**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСЬКИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

**КУРСОВА РОБОТА**

з дисципліни

**«Програмування (Рівень В- Об’єктно-орієнтовне програмування)»**

(назва дисципліни)

на тему: **Simulator of the motion planets**

Студента(ки) 1 курсу 107-Кбс групи

спеціальності «Кібербезпека»

. Москалик А.В. .

(прізвище та ініціали)

Керівник

к.т.н., доцент .

(посада, вчене звання, науковий ступінь,

. Гордєєв О.О. .

прізвище та ініціали)

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

Кількість балів: \_\_\_\_

Оцінка: ECTS \_\_\_\_

Члени комісії

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

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

(підпис) (прізвище та ініціали)

Київ - 2019 рік

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСКЬИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

Спеціальність: «Кібербезпека»

Курс 1 Група 107-Кбс

Завдання

На курсову роботу студента

Москалика Артема

1. Тема курсової роботи: Simulator of the motion planets
2. Термін здачі курсової роботи студентом: 19.12.2019
3. Вихiднi данi до (роботи): інформаційно-аналітичні інтернет джерела, зразки рішень практичних завдань та задач, нормативно-правова база, щодо оформлення технічної документації програмних продуктів, початкові значення для обробки.
4. Змiст пояснювальної записки (перелiк питань, які належить розробити) Вступ. Формулювання вимог до програми. Проектування та практична реалізація розв’язку поставленої задачі. Тестування програми і результати її виконання. Висновки.
5. Перелiк графiчного матерiалу (з точним зазначенням обов`язкових креслень) Таблиці та схеми представлено в роботі.
6. Дата видачі завдання «15» листопада 2019 р.

**Зміст**

[**1. ВСТУП** 4](#_Toc532674954)

[**2. оСНОВНА чАСТИНА** 5](#_Toc532674955)

[**2.1 Опис предметної області** 5](#_Toc532674956)

[**2.1.1 Класифікація баз даних** 6](#_Toc532674957)

[**2.1.2 Області застосування баз даних** 7](#_Toc532674958)

[**2.1.3 Основні відомості про середовище розробки** 8](#_Toc532674959)

[**2.1.3.1 Microsoft Visual Studio** 8](#_Toc532674960)

[**2.1.3.2 Microsoft SQL Server** 11](#_Toc532674965)

[**2.3 Структура таблиці** 14](#_Toc532674968)

[**2.4 Охорона праці** 15](#_Toc532674969)

[**3.ВИСНОВОК** 21](#_Toc532674970)

[**Список використаних джерел** 22](#_Toc532674971)

Додатки

Додаток А. Блок-схема програмного продукту.

Додаток В. Лістінг програмного продукту.

Додаток С. Скрін-шоти функціонування програмного продукту.

**1. ВСТУП**

В даній курсовій роботі була виконана розробка програми на мові програмування С++ з використанням графіки SFML 2.4.2, яка допомагає розробнику програми зробити інтерфейс та саме вікно привабливішим.

У роботі детально обґрунтована інформація щодо планет сонячної системи, до дрібниць опрацьований розмір, масштабування та швидкість.

Практична частина курсової роботи виконана у програмі Microsoft Visual Studio 13, в середовищі розробки C++/C#/SFML 2.4.2 з підключенням бази даних SQL.

Мій проект надає користувачеві ознайомитися з сонячною системою, рухом планет і інформацією про них. В даному проекті користувач моє можливість поглянути на симуляцію руху планет з точнім рухом та розміщенням. Також вся інформація зберігається у базі даних, і сам користувач(якщо знайде помилку) редагувати інформацію про всі планети та зірку Сонце. Практична цінність мого проекту полягає в тому що користувачеві надається у одній програмі все зразу.

В проекті використовуються 4 основні функції(Insert, Select, Update, Delete). Кожен користувач за власним бажанням може змінювати інформацію для своїх потреб у житті.

А також в меню проекту є системні мінімальні характеристики, якщо у користувача менші характеристики то шанс запуску стимулятора зменшується.

**2. оСНОВНА чАСТИНА**

**2.1 Опис предметної області**

Протягом життя людина накопичує різноманітну інформацію: телефони, адреси, імена та дати народження друзів і знайомих; відомості про те, які книги зберігаються у власній бібліотеці, які ігри, аудіо- та відеозаписи знаходяться на дисках тощо. Такі дані звичайно зберігаються в систематизованому вигляді з тим, щоб ними було зручно користуватися.

Діяльність будь-якої організації, фірми, підприємства невід’ємна від створення й застосування сховищ даних, які стосуються осіб, що працюють на виробництві, товарів, що випускаються, матеріалів, які необхідні для їх виготовлення, тощо. Такі сховища потрібні для управління виробництвом, організації економічних зв’язків, реклами продукції. Сховища різноманітної широковживаної інформації створюються для подальшого використання. Такими сховищами людина можне користуватися при замовленні квитка на літак, при купівлі ліків, при виборі автомобіля, для ознайомлення з новою книжковою продукцією тощо.

Систематизоване сховище даних називається базою даних (БД).

Сучасні бази даних створюються на електронних носіях. Це забезпечує надійне збереження даних і зручні засоби роботи з ними.

Для створення баз даних і маніпулювання ними використовують спеціальні програми — системи управління базами даних (СУБД). Основними функціями СУБД є:

1. уведення даних у базу та їх збереження;
2. редагування даних;
3. пошук даних у базі

Широкого розповсюдження набули СУБД, які зберігають базу даних з інструментами для її опрацювання. Разом з тим, існують інформаційно-пошукові системи, які здатні працювати з різними базами даних. Такі системи здійснюють пошук інформації за запитом користувача.

**2.1.1 Класифікація баз даних**

Відзначимо тільки основні класифікації.

Класифікація БД за моделлю даних:

1. ієрархічні,
2. мережеві,
3. реляційні,
4. об'єктні,
5. об'єктно-орієнтовані,
6. об'єктно-реляційні.

Класифікація БД за технологією фізичного зберігання:

1. БД у вторинній пам'яті (традиційні);
2. БД в оперативній пам'яті (in-memory databases);
3. БД у третинній пам'яті (tertiary databases).

Класифікація БД за вмістом:

1. географічні.
2. історичні.
3. наукові.
4. мультимедійні.

Класифікація БД за ступенем розподіленості:

1. централізовані (зосереджені);
2. розподілені.

Окреме місце в теорії та практиці займають просторові (англ. spatial), тимчасові, або темпоральні (temporal) і просторово-часові (spatial-temporal) БД.

Ієрархічні бази даних можуть бути представлені як дерево, що складається з об'єктів різних рівнів. Верхній рівень займає один об'єкт, другий - об'єкти другого рівня і т.д.

Між об'єктами існують зв'язки, кожен об'єкт може включати в себе декілька об'єктів більш низького рівня. Такі об'єкти перебувають у відношенні предка (об'єкт більш близький до кореня) до нащадка (об'єкт більш низького рівня), при цьому можлива ситуація, коли об'єкт-предок не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок. Об'єкти, що мають загального предка, називаються близнюками.

