast\_Click(System::Object^ sender, System::EventArgs^ e) {

//перехід до останнього елемента

linesCount = lstBoxSelectCountry->Items->Count;

if (linesCount != 1)

{

lblNum->Text = Convert::ToString(linesCount);

selCountry = linesCount - 1;

lstBoxSelectCountry->SelectedIndex = selCountry;

}

}

private: System::Void txtBoxSearchCountry\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

//введення тексту в пошуковий текстбокс

//створюємо зв'язок

oleConnection = CreateConnection();

lstBoxSelectCountry->Items->Clear();

//якщо в полі є текст

if(txtBoxSearchCountry->Text != "")

{

String^ search = txtBoxSearchCountry->Text;

lstBoxSelectCountry->Items->Clear();

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table] where [Назва] like ('%" + search + "%');", oleConnection);

//заносимо в кеш

oleDataAdapter->Fill(dataSet, "[table]");

for (int i = 0; i<dataSet->Tables[0]->Rows->Count; i++)

{

//заповнюємо комбобокс

String^ country = Convert::ToString(dataSet->Tables[0]->Rows[i]->ItemArray[1]);

lstBoxSelectCountry->Items->Add((Object^)country);

}

}

else

{

//виводимо всі дані

//отримуємо дані

oleDataAdapter = gcnew OleDb::OleDbDataAdapter("Select \* From [table]", oleConnection);

//заносимо в кеш

oleDataAdapter->Fill(dataSet, "[table]");

for (int i = 0; i<dataSet->Tables[0]->Rows->Count; i++)

{

//заповнюємо комбобокс

String^ country = Convert::ToString(dataSet->Tables[0]->Rows[i]->ItemArray[1]);

lstBoxSelectCountry->Items->Add((Object^)country);

}

}

}

private: System::Void lstBoxSelectCountry\_MouseClick(System::Object^ sender, System::Windows::Forms::MouseEventArgs^ e) {

//клік по лістбоксі

lstBoxSelectCountry\_SelectedIndexChanged( sender, e);

}

private: System::Void btnPrint\_Click(System::Object^ sender, System::EventArgs^ e) {

printPreviewDialog1->ShowDialog();

}

private: System::Void printDocument1\_PrintPage(System::Object^ sender, System::Drawing::Printing::PrintPageEventArgs^ e) {

//формуємо текст на друк

String^ fullText = txtBoxFullInfo->Text;

for (int i = 0; i < txtBoxFullInfo->Text->Length - 1; i+=90)

{

fullText = fullText->Insert(i, "\n");

}

String^ printText = "Назва країни: " + lblCountryName->Text +

"\nКоротка інформація:\n" + txtBoxMainInfo->Text +

"\nПовна інформація:\n" + fullText;

//відображаємо аркуш

e->Graphics->DrawString(printText,

gcnew System::Drawing::Font("Arial", 12, FontStyle::Regular),

Brushes::Black, 20, 25);

}

private: System::Void проПрограмуToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

showAbout();

}

private: void showAbout()

{

//запуск форми

AboutForm^ frm = gcnew AboutForm;

frm->Show(this);

}

private: System::Void зберегтиToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

saveToFile();

}

private: void saveToFile()

{

//записуємо в файл

String^ main = txtBoxMainInfo->Text;

String^ full = txtBoxFullInfo->Text;

String^ name = lblCountryName->Text;

String^ pict = pictCountryFlag->ImageLocation;

//об'єднуємо строки

String^ strFile = "\n \tВихідний файл програми 'Географічний довідник' \n\n" + "Країна: " + name

+ "\n\nРозташування зображення прапору: " + pict

+ "\nЗагальна інформація:\n" + main

+ "\nПовна інформація:\n" + full;

//відкриваємо діалог

saveFileDialog1->ShowDialog();

saveFileDialog1->DefaultExt = "\*.cpp";

saveFileDialog1->Filter = "Cpp files |\*.cpp";

String^ filename = saveFileDialog1->FileName;

System::IO::StreamWriter^ file = gcnew System::IO::StreamWriter(filename);

//записуємо

file->WriteLine(strFile);

//закриваємо потік

file->Close();

}

};

}