

Тема 3: Спадкування та віртуальність.

Мета

Отримати навички розробки власних ієрархій класів із використанням принципу розширення та віртуальності.

1. Загальне завдання

Створити клас `Data2`, використовуючи спадкування від `Data1` із додаванням нових полів. Розподілити та обґрунтувати права доступу до полів.

Виділити базовий клас `BaseView` для `View1` та `View2` із функцією `BaseView::Display()` та наступними віртуальними функціями, котрі викликаються з неї: `protected: BaseView::showHeader(); BaseView::showContent(); BaseView::showFooter();` Перенести основний функціонал відображення в базовий клас, реалізувавши специфічну поведінку у відповідних віртуальних методах. Після вказаних модифікацій `View1` та `View2` повинні працювати аналогічно, як у роботі №2 із об'єктами класу `Data1` та нащадками. Створити клас `View3`, котрий задає специфіку відображення для об'єктів `Data2`. Вибрати необхідне місце у ієрархії для цього класу. Показати роботу віртуальності на прикладі використання нащадка через покажчик на базовий клас для об'єкту `Data1` та об'єкту `Data2` із використанням `View`-класів. Предметна область: Мишки Назва `Data2`: Мишка комп'ютерна Поля `Data2`:

- інтерфейс з'єднання
- тип сенсора

2. Розробка програми

2.1. Засоби ООП

У розробленій програмі були використані наступні засоби ООП:

- інкапсуляція
- спадкування
- поліморфізм

2.2. Ієрархія та структура класів

На рис. 2.1 наведена ієрархія класів: `BaseView`, `ManipulatorView`, `GraphicalView` and `ComputerManipulatorView`. Та на рис. 2.2 зображена ієрархія класів `Manipulator` та `ComputerManipulator`.