**2.1.2 Області застосування баз даних**

Історично системи управління базами даних орієнтувалися на вирішення завдань, пов'язаних у першу чергу з транзакційною обробкою структурованої інформації. Безумовно, найкращим, перевіреним часом рішенням тут була і залишається реляційна модель СУБД. Однак в останні роки область застосування баз даних незмінно розширювалася. З одного боку, потрібно керувати більш широким набором форматів даних, переходячи до вирішення спільних проблем управління корпоративною інформацією. З іншого - саме СУБД беруть на себе основні функції інтеграції даних і додатків корпоративних систем. (За даними Gartner Group, інформаційні відділи підприємств витрачають до 40% свого бюджету на вирішення завдань інтеграції діючих компонентів баз даних.) Саме цим пояснюється активний інтерес до обговорення архітектурних принципів і можливостей реалізації баз даних різних моделей - постреляційних, об'єктно-реляційних, XML.

Якщо постаратися класифікувати існуючі області застосування баз даних, а так само оцінити перспективи їхнього розвитку в даний час, то можна отримати приблизний список найбільш поширених класів:

1. документографічні й документальні застосовуються у всіх базах органів влади та управління;
2. бази даних з промислової, будівельної та сільськогосподарської продукції;
3. бази даних з економічної та кон'юнктурної інформації (статистична, кредитно-фінансова, зовнішньоторговельна);
4. фактографічні бази соціальних даних, які включають відомості про населення і про соціальні середовища;
5. бази даних транспортних систем;
6. довідкові дані для населення та установ (енциклопедії та довідники, розклади літаків і поїздів, адреси та телефони громадян і організацій);
7. ресурсні бази даних, що включають фактографічну інформацію про природні ресурси (земля, вода, надра, біоресурси, гідрометеорологія, вторинні ресурси і відходи, екологічна обстановка);
8. фактографічні бази і банки наукових даних, щоб забезпечити фундаментальні наукові дослідження;
9. фактографічні бази даних у галузі культури і мистецтва;
10. лінгвістичні бази даних, тобто машинні словники різного типу і призначення.

Останнім часом утворилися нові важливі області застосування баз даних, і кожна з них представляє принципово нове середовище, до якого необхідно адаптувати технології СУБД. Ці області отримали на ринку назви інтелектуально аналізу даних (data mining), сховищ даних (data warehousing), репозитаріїв даних (data repository).

**2.1.3 Основні відомості про середовище розробки**

**2.1.3.1 Microsoft Visual Studio**

Microsoft Visual Studio — серія продуктів фірми [Майкрософт](https://uk.wikipedia.org/wiki/Microsoft), які включають [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як [консольні програми](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), так і програми з [графічним інтерфейсом](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), в тому числі з підтримкою технології [Windows Forms](https://uk.wikipedia.org/wiki/Windows_Forms), а також [веб-сайти](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), [веб-застосунки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), [веб-служби](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) як в [рідному](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4), так і в [керованому](https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4)кодах для всіх платформ, що підтримуються [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](https://uk.wikipedia.org/wiki/Windows_Mobile), [Windows Phone](https://uk.wikipedia.org/wiki/Windows_Phone_8), [Windows CE](https://uk.wikipedia.org/wiki/Windows_CE), [.NET Framework](https://uk.wikipedia.org/wiki/.NET_Framework), [.NET Compact Framework](https://uk.wikipedia.org/wiki/.NET_Compact_Framework) та [Microsoft Silverlight](https://uk.wikipedia.org/wiki/Microsoft_Silverlight). Microsoft Visual Studio — серія продуктів фірми [Майкрософт](https://uk.wikipedia.org/wiki/Microsoft), які включають [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як [консольні програми](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), так і програми з [графічним інтерфейсом](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), в тому числі з підтримкою технології [Windows Forms](https://uk.wikipedia.org/wiki/Windows_Forms), а також [веб-сайти](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), [веб-застосунки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), [веб-служби](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) як в [рідному](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4), так і в [керованому](https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4)кодах для всіх платформ, що підтримуються [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](https://uk.wikipedia.org/wiki/Windows_Mobile), [Windows Phone](https://uk.wikipedia.org/wiki/Windows_Phone_8), [Windows CE](https://uk.wikipedia.org/wiki/Windows_CE), [.NET Framework](https://uk.wikipedia.org/wiki/.NET_Framework), [.NET Compact Framework](https://uk.wikipedia.org/wiki/.NET_Compact_Framework) та [Microsoft Silverlight](https://uk.wikipedia.org/wiki/Microsoft_Silverlight).

Visual Studio включає один або декілька з наступних компонентів:

* [Visual Basic .NET](https://uk.wikipedia.org/wiki/Visual_Basic_.NET), а до його появи — [Visual Basic](https://uk.wikipedia.org/wiki/Visual_Basic)
* [Visual C++](https://uk.wikipedia.org/wiki/Microsoft_Visual_C%2B%2B)
* Visual C#
* Visual F# (входить до складу Visual Studio 2010);
* [Visual Studio Debugger](https://uk.wikipedia.org/wiki/Visual_Studio_Debugger)

Багато варіантів постачання також включають:

* [Microsoft SQL Server](https://uk.wikipedia.org/wiki/Microsoft_SQL_Server) або
* MSDE Visual Source Safe — файл-серверна [система управління версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8)

У минулому, до складу Visual Studio також входили продукти:

* Visual InterDev
* Visual J++
* [Visual J#](https://uk.wikipedia.org/wiki/Visual_J_Sharp)
* [Visual FoxPro](https://uk.wikipedia.org/w/index.php?title=Visual_FoxPro&action=edit&redlink=1)
* Visual Source Safe – файл-серверна [система управління версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8).

### Visual Studio 2013

Побачивши світ у вигляді попередьої версії 26 червня 2013 року, нова версія з типовим кодовим ім'ям Dev12 остаточно вийшла у продаж 17 жовтня 2013 року.

Помітними змінами були покращення інтефейсу користувача для командної розробки, підтримка платформи десктопної та мобільної розробки [Windows 8.1](https://uk.wikipedia.org/wiki/Windows_8.1), а також покращення для [Web](https://uk.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%94_%D0%BF%D0%B0%D0%B2%D1%83%D1%82%D0%B8%D0%BD%D0%BD%D1%8F)-розробників. Були перероблені та вдосконалені засоби діагностики та [відлагожування](https://uk.wikipedia.org/wiki/%D0%97%D0%BD%D0%B5%D0%B2%D0%B0%D0%B4%D0%B6%D1%83%D0%B2%D0%B0%D1%87), представлені нові засоби аналізу [витоків пам'яті](https://uk.wikipedia.org/wiki/%D0%92%D0%B8%D1%82%D1%96%D0%BA_%D0%BF%D0%B0%D0%BC%27%D1%8F%D1%82%D1%96) тощо.

Усього вийшло 5 оновлень даної версії, останнє відбулося 20-го липня 2015-го року.

