Тема 3: Стекові об'єкти. Конструктор, деструктор, відображення, передача.

Мета

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

1 ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Створити клас **HousingRoom** використовуючи спадкування від **Room** з додаванням нових полів та класс відображення даних - **HousingRoomScreen**. Виділити базовий клас **BaseView** для **Screen** та **GraphScreen** із функцією відображення та віртуальними методами ShowHeader(), ShowContent(),ShowFooter(). Перенести основний функціонал у базовий клас, реалізувавши спецефічну поведінку у відповідних віртуальних методах. Створити клас **HousingRoomScreen** та вибрати необхідне місце у ієрархії відображень для цього класу. Показати роботу віртуальності на прикладі викоритсання нащадка через покажчик на базовий клас для об'єкту **Room** та **HousingRoom**.

2 РОЗРОБКА ПРОГРАМИ

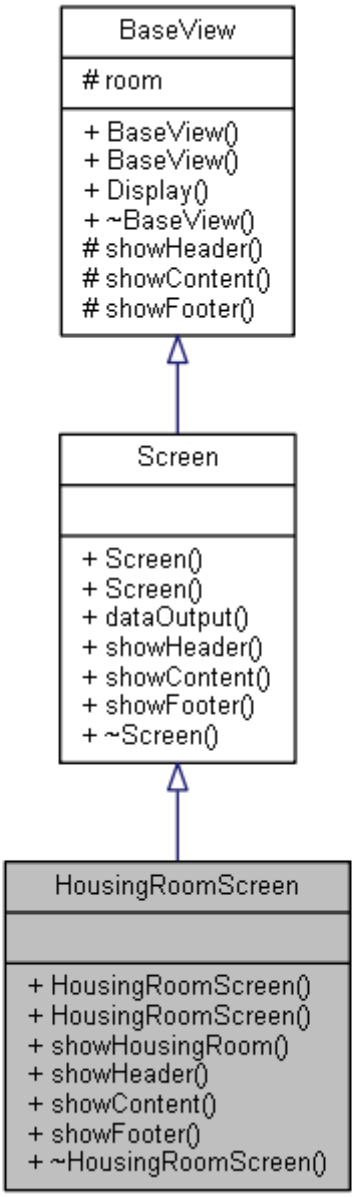
2.1 Засоби ООП

В ході розробки програми були використані такі засоби ООП:

Абстракція – кожен об’єкт описує свою сутність, яка визначається його полями. Спадкування - механізм утворення нових класів на основі використання вже існуючих Інкапсуляція - поля об'єктів закриті для користувача,натомість ми даємо доступ до даних за допомогою геттерів та сеттерів, так користувач має можливість отримати готові дані, а не обробляти їх, для подпльшого вікористання.

2.2 Ієрархія та структура класів

Ієрархія класів наведена на рис. 2.1


BaseView
room
+ BaseView()
+ BaseView()
+ Display()
+ ~BaseView()
# showHeader()
# showContent()
# showFooter()
Screen
+ Screen()
+ Screen()
+ dataOutput()
+ showHeader()
+ showContent()
+ showFooter()
+ ~Screen()
HousingRoomScreen
+ HousingRoomScreen()
+ HousingRoomScreen()
+ showHousingRoom()
+ showHeader()
+ showContent()
+ showFooter()
+ ~HousingRoomScreen()

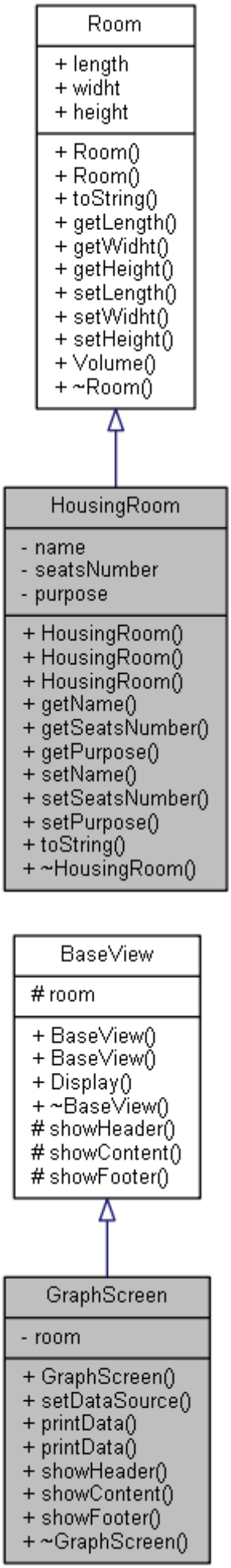


Рисунок 2.1 – Ієрархія класів

2.3 Опис програми

Структура проекту наведена на рис. 2.2.