Hierarchy

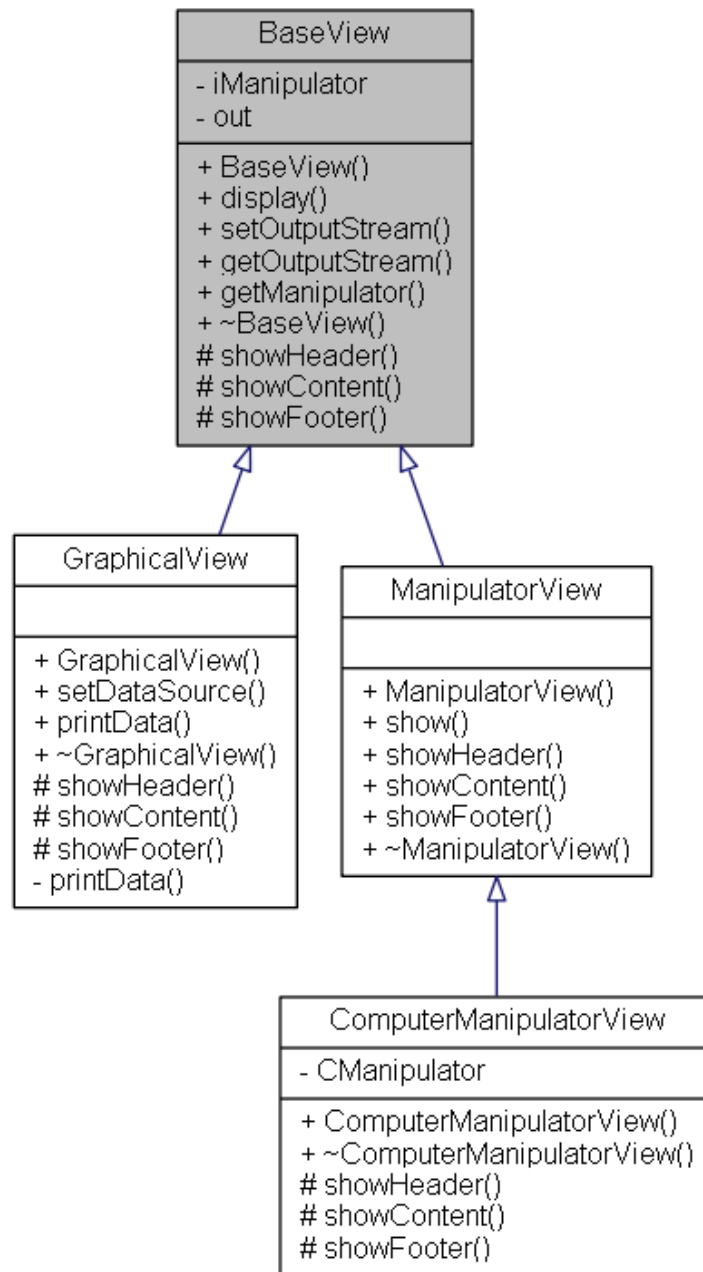
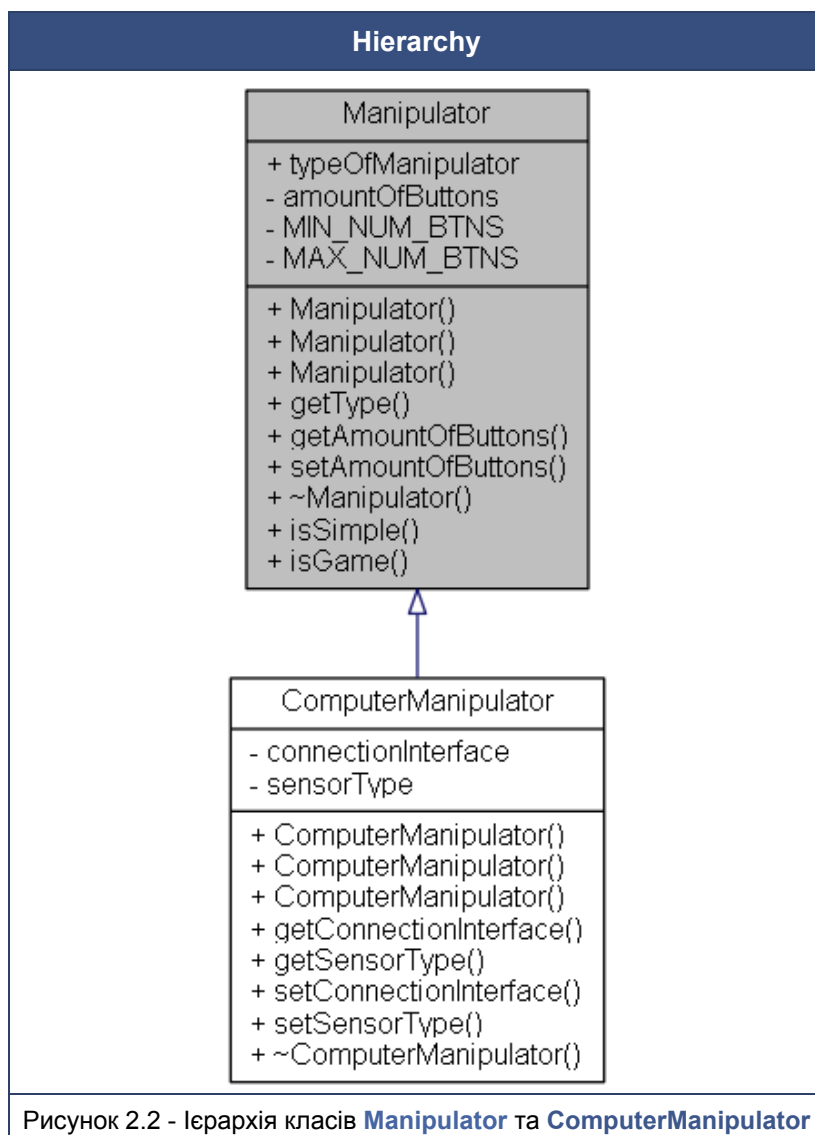
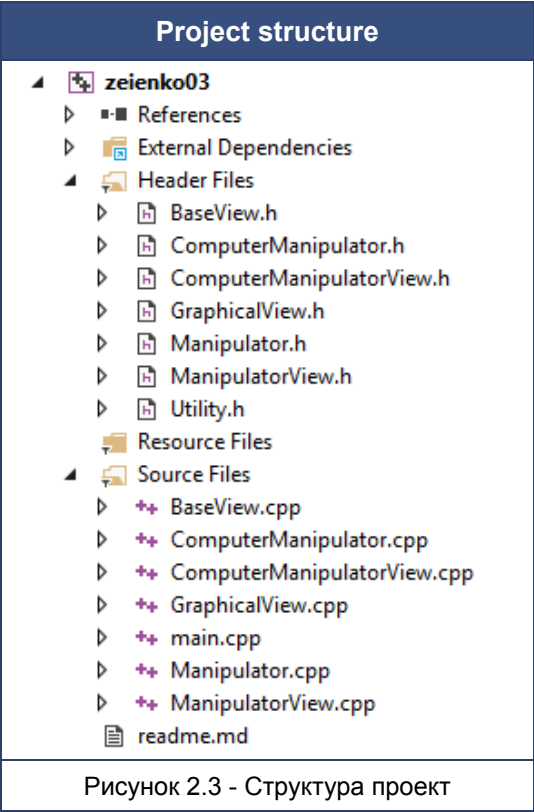