### Visual Studio 2015

Наступна версія Visual Studio, під кодовим ім'ям Dev14 була представлена 20 червня 2015 року. Суттєвою зміною стала [підтримка багатьох цільових платформ](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%BD%D1%96%D1%81%D1%82%D1%8C): окрім базової Windows з'явилась можливість будувати проекти для [IOS](https://uk.wikipedia.org/wiki/IOS) та [Android](https://uk.wikipedia.org/wiki/Android). Для розробників комп'ютерних ігор була додана підтримка фреймворку [Unity](https://uk.wikipedia.org/wiki/Unity_(%D1%80%D1%83%D1%88%D1%96%D0%B9_%D0%B3%D1%80%D0%B8)). Був оновлений механізм [автентифікації](https://uk.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F): користувач під час запуску Visual Studio синхронізується з єдиним аккаунтом Microsoft.

Версія включає в собі [.NET Framework](https://uk.wikipedia.org/wiki/.NET_Framework) 4.6 та підтримку універсальної платформи [Windows 10](https://uk.wikipedia.org/wiki/Windows_10). Розробників на мові [C++](https://uk.wikipedia.org/wiki/C%2B%2B) потішили новим функціоналом стандарту [C++14](https://uk.wikipedia.org/wiki/C%2B%2B14), та навіть деякими поліпшеннями з [C++17](https://uk.wikipedia.org/wiki/C%2B%2B17).

Останнім оновленням на данний момент є Update 2 від 30 березня 2016-го року, у якому багато уваги приділено стабільності, та продовжено роботу у напрямку підтримки нових стандартів мови С++.

### Visual Studio 2017

Перший нестабільний випуск наступної версії програми під умовною назвою "15" побачив світ 30 березня 2016 року. Основними змінами стали інтерфейс інсталятора та численні незначні покращення у різних компонентах середовища розробки. Очікується підтримка мови програмування [Solidity](https://uk.wikipedia.org/wiki/Solidity).

Версія остаточно побачила світ під назвою Visual Studio 2017 7-го березня 2017-го року.

Visual Studio побудована в архітектурі, що підтримує можливість використання доповнень (Add-Ins), — плагінів від сторонніх розробників, що дозволяє розширювати можливості середовища розробки.

Деякі з найпопулярніших доповнень:

* DevPartner Studio
* Visual Assist
* [ReSharper](https://uk.wikipedia.org/wiki/ReSharper)
* [IncrediBuild](https://uk.wikipedia.org/w/index.php?title=IncrediBuild&action=edit&redlink=1)
* Workspace Whiz
* Viva64

**2.1.3.2 Microsoft SQL Server**

**Microsoft SQL Server** — комерційна [система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), що розповсюджується корпорацією [Microsoft](https://uk.wikipedia.org/wiki/Microsoft). Мова, що використовується для запитів — [Transact-SQL](https://uk.wikipedia.org/wiki/Transact-SQL), створена спільно Microsoft та [Sybase](https://uk.wikipedia.org/wiki/Sybase). Transact-SQL є реалізацією стандарту [ANSI](https://uk.wikipedia.org/wiki/%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B0%D0%BD%D1%81%D1%8C%D0%BA%D0%B8%D0%B9_%D0%BD%D0%B0%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%81%D1%82%D0%B8%D1%82%D1%83%D1%82_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D1%96%D0%B2) / [ISO](https://uk.wikipedia.org/wiki/%D0%9C%D1%96%D0%B6%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%B0_%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B7%D1%96_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D1%96%D1%97) щодо структурованої мови запитів [SQL](https://uk.wikipedia.org/wiki/SQL) із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

### Походження

Базовий код MS SQL Server (до версії 7.0) ґрунтувався на коді [Sybase SQL Server](https://uk.wikipedia.org/w/index.php?title=Sybase_SQL_Server&action=edit&redlink=1). Це дозволило [Microsoft](https://uk.wikipedia.org/wiki/Microsoft) вийти на ринок баз даних для підприємств, де конкурували [Oracle](https://uk.wikipedia.org/wiki/Oracle_Corporation), [IBM](https://uk.wikipedia.org/wiki/IBM), і, пізніше, сама Sybase. Microsoft, Sybase і [Ashton-Tate](https://uk.wikipedia.org/w/index.php?title=Ashton-Tate&action=edit&redlink=1) спочатку об'єдналися для створення і випуску на ринок першої версії програми, що отримала назву SQL Server 1.0 для [OS/2](https://uk.wikipedia.org/wiki/OS/2) (близько [1989 року](https://uk.wikipedia.org/wiki/1989)), яка фактично була еквівалентом Sybase SQL Server 3.0 для [Unix](https://uk.wikipedia.org/wiki/Unix), [VMS](https://uk.wikipedia.org/w/index.php?title=VMS&action=edit&redlink=1) та ін. Microsoft SQL Server 4.2 був випущений у [1992 році](https://uk.wikipedia.org/wiki/1992) та входив до складу операційної системи [Microsoft OS/2](https://uk.wikipedia.org/w/index.php?title=Microsoft_OS/2&action=edit&redlink=1) версії 1.3. Офіційний реліз Microsoft SQL Server версії 4.21 для ОС [Windows NT](https://uk.wikipedia.org/wiki/Windows_NT) відбувся одночасно з релізом самої [Windows NT](https://uk.wikipedia.org/wiki/Windows_NT) (версії 3.1). Microsoft SQL Server 6.0 був першою версією SQL Server, створеною виключно для архітектури NT і без участі в процесі розробки Sybase.

До того часу, як вийшла на ринок ОС Windows NT, Sybase і Microsoft розійшлися та створювали вже власні моделі цього програмного продукту. Microsoft намагалася отримати виняткові права на всі версії SQL Server для [Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows). Пізніше Sybase змінила назву свого продукту на [Adaptive Server Enterprise](https://uk.wikipedia.org/w/index.php?title=Adaptive_Server_Enterprise&action=edit&redlink=1) щоб уникнути плутанини з Microsoft SQL Server. До [1994 року](https://uk.wikipedia.org/wiki/1994) Microsoft отримала від [Sybase](https://uk.wikipedia.org/wiki/Sybase) три повідомлення про авторські права як натяк на походження Microsoft SQL Server.

Після розділення компанії зробили декілька самостійних релізів програм. SQL Server 7.0 був першим сервером баз даних зі справжнім [графічним інтерфейсом](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0) адміністрування. Для усунення претензій з боку Sybase у порушенні авторських прав, весь успадкований код в сьомій версії був переписаний. Це забезпечило також й успіх SQL Server 2000, який був першою редакцією, орієнтованою на архітектуру [IA-64](https://uk.wikipedia.org/wiki/IA-64).