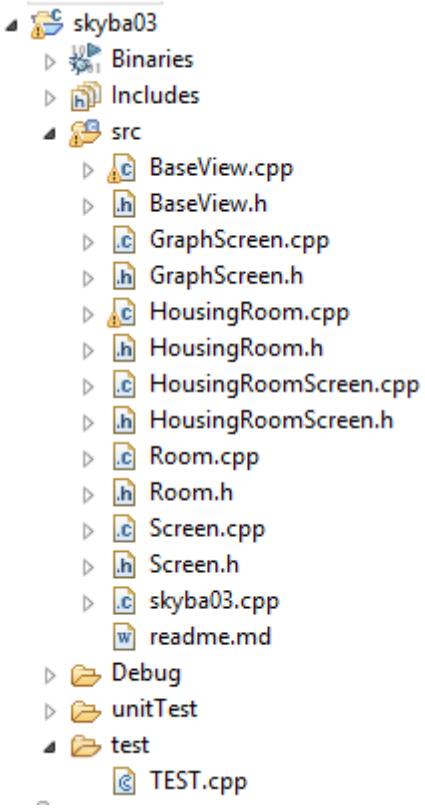


Рисунок 2.2 – Структура проекту

В програмі було створено два класи. Призначення спроектованих класів наведено на рис. 2.3.

Класи, структури, об'єднання та інтерфейси з коротким описом.

BaseView	Class which views information of **Room** object
GraphScreen	Class which views **Room** object with use of pseudo-graphic
HousingRoom	Class containing the implementation of a **HousingRoom**
HousingRoomScreen	Class which views information of Button object
Room	Class containing the implementation of a **Room**
Screen	Class which views information of **Room** object

Рисунок 2.3 – Призначення класів Класс **Room** описує сутність кімнати, а саме розміри кімнати (довжина, ширина, висота). Класс **Screen** використовується для виводу до консолі даних з об'єкту **Room**.

2.4 Важливі фрагменти програми

У программі слід зауважити увагу на таких моментах:

Клас **BaseView.h** функція відображення та віртуальні методи:

```
        //Object to be viewed.
        Room *room;
        /**
        * Outputs header of information.
        */
        virtual void showHeader() = 0;
        /**
         * Outputs main information about room.
        */
        virtual void showContent() = 0;
        /**
        * Outputs footer of information.
        */
        virtual void showFooter() = 0;

      blic:
        /**
        * Default constructor, room set to default.
        */
        BaseView();
        /**
        * Constructor for all fields.
        */
        BaseView(Room *room);
        /**
        *Outputs information about `room` into console.
        */
        void Display();
```

Рисунок 2.4 - Функція відображення та віртуальні методи

```
const float LENGTH1 = 5.05;
const float WIDHT1 = 3.03;
const float HEIGHT1 = 3.57;
Room room1(LENGTH1, WIDHT1, HEIGHT1);

const float LENGTH2 = 8.73;
const float WIDHT2 = 4.19;
const float HEIGHT2 = 3.40;
Room room2(LENGTH2, WIDHT2, HEIGHT2);

const float LENGTH3 = 6.91;
const float WIDHT3 = 5.21;
const float HEIGHT3 = 2.9;
const string NAME1 = "Room #313";
const int SEATSNUMBER1 = 20;
const string PURPOSE1 = "Laboratory room";
HousingRoom housingRoom(LENGTH3, WIDHT3, HEIGHT3, NAME1, SEATSNUMBER1, PURPOSE1);

cout << "From Screen " << endl;

cout << "room1: \n";
Screen view(&room1);
view.Display();

cout << "room2: \n";
Screen view2(&room2);
view2.Display();
```

Рисунок 2.5 - Демонстрація работи функції main

Результати работи

Результати роботи показано на рис. 3.1, 3.2.

```
From Screen
room1:

Screen constructor
Screen::showHeader()
Length: 5.050000
Widht: 3.030000
Height: 3.570000

 Screen::showFooter()
room2:

Screen constructor
Screen::showHeader()
Length: 8.730000
Widht: 4.190000
Height: 3.400000

 Screen::showFooter()

GraphScreen constructor
From Graph Screen
GraphScreen::showHeader()
```

Рисунок 3.1 – Результати роботи

```
[----------] 1 test from classRoom
[ RUN      ] classRoom.constructorsGeters
[       OK ] classRoom.constructorsGeters (0 ms)
[----------] 1 test from classRoom (0 ms total)

[----------] 1 test from classHousingRoom
[ RUN      ] classHousingRoom.constructorsGeters
[       OK ] classHousingRoom.constructorsGeters (0 ms)
[----------] 1 test from classHousingRoom (0 ms total)

[----------] 1 test from toString
[ RUN      ] toString.display

Screen constructor

Screen destructor
[       OK ] toString.display (0 ms)
[----------] 1 test from toString (0 ms total)

[----------] 1 test from toStringHousingRoom
[ RUN      ] toStringHousingRoom.display
[       OK ] toStringHousingRoom.display (0 ms)
[----------] 1 test from toStringHousingRoom (0 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 4 test cases ran. (0 ms total)
[  PASSED  ] 4 tests.
```

Рисунок 3.2 – Результати проходження тестів

Висновок

Отримав основні нивики розробки власних ієрархій класів із використанням принципу розширення та віртуальності.