Рисунок 2.1 - Ієрархія класів: **BaseView**, **ManipulatorView**, **GraphicalView** and **ComputerManipulatorView**



2.3. Опис програми

У даній роботі були додані декілька класів: **ComputerManipulator** та **ComputerManipulatorView**, а також був додан новий файл **Utility.h**, у котрому знаходиться функція `instanceof` для перевірки спадкування. На рисунку 2.3 наведена структура розробленого проекту:



2.4. Важливі фрагменти програми

У розробленій програмі слід зауважити увагу на спрацьбовуванні віртуальних методів. Викликаючи метод **BaseView::display()**, у котрому визиваються перевизначені методи: `showHeader()`, `showContent()`, `showFooter()` у нащадках класа **BaseView**, за допомогою покажчика **BaseView** на посилку об'єкта нижче стоячого класу в ієрархії спадкування буде виводитися інформація про об'єкт чию посилку було передано покажчику **BaseView**. Дане ствердження продемонстроване у п. Результат роботи, рисунок 3.1. На рисунку 2.4 приведений фрагмент функції `main()`.

Function main()	
32	{
33	Manipulator* manipulator = new Manipulator();
34	ComputerManipulator* computerManipulator = new ComputerManipulator();
35	ComputerManipulatorView compManipView(*computerManipulator, &(std::cout));
36	BaseView* bv = &compManipView;
37	bv->display();
38	delete computerManipulator;
39	delete manipulator;
40	}
41	

Рисунок 2.4. – Фрагмент функції main()

Для даного проекту були розроблені модульні тести за допомогою GoogleTest Framework. Ці тести перевіряють працювання окремого модуля класів **Manipulator** та **ComputerManipulator**. Функція визову на виконання всіх тестів зображена на рисунку 2.5

Unit Test main() function

```
126  GTEST_API_ int main(int argc, char **argv) {
127      testing::InitGoogleTest(&argc, argv);
128      const int RUN_ALL_TESTS_CONST = RUN_ALL_TESTS();
129      system("pause");
130      return RUN_ALL_TESTS_CONST;
131  }
```

Рисунок 2.5. – Функція main() GoogleTest Framework

Призначення спроектованих класів наведено на рис. 2.6.

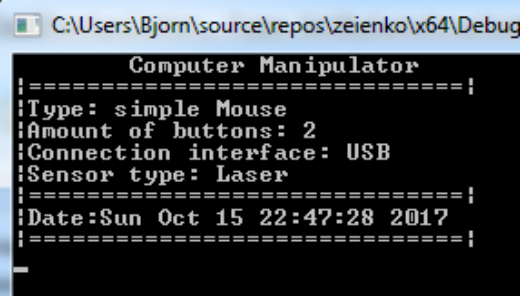
Predestination structure	
Класи	
Класи, структури, об'єднання та інтерфейси з коротким описом.	
C BaseView	The base class for all other classes which are responsible for output
C ComputerManipulator	This class represents computer's manipulators
C ComputerManipulatorView	This class provides output of the CpmputerMnipulator's information
C GraphicalView	Representation of text-pseudo-graphic display of data
C Manipulator	Represents abstraction of mouse periphery
C ManipulatorView	This class provides an output information about manipulator

Рисунок 2.6. – Призначення спроектованих класів

3. Результат работы

Результат роботи програми зображений на рисунку 3.1.