Протягом подальших шести років [корпорація Microsoft](https://uk.wikipedia.org/wiki/Microsoft) працювала над вдосконаленням вже існуючої версії SQL Server 2000 доки не збудувала зручнішу систему Microsoft SQL Server 2005. Були вдосконалені продуктивність, кліентські інструменти [інтегрованого середовища розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8), а також у декількох додаткових системах, що встановлюються разом із SQL Server 2005. Змінено: інструментарій [процесів керування сховищами даних](https://uk.wikipedia.org/wiki/ETL) (SQL Server Integration Services або [SSIS](https://uk.wikipedia.org/wiki/SQL_Server_Integration_Services)), сервер звітів, сервер [OLAP](https://uk.wikipedia.org/wiki/OLAP) та [інтелекутального аналізу даних](https://uk.wikipedia.org/w/index.php?title=%D0%86%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%83%D1%82%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D1%96%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&action=edit&redlink=1) ([Analysis Services](https://uk.wikipedia.org/w/index.php?title=Microsoft_Analysis_Services&action=edit&redlink=1)), а також декілька технологій повідомлень, особливо Service Broker та Notification Services.

**Функціональність**

Microsoft SQL Server як мову запитів використовує версію [SQL](https://uk.wikipedia.org/wiki/SQL), що отримала назву [Transact-SQL](https://uk.wikipedia.org/wiki/Transact-SQL) (скорочено T-SQL), яка є реалізацією SQL-92 (стандарт [ISO](https://uk.wikipedia.org/wiki/ISO) для SQL) з багатьма розширеннями. T-SQL дозволяє використовувати додатковий синтаксис процедур, що зберігаються і забезпечує підтримку транзакцій (взаємодія бази даних з керуючим застосунком). Microsoft SQL Server та Sybase ASE для взаємодії з мережею використовують протокол рівня застосунка під назвою [Tabular Data Stream](https://uk.wikipedia.org/w/index.php?title=Tabular_Data_Stream&action=edit&redlink=1) (TDS, протокол передачі табличних даних).

Microsoft SQL Server також підтримує [Open Database Connectivity](https://uk.wikipedia.org/wiki/ODBC) (ODBC)—інтерфейс взаємодії застосунків з СУБД. Версія SQL Server 2005 надає можливість підключення користувачів через веб-сервер-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не призначеним для [Windows](https://uk.wikipedia.org/wiki/Windows), кроссплатформенно з'єднуватися з SQL Server. Microsoft також випустила сертифікований драйвер [JDBC](https://uk.wikipedia.org/wiki/JDBC), що дозволяє застосункам під керування [Java](https://uk.wikipedia.org/wiki/Java) (таким як BEA і [IBM Websphere](https://uk.wikipedia.org/wiki/IBM_Websphere)) з'єднуватися з Microsoft SQL Server 2000 і 2005.

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL—це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами. Усі сервери мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

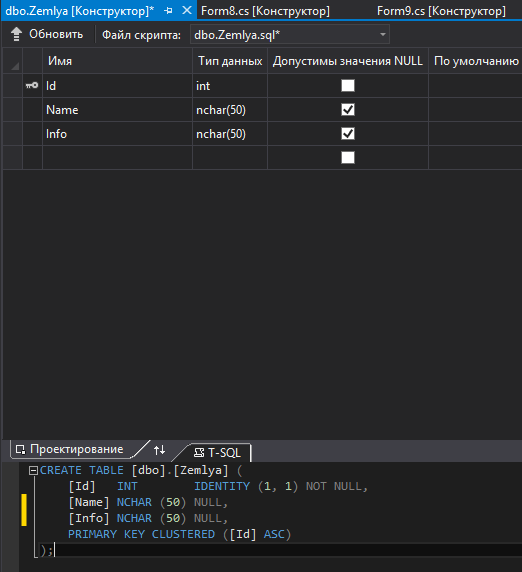
* Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.
* Історія змін: Всі зміни бази даних безперервно передаються користувачам.
* [Синхронізація](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

SQL Server 2005 має вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework. На відміну від інших процесів, .NET Framework виділяє додаткову пам'ять і будує засоби керування SQL Server, не використовуючи вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання у структурах SQL Server.

**2.3 Структура таблиці**

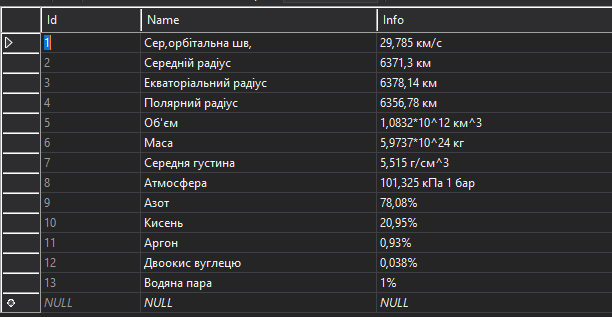
В таблиця Zemlya зберігаються такі стовбці як:

* 1. Id - тип INT, є первинним ключем;
  2. Name – тип NCHAR(50), може бути NULL;
  3. Info – тип NCHAR(50), може бути NULL;

****

**Рис1.Таблиця Zemlya**

БД містить інформацію про характеристики планет, яка продемонстрована в скріншоті знизу:

****

**Рис2. Інформація про нашу планету**

**2.4 Охорона праці**

**Вимоги щодо розміщення і планування приміщень для роботи з комп’ютером:** відповідні робочі місця заборонено облаштовувати у підвальних або цокольних приміщеннях будинків. В обладнанні приміщень забороняється використання полімерних матеріалів, що виділяють шкідливі хімічні речовини. Також слід приділити увагу забезпеченню достатнім для здійснення роботи рівнем освітлення (природного та штучного – у темну пору доби) та звукоізоляції. Для регуляції рівня освітлення природним світлом бажано застосовувати жалюзі. Окрім того, у приміщеннях, де здійснюється робота з комп’ютерами, щодня має здійснюватися вологе прибирання з метою недопущення запиленості підлоги та меблів.

Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця операторів (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном), мають бути надійно захищені діелектричними щитками або сітками з метою недопущення потрапляння людини під напругу.

Особливої уваги заслуговують заходи дотримання протипожежної безпеки. Так, у всьому офісі лінії електромережі мають бути забезпечені від виникнення короткого замикання, а також від перепадів мережевої напруги, що може спричинити збої в роботі електронно-обчислювальної техніки. Приміщення (окрім тих, де розташовуються сервери) мають бути оснащені системою автоматичної пожежної сигналізації та вогнегасниками. Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела загоряння внаслідок короткого замикання та перевантаження проводів, обмежувати застосування проводів з легкозаймистою ізоляцією і, за можливості, застосовувати негорючу ізоляцію. У приміщенні, де одночасно експлуатуються понад п'ять комп’ютерів, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення.

**Вимоги щодо організації та обладнання робочих місць:** площа, відведена на одне робоче місце має становити не менше 6 кв. м., а об’єм – не менше 20 куб. м. Конструкція робочого місця повинна забезпечувати підтримання оптимальної робочої пози (тобто такої, яка дозволяє працівникові виконувати роботу з мінімальним напруженням тіла, і яка дозволяє уникнути перевтоми в ході і після закінчення робочого процесу). Раціональна робоча поза має важливе значення для збереження здоров’я працівника, оскільки тривале перебування його в незручній і напруженій позі може призвести до таких захворювань, як сколіоз (викривлення хребта), варикозне розширення вен, плоскостопість тощо. Установлено, що робота в зігнутому положенні збільшує затрати енергії на 20%, а при значному нахиленні — на 45% порівняно з прямим положенням корпуса.

