Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

**ЛАБОРАТОРНА РОБОТА №6**

**Тема:** Шаблони в С++

з дисципліни: Об’єктно-орієнтоване програмування

|  |
| --- |
| Виконав: ст. гр. КН-23 |
| Царенко С. О. |
| Перевірив: асистент  Козірова Н. Л. |
|  |

Кропивницький

2024

**Тема:** Шаблони в С++

**Мета:** Ознайомитись з основними поняттями шаблони та навчитись їх програмно реалізовувати мовою С++.

**Варіант 5**

***Завдання 1***

Що ж, вітаю гра «Спіймай муху», доходить до свого логічного завершення. На даному етапі, вашій мусі нудно літати по пустій платформі, створіть для неї інтерфейс кімнати, в кімнаті уже є вікно. Також на другому рівні, має залишитись муха з першого рівня і прилетіти нова. Коли на другому рівні одна з мух вилітає у вікно або ловиться в пастку, настає третій рівень, на третьому рівні через відкриті двері до нас заскакує клас «Жаба», котра намагається спіймати муху, при цьому вікно закривається, а об’єкт «пастка» знищується . Додайте лічильник, щоб було цікавіше.

***Завдання 2***

**Варіант 5**

Створи програму для керування рестораном. У ресторані є різні страви, які клієнти можуть замовити. Кожна страва має назву, опис та ціну. Необхідно створити клас Dish, який містить ці дані.

Крім того, створи шаблонний клас Menu, який представляє меню ресторану. Цей клас має методи для додавання страви до меню, видалення страви за назвою, виведення списку всіх страв та пошуку страви за ціною.

Забезпеч універсальність класу Menu, використовуючи шаблонний параметр для типу страви. Це дозволить тобі працювати з різними типами страв (наприклад, страви із м'ясом, страви для вегетаріанців тощо).

Для демонстрації роботи програми:

1. Створи об'єкт Menu для зберігання страв.
2. Додай до меню декілька різних страв.
3. Виведи на екран список всіх страв у меню.
4. Знайди та виведи на екран всі страви за певною ціною.
5. Видали з меню страву за назвою.
6. Знову виведи на екран оновлений список страв у меню.

Не забудь використовувати шаблонний клас Menu для зберігання страв будь-якого типу. Також, забезпеч, щоб клас Dish мав методи доступу до своїх полів