ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ

за курсом «Програмування»

студент(а/ки) групи ПЗ-24у-1

Кондрачука Олександра Вадимовича

кафедра математичного забезпечення ЕОМ, ДНУ

2024/2025 навч.р.

1. **Патерн Flyweight**

Цей патерн реалізовано за допомогою декількох класів:

1. Клас "ShapeFactory":

Цей клас відповідає за створення класу фігури з бібліотеки SFML. Він є singletone, тому його може використати будь-який об’єкт де завгодно у коді. У ньому є один публічний метод CreateShape, що приймає enum фігури (наразі це коло або прямокутник). Цей метод шукає в уже до цього створених фігур чи була вже така, яку хоче отримати користувач. Якщо так, він повертає створений до цього об’єкт тим часим перевикористовуючи його. Якщо ж ні, метод створює новий об’єкт, запам’ятовує його для перевикористання, та повертає з методу.

static sf::Shape\* CreateShape(const GraphicForm form)

{

if (instance\_->shapes\_.count(form) == 0)

{

sf::Shape\* shape = nullptr;

switch (form) {

case circle:

shape = new sf::CircleShape(20.f);

break;

case rectangle:

shape = new sf::RectangleShape({ 40.0f, 40.0f });

break;

case none:

shape = nullptr;

break;

}

instance\_->shapes\_[form] = shape;

return shape;

}

return instance\_->shapes\_[form];

}

1. Клас "Graphic":

Цей клас є графічним контролером об’єкта. Він відповідаю за відображення, зміну кольору, видимість об’єкта та обведення. Також при зміні фігури, цей клас використовує ShapeFactory та його метод Create.

void set\_form(GraphicForm form)

{

form\_ = form;

shape\_ = ShapeFactory::CreateShape(form);

shape\_->setOutlineThickness(outlineThickness\_);

}

1. **Патерн Query Object**

У моєму випадку патерн "Query Object" застосовується для управління бібліотекою.

1. Клас "Book":

Цей клас є простим об'єктом, що представляє книгу. Він має 3 поля: назву, автора та рік видання.

#pragma once

#include <string>

struct Book

{

std::string title;

std::string author;

int publish\_year;

};

1. Клас "BookRepository":

Цей клас є реалізацією паттерну "Repository". Він містить вектор книг та методи для роботи з ними.

Метод "add\_book":

Додає нову книгу до репозиторію.

void add\_book(const Book& book)

{

cashed\_books\_.push\_back(book);

save();

}

Метод "save":

Зберігає інформацію про книги у файл.

void save() const

{

std::ofstream file(FileName);

if (!file.is\_open()) return;

for (const auto& book : cashed\_books\_) {

file << book.title << '\t' << book.author << '\t' << book.publish\_year << '\n';

}

file.close();

}

Метод "load":

Зчитує інформацію з файлу.

void load() {

std::ifstream file(FileName);

if (!file.is\_open()) return;

cashed\_books\_.clear();

std::string line;

while (std::getline(file, line))

{

std::istringstream iss(line);

Book book;

if (std::getline(iss, book.title, '\t') &&

std::getline(iss, book.author, '\t') &&

iss >> book.publish\_year)

{

cashed\_books\_.push\_back(book);

}

}

file.close();

}

Метод "find":

Фільтрує книги та повертаю результат базуючись на переданому параметру фільтру.

std::vector<Book> find(BaseBookQuery\* query)

{

std::vector<Book> filtered;

std::copy\_if(cashed\_books\_.begin(), cashed\_books\_.end(), std::back\_inserter(filtered),

[query](const Book& book) {

return query->matches(book);

});

return filtered;

}

1. Клас "BaseBookQuery":

Цей клас є базовим класом для усіх Query Object, у нбого є один чисто віртульний метод, що приймає книгу та повертає, чи проходить вона фільтрацію.

#pragma once

#include "Book.h"

class BaseBookQuery

{

public:

virtual bool matches(Book book) = 0;

};

1. Клас "PublishingYearBookRepository":

Цей клас є нащадком класу BaseBookQuery та фільтрує книги за їх роком видання, якщо він співпадає з роком видання, переданим до конструктора класу.

#pragma once

#include "BaseBookQuery.h"

class PublishingYearBookQuery : public BaseBookQuery

{

int year\_;

public:

PublishingYearBookQuery(const int year)

{

year\_ = year;

}

bool matches(const Book book) override

{

return book.publish\_year == year\_;

}

};

Функція "main":

У функції "main" створюється репозиторій та завантажує дані з файли. Потім створюється екземпляр об’єкту PublishingYearBookQuery з параметром року видання, по якому буде йти фільтрація книг. Далі книги фільтруються по Query та результат виводиться на екран.

Цей підхід дозволяє легко абстрактно та інкапсульовано додавати нові способи фільтрацію. Патерн допомагає зберегти бізнес-логіку відокремленою від деталей роботи з даними, що полегшує розширення та підтримку програмного забезпечення.

#include <iostream>

#include "BookRepository.h"

#include "PublishingYearBookQuery.h"

int main()

{

BookRepository repository;

repository.load();

BaseBookQuery\* query = new PublishingYearBookQuery(222);

for (const auto& book : repository.find(query))

{

std::cout << book.title << ' ' << book.author << ' ' << book.publish\_year << '\n';

}

}