За потреби особливої концентрації уваги під час виконання робіт суміжні робочі місця операторів необхідно відділяти одне від одного перегородками висотою 1,5 - 2 м.

Робочі місця слід розташовувати відносно джерела природного світла (вікон) таким чином, щоб світло падало збоку, переважно зліва. Також робоче місце має відповідати сучасним вимогам ергономіки:

- стіл повинен мати висоту поверхні 680 - 800 мм., ширину 600 - 1400 мм. і глибину 800 - 1000 мм. (такі параметри забезпечують можливість виконання операцій в зоні досяжності працівника);

- робочий стілець робочий стілець має бути підйомно-поворотним, з можливістю регулювання висоти, бажано зі стаціонарними або змінними підлікотниками і напівм’якою нековзкою поверхнею сидіння, що легко чиститься і не електризується;

- екран комп’ютера має розташовуватися на оптимальній відстані від користувача, що становить 600 – 700 мм., але не менше за 600 мм. з урахуванням літерно-цифрових знаків і символів.

**Вимоги безпеки під час роботи з комп’ютером:**

Щодня перед початком роботи оператор повинен:

- оглянути своє робоче місце; про виявлення ознак пошкодження обладнання інформувати свого безпосереднього керівника;

- відрегулювати освітленість на робочому місці, переконатися в відсутності відблисків на екрані комп’ютера, відсутності зустрічного світла;

- перевірити правильність підключення обладнання ЕОМ до електромережі;

- очистити екран комп’ютера від пилу та інших забруднень;

- перевірити правильність організації робочого місця й за необхідності провести відповідні коригування.

Оператор під час роботи зобов'язаний:

- виконувати тільки ту роботу, яку йому було доручено;

- підтримувати порядок і чистоту на робочому місці;

- тримати відкритими всі вентиляційні отвори обладнання;

- коректно закрити всі активні завдання у разі припинення роботи з комп’ютером;

- негайно відключити комп’ютером від електричної мережі у разі виникнення аварійної ситуації.

У ході виконання робіт оператор комп’ютера повинен:

- витримувати відстань від очей до екрана комп’ютером в межах 60 - 70см;

- дотримуватися внутрішньозмінного режиму праці та відпочинку, регламентованих перерв у роботі, а саме (при 8-годинній денній робочій зміні):

- для розробників програм - тривалістю 15 хвилин через кожну годину роботи;

- для інших категорій працівників - тривалістю 15 хвилин через кожні дві години роботи;

- для операторів комп'ютерного набору - тривалістю 10 хвилин, після кожної години роботи.

Під час регламентованих перерв рекомендується виконувати комплекси вправ для очей, рук, хребта, поліпшення мозкового кровообігу тощо. Про виявлення несправності обладнання або інших факторів, які створюють загрозу для життя або здоров'я працівників, необхідно негайно інформувати свого безпосереднього керівника.

Не допускається:

- виконання ремонту та налагодження комп’ютерої техніки безпосередньо на робочому місці оператора;

- зберігання біля комп’ютера паперу, дискет, інших носіїв інформації, запасних блоків, деталей тощо, якщо вони не використовуються для поточної роботи;

- відключення захисних пристроїв, самочинні зміни в конструкції комп’ютера;

- використання комп’ютерів, на екранах яких під час роботи з'являються нехарактерні сигнали, нестабільне зображення на екрані тощо;

- доторкання до задньої панелі системного блоку при включеному живленні;

- вимикання живлення під час виконання активного завдання;

- попадання вологи на поверхню системного блоку, монітора, клавіатури, дисководів, принтерів та інших пристроїв;

- приймання напоїв та їжі на робочому місці.

Після закінчення роботи з використанням необхідно дотримуватися такої послідовності вимикання обладнання:

- закрити всі активні завдання;

- переконатися у відсутності дискет та дисків у дисководах;

- використавши опцію "Завершення роботи" у меню "Пуск", вимкнути живлення системного блоку;

- вимкнути живлення всіх комп’ютерів;

- вимкнути блок аварійного живлення (за наявності);

- відключити комп’ютер від електромережі, при цьому забороняється тягнути штепсельну вилку за дріт.

У випадку виникнення аварійної ситуації оператор зобов'язаний:

- у всіх випадках виявлення пошкодження проводів електричного живлення, несправності заземлення та інших пошкодженнях електрообладнання, виникненні запаху гарі, диму - негайно вимкнути електричне живлення і повідомити про аварійну ситуацію свого безпосереднього керівника й чергового електрика;

- при попаданні людини під електричну напругу негайно звільнити її від дії струму шляхом вимкнення електричного живлення, до прибуття лікаря надати потерпілому долікарську медичну допомогу;

- при будь-яких випадках порушень роботи технічного обладнання або програмного забезпечення негайно викликати представника технічної служби з питань експлуатації обчислювальної техніки;

- у випадку виникнення різі в очах, різкого погіршення зору, виникнення головного болю, больових відчуттів у пальцях та кистях рук, посилення серцебиття - негайно припинити роботу з використанням ЕОМ, повідомити про те, що сталося, свого безпосереднього керівника й звернутися до медичної установи;

- при загорянні обладнання негайно відключити його від електромережі;

- про загорання повідомити свого безпосереднього керівника, оперативного чергового, пожежну службу; ужити заходів щодо ліквідації вогню за допомогою вуглекислотного або порошкового вогнегасника.

# **3.ВИСНОВОК**

Під час роботи над курсовою роботою я на практиці застосував свої знання з об’єктно-орієнтованого програмування та запитами в базі даних. При використанні додаткової бібліотеки SFML я навчився працювати з об’єктами, спрайтами, текстурами та підключенням бази даних для ще кращого подання інформації користувачеві.

Основні ідеї мого підходу спираються на наступні положення:

* програма являє собою модель реального процесу, частини реального світу.
* модель реального світу або її частини може бути описана як сукупність взаємодіючих між собою об'єктів.
* об'єкт описується набором параметрів, значення яких визначають стан об'єкта, і набором операцій, які може виконувати об'єкт.
* інформація яка добавлена у базу даних може використовуватися користувачем у своїх цілях.
* таблиці у базі даних легко корегуються користувачем

Поставлена задача виконана.