Output to the console



```
Computer Manipulator
=====
!Type: simple Mouse
!Amount of buttons: 2
!Connection interface: USB
!Sensor type: Laser
=====
!Date:Sun Oct 15 22:47:28 2017
=====
```

Рисунок 3.1. – Результат виконання програми

Результат виконання всіх модульних тестів зображений на рисунку 3.2. Тести відсортировані за класом тестування.

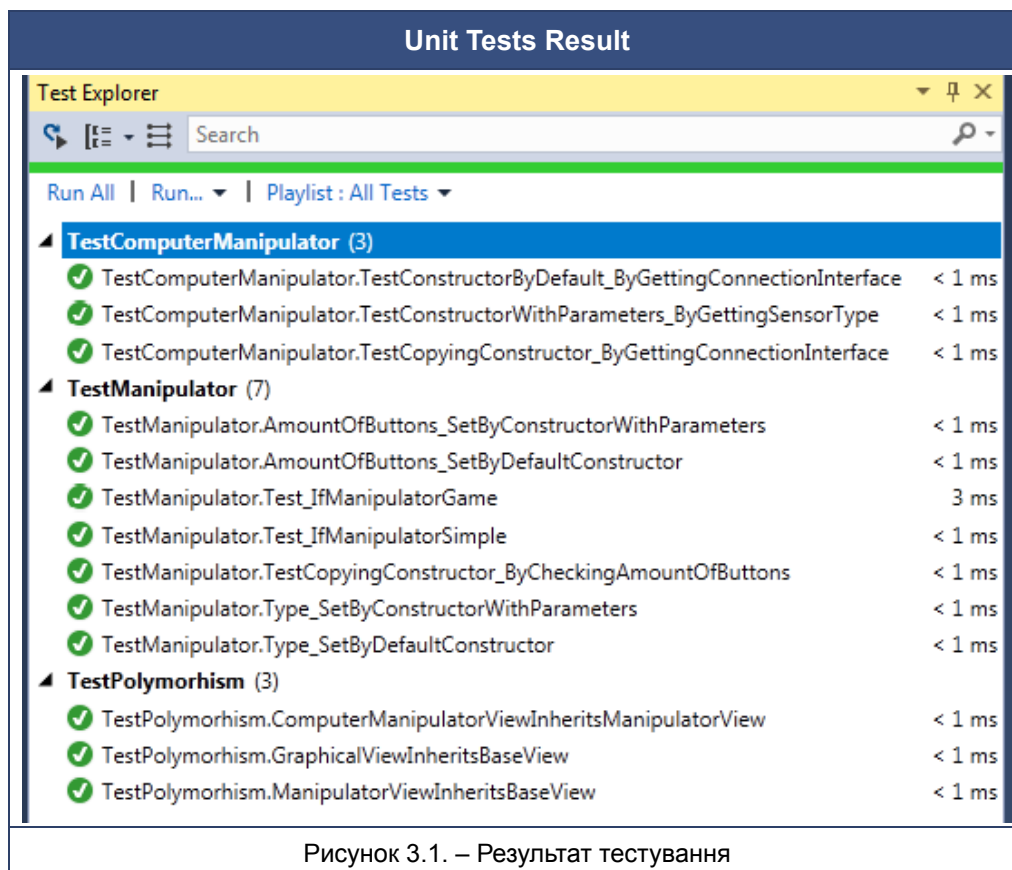


Рисунок 3.1. – Результат тестування

Висновок

В ході виконання лабораторної роботи були отримані навички розробки власних ієрархій класів із використанням принципу розширення та віртуальності.