# **Список використаних джерел**

1. Войтович, Володимир Кіндратович. Принципово нова гіпотеза утворення та розвитку Сонячної системи – Л. ДУЛП, 1995. – 40 с
2. Застосування ПЗЗ- методів для досліджень тіл Сонячної системи 2000. – 112 с.: іл.
3. Ходько А. Є. Виноградова М. Г. Від атома водню до Сонячної системи або основи нової космогонічної теорії. СПб.: Вид-во «Надра», 1996.
4. Міессеров К. Г. Новий погляд на освіту Сонячної системи і еволюцію Всесвіту. М.: Изд-во «Машинобудування», 1993.
5. Брандт Дж. Ходж П. Астрофізика Сонячної системи. М.: Изд-во «Світ», 1999.
6. Ламзин С. А., Сурдин В. Г. [Протозвёзды. Где, как и из чего формируются звёзды](http://www.astronet.ru/db/msg/1190949/index.html). – М. : Наука, 1992.
7. Верхні планети [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D1%80%D1%85%D0%BD%D1%96_%D0%BF%D0%BB%D0%B0%D0%BD%D0%B5%D1%82%D0%B8>
8. Об’єктно- орієнтоване програмування [Електронний ресурс]. –

Режим доступу: <http://programming.in.ua/programming/basisprogramming/25-oop.html>

9. Соболева Н.В. Методические указания по оформлению курсовых работ, курсовых и дипломных проектов. – Ижевский государственный технический университет, 2004.

10. Т. Тиори, Дж. Фрай. Проектирование структур баз данных, т. 1,- М.: Мир, 1985.

11. Горев А., Ахаян Р., Макашарипов С. Эффективная работа с СУБД-СПб.: Питер, 1997.

12. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных: Учебник для высших учебных заведений/ Под ред. Проф. А.Д. Хомоненко. – СПб: КОРОНА принт, 2000.

**ДОДАТКИ**

**Додаток А.**

window.isOpen()

true

Event k;

false

window.pollEvent(k)

true

window.close();

Solncesprite.rotate(0.0194);

Solncesprite.setOrigin(100 / 2, 100 / 2);

double MERKURII = (pow((54224.125 / 58)\*1.57, 0.5))/100;

Plunet1\_1sprite.rotate(MERKURII);

Plunet1\_1sprite.setOrigin(4/2,4/2);

Plunet1\_1sprite.setOrigin(60, 60);

window.clear();window.draw(herosprite); window.draw(Solncesprite);window.display();

Texture herotexture;

herotexture.loadFromFile("шлях до файлу");

Sprite herosprite;

herosprite.setTexture(herotexture);

herosprite.setPosition(х,у);

RenderWindow window(VideoMode(1366, 768), "назва вікна");

Texture Solncetextura;

Solncetextura.loadFromFile(" шлях до файлу ");

Sprite Solncesprite;

Solncesprite.setTexture(Solncetextura);

Solncesprite.setTextureRect(IntRect(х1,у1,х2,у2));

Solncesprite.setPosition(х,у);

false

k.type == Event::Closed

**Додаток В.**

Блок схема 1. Симулятор

Користувач вибирає планету

Інформація про вибрану планету

Корегує

Не корегує

Користувачу надається можливість змінення інформації про дану планету

Користувач додає, змінює або видаляє інформацію

Блок схема 2. База даних планет

**Додаток В**

**Лістинг програми симулятора**.

#include <SFML/Graphics.hpp>

#include <iostream>

#include <locale.h>

#include <iomanip>

#include <stdio.h>

#include <time.h>

#include <cmath>

using namespace sf;

int main()

{setlocale(LC\_ALL, "Russian");

Texture herotexture; //Створюється текстура заднього фону

herotexture.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/004.jpg"); //привласнення текстурі задній фон

Sprite herosprite;//Створюється спрайт заднього фону

herosprite.setTexture(herotexture);//нашу текстуру присвоюємо до спрайту

herosprite.setPosition(0, 0);//задаємо координати спрайту

RenderWindow window(VideoMode(1366, 768), "Simulator of the motion planets");//створюємо вікно з розмірністю вказаної в програмі і присвоюємо їм'я вікна

Texture Solncetextura;//Створюється текстура сонця

Solncetextura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/Solnce.png");//привласнення текстурі задній фон

Sprite Solncesprite;//Створюється спрайт сонця

Solncesprite.setTexture(Solncetextura);//нашу текстуру присвоюємо до спрайту

Solncesprite.setTextureRect(IntRect(0, 0, 100, 100));// розмірність сонця

Solncesprite.setPosition(680, 400);// розташування сонця на формі

Texture Plunet1textura;

Plunet1textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/Merkuriy.png");

Sprite Plunet1\_1sprite;

Plunet1\_1sprite.setTexture(Plunet1textura);

Plunet1\_1sprite.setTextureRect(IntRect(0,0,4,4));

Plunet1\_1sprite.setPosition(665, 400);

Texture Plunet2textura;

Plunet2textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/VENERA.png");

Sprite Plunet2\_1sprite;

Plunet2\_1sprite.setTexture(Plunet2textura);

Plunet2\_1sprite.setTextureRect(IntRect(0, 0, 10, 10));

Plunet2\_1sprite.setPosition(680, 400);

Texture Plunet3textura;

Plunet3textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/earth1.png");

Sprite Plunet3\_1sprite;

Plunet3\_1sprite.setTexture(Plunet3textura);

Plunet3\_1sprite.setTextureRect(IntRect(0, 0, 15, 15));

Plunet3\_1sprite.setPosition(680, 400);

Texture Plunet4textura;

Plunet4textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/mars.png");

Sprite Plunet4\_1sprite;

Plunet4\_1sprite.setTexture(Plunet4textura);

Plunet4\_1sprite.setTextureRect(IntRect(0, 0, 9, 9));

Plunet4\_1sprite.setPosition(680, 400);

Texture Plunet5textura;

Plunet5textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/jupiter.png");

Sprite Plunet5\_1sprite;

Plunet5\_1sprite.setTexture(Plunet5textura);

Plunet5\_1sprite.setTextureRect(IntRect(0, 0, 77, 77));

Plunet5\_1sprite.setPosition(680, 400);

Texture Plunet6textura;

Plunet6textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/Saturn.png");

Sprite Plunet6\_1sprite;

Plunet6\_1sprite.setTexture(Plunet6textura);

Plunet6\_1sprite.setTextureRect(IntRect(0, 0, 65, 65));

Plunet6\_1sprite.setPosition(680, 400);

Texture Plunet7textura;

Plunet7textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/uranus.png");

Sprite Plunet7\_1sprite;

Plunet7\_1sprite.setTexture(Plunet7textura);

Plunet7\_1sprite.setTextureRect(IntRect(0, 0, 40, 40));

Plunet7\_1sprite.setPosition(680, 400);

Texture Plunet8textura;

Plunet8textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/neptune.png");

Sprite Plunet8\_1sprite;

Plunet8\_1sprite.setTexture(Plunet8textura);

Plunet8\_1sprite.setTextureRect(IntRect(0, 0, 35, 35));

Plunet8\_1sprite.setPosition(680, 400);

Texture asteroids4textura;

asteroids4textura.loadFromFile("C:/Users/Артем/Desktop/програма/ConsoleApplication19/foto/asteroids2.png");

Sprite asteroids4\_1sprite;

asteroids4\_1sprite.setTexture(asteroids4textura);

asteroids4\_1sprite.setTextureRect(IntRect(0, 0, 500, 500));

asteroids4\_1sprite.setPosition(680, 400);

while (window.isOpen())//поки відкрита форма

{Event k; //подія у SFML

while (window.pollEvent(k))

{if (k.type == Event::Closed) //треба щоб закрилася форма при натисканні на кнопку виходу

window.close();//вікно закрите}

Solncesprite.rotate(0.0194);// швидкість Сонця по своїй осі

Solncesprite.setOrigin(100 / 2, 100 / 2);// центр сонця

double MERKURII = (pow((54224.125 / 58)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet1\_1sprite.rotate(MERKURII);// швидкість Меркурія по своїй осі

Plunet1\_1sprite.setOrigin(4/2,4/2);// центр Меркурія

Plunet1\_1sprite.setOrigin(60, 60);// оберт планети по заданим координатам

double VENERA = (pow((54224.125 / 108)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet2\_1sprite.rotate(VENERA);// швидкість Венери по своїй осі

Plunet2\_1sprite.setOrigin(10 / 2, 10 / 2);// центр Венери

Plunet2\_1sprite.setOrigin(85, 85);// оберт планети по заданим координатам

double ZEMLYA = (pow((54224.125 / 150)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet3\_1sprite.rotate(ZEMLYA);// швидкість Землі по своїй осі

Plunet3\_1sprite.setOrigin(15 / 2, 15 / 2);// центр Землі

Plunet3\_1sprite.setOrigin(120, 120);// оберт планети по заданим координатам

double MARS = (pow((54224.125 / 228)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet4\_1sprite.rotate(MARS);// швидкість Марсу по своїй осі

Plunet4\_1sprite.setOrigin(16 / 2, 16 / 2);// центр Марса

Plunet4\_1sprite.setOrigin(150, 150);// оберт планети по заданим координатам

double UPITER = (pow((54224.125 / 778)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet5\_1sprite.rotate(UPITER);// швидкість Юпітера по своїй осі

Plunet5\_1sprite.setOrigin(77 / 2, 77 / 2);// центр Юпітера

Plunet5\_1sprite.setOrigin(250, 250);// оберт планети по заданим координатам

double SATURN = (pow((54224.125 / 1426)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet6\_1sprite.rotate(SATURN);// швидкість Сатурна по своїй осі

Plunet6\_1sprite.setOrigin(65 / 2, 65 / 2);// центр Сатурна

Plunet6\_1sprite.setOrigin(340,340);// оберт планети по заданим координатам

double URANUS = (pow((54224.125 / 2869)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet7\_1sprite.rotate(URANUS);// швидкість Урана по своїй осі

Plunet7\_1sprite.setOrigin(40 / 2, 40 / 2);// центр Урана

Plunet7\_1sprite.setOrigin(400, 400);// оберт планети по заданим координатам

double NEPTUN = (pow((54224.125 / 5929)\*1.57, 0.5))/100;//формула розрахунку швидкості планет навколо сонця

Plunet8\_1sprite.rotate(NEPTUN);// швидкість Нептуна по своїй осі

Plunet8\_1sprite.setOrigin(35 / 2, 35 / 2);// центр Нептуна

Plunet8\_1sprite.setOrigin(450, 450);// оберт планети по заданим координатам

asteroids4\_1sprite.rotate(15);// швидкість Пояса астероїдів

asteroids4\_1sprite.setOrigin(500 / 2, 500 / 2);

window.clear();// очистити вікно

window.draw(herosprite);// малює на екрані

window.draw(Solncesprite);// малює на екрані

window.draw(Plunet1\_1sprite);// малює на екрані

window.draw(Plunet2\_1sprite);// малює на екрані

window.draw(Plunet3\_1sprite);// малює на екрані

window.draw(Plunet4\_1sprite);// малює на екрані

window.draw(Plunet5\_1sprite);// малює на екрані

window.draw(Plunet6\_1sprite);// малює на екрані

window.draw(Plunet7\_1sprite);// малює на екрані

window.draw(Plunet8\_1sprite);// малює на екрані

window.draw(asteroids4\_1sprite);// малює на екрані

window.display();// вивід на екран}

return 0;}

**Меню симулятора**

void menu(RenderWindow & window) {

Texture Punct1, Punct2, Punct3, aboutTexture, menuBackground,avtor;

Punct1.loadFromFile("Punct1.jpg");//завантаження файлу

Punct2.loadFromFile("Punct2.jpg");

Punct3.loadFromFile("Punct3.jpg");

aboutTexture.loadFromFile("PUPKI.jpg");

menuBackground.loadFromFile("FON.jpg");

avtor.loadFromFile("AVTOR.jpg");

Sprite menu1(Punct1), menu2(Punct2), menu3(Punct3), about(aboutTexture), menuBg(menuBackground),Avtor(avtor);

bool isMenu = 1;//значення True

int menuNum = 0;

menu1.setPosition(100, 185);//розташування

menu2.setPosition(100, 250);

menu3.setPosition(100, 310);

about.setPosition(450,225);

menuBg.setPosition(0, 0);

Avtor.setPosition(1250, 650);

while (isMenu){

menu1.setColor(Color::White);//присвоєння коліру

menu2.setColor(Color::White);

menu3.setColor(Color::White);

menuNum = 0;

if (IntRect(100, 185, 300, 50).contains(Mouse::getPosition(window))) { menu1.setColor(Color::Blue); menuNum = 1; }//коли наводшьна клавішу вона фарбується у синій колір

if (IntRect(100, 250, 300, 50).contains(Mouse::getPosition(window))) { menu2.setColor(Color::Blue); menuNum = 2; }//коли наводшьна клавішу вона фарбується у синій колір

if (IntRect(95, 310, 95, 50).contains(Mouse::getPosition(window))) { menu3.setColor(Color::Blue); menuNum = 3; }//коли наводшьна клавішу вона фарбується у синій колір

if (Mouse::isButtonPressed(Mouse::Left)){

if (menuNum == 1) isMenu = false;//закриття меню та відкриття симулятора

if (menuNum == 2) { window.draw(about); window.display(); while (!Keyboard::isKeyPressed(Keyboard::Escape)); }//відкриття системних характеристик

if (menuNum == 3) { window.close(); isMenu = false; }//вихід з програмии}

window.draw(menuBg);

window.draw(menu1);

window.draw(menu2);

window.draw(menu3);

window.draw(Avtor);

window.display();}}

**Лістинг програми з базою даних**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace MENU\_PLANETI

{

public partial class Form5 : Form

{

SqlConnection sqlconnection;

public Form5()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private async void Form5\_Load(object sender, EventArgs e)

{

string connectionString = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=E:\курсач\програма\MENU\_PLANETI\MENU\_PLANETI\Database1.mdf;Integrated Security=True";

sqlconnection = new SqlConnection(connectionString);

await sqlconnection.OpenAsync();

SqlDataReader sqlReader = null;

SqlCommand command = new SqlCommand("SELECT \* FROM [Planets]",sqlconnection);

try

{

sqlReader = await command.ExecuteReaderAsync();

while (await sqlReader.ReadAsync())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]) + " " + Convert.ToString(sqlReader["Name"]) + " " +Convert.ToString(sqlReader["Info"]));

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (sqlReader != null)

sqlReader.Close();

}

}

private void вихідToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (sqlconnection != null && sqlconnection.State != ConnectionState.Closed)

sqlconnection.Close();

}

private void Form5\_FormClosing(object sender, FormClosingEventArgs e)

{

if (sqlconnection != null && sqlconnection.State != ConnectionState.Closed)

sqlconnection.Close();

}

private async void button1\_Click(object sender, EventArgs e)

{

if(label7.Visible)

label7.Visible = false;

if(!string.IsNullOrEmpty(textBox1.Text) && !string.IsNullOrWhiteSpace(textBox1.Text) && !string.IsNullOrEmpty(textBox2.Text) && !string.IsNullOrWhiteSpace(textBox2.Text))

{

SqlCommand command = new SqlCommand("INSERT INTO [Planets] (Name,Info) VALUES (@Name,@Info)", sqlconnection);

command.Parameters.AddWithValue("Name",textBox1.Text);

command.Parameters.AddWithValue("Info",textBox2.Text);

await command.ExecuteNonQueryAsync();

}

else

{

label7.Visible = true;

label7.Text = "Поля Name та Info повинні бути заповнені";

}

}

private async void оновитиToolStripMenuItem\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

SqlDataReader sqlReader = null;

SqlCommand command = new SqlCommand("SELECT \* FROM [Planets]", sqlconnection);

try

{

sqlReader = await command.ExecuteReaderAsync();

while (await sqlReader.ReadAsync())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]) + " " + Convert.ToString(sqlReader["Name"]) + " " + Convert.ToString(sqlReader["Info"]));

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (sqlReader != null)

sqlReader.Close();

}

}

private async void button2\_Click(object sender, EventArgs e)

{

if(label8.Visible)

label8.Visible = false;

if (!string.IsNullOrEmpty(textBox5.Text) && !string.IsNullOrWhiteSpace(textBox5.Text)

&& !string.IsNullOrEmpty(textBox4.Text) && !string.IsNullOrWhiteSpace(textBox4.Text)

&& !string.IsNullOrEmpty(textBox3.Text) && !string.IsNullOrWhiteSpace(textBox3.Text))

{

SqlCommand command = new SqlCommand("UPDATE [Planets] SET [Name] = @Name,[Info] = @Info WHERE [Id] = @id",sqlconnection);

command.Parameters.AddWithValue("Id",textBox5.Text);

command.Parameters.AddWithValue("Name",textBox4.Text);

command.Parameters.AddWithValue("Info",textBox3.Text);

await command.ExecuteNonQueryAsync();

}

else if (!string.IsNullOrEmpty(textBox5.Text) && !string.IsNullOrWhiteSpace(textBox5.Text))

{

label8.Visible = true;

label8.Text = "ID повинен бути заповнений";

}

else

{

label8.Visible = true;

label8.Text = "Поля Name та Info повинні бути заповнені";

}

}

private async void button3\_Click(object sender, EventArgs e)

{

if(label9.Visible)

label9.Visible = false;

if (!string.IsNullOrEmpty(textBox6.Text) && !string.IsNullOrWhiteSpace(textBox6.Text) )

{

SqlCommand command = new SqlCommand("DELETE FROM [Planets] WHERE [Id] = @Id",sqlconnection);

command.Parameters.AddWithValue("Id", textBox6.Text);

await command.ExecuteNonQueryAsync();

}

else

{

label9.Visible = true;

label9.Text = "ID повинен бути заповнений";

}

}

}

}

**Додаток С.**

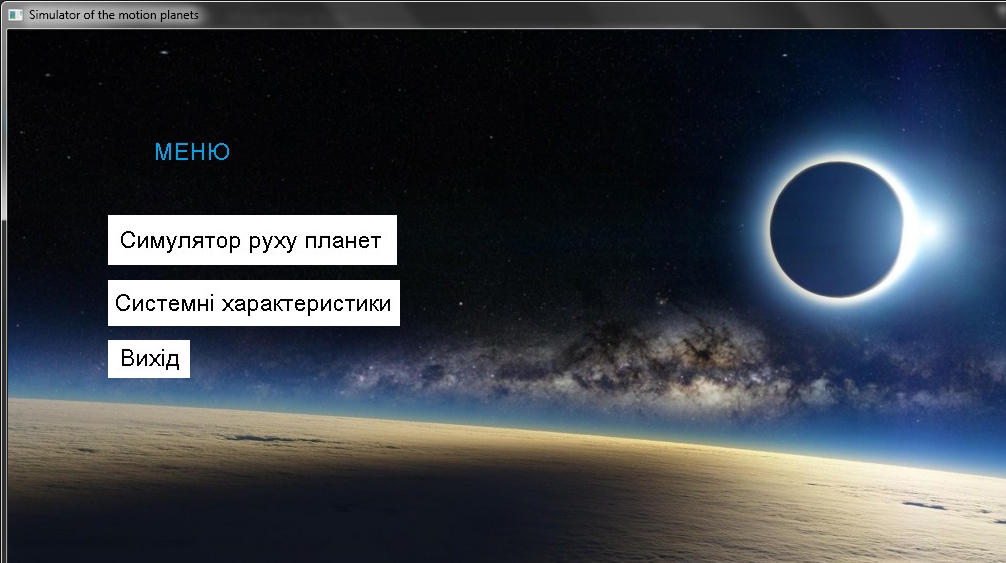
****

Рис. 3Тестовий приклад меню стимулятора.

****

Рис. 4Тестовий приклад симулятора.

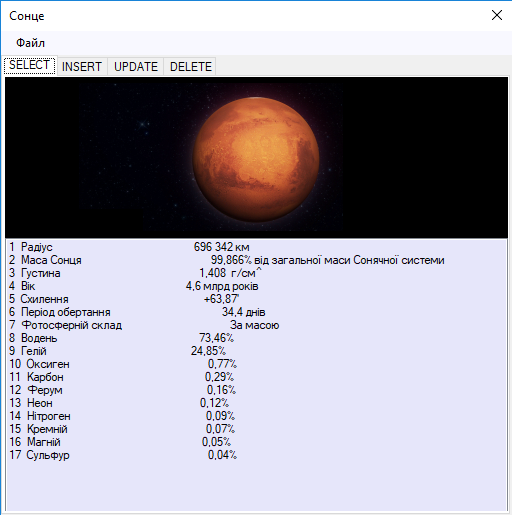
****

Рис. 5Тестовий приклад інформації з бази даних.

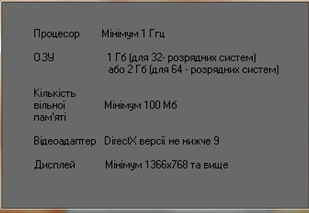
****

Рис. 6Тестовий приклад системних характеристик.



Рис. 7Тестовий приклад автора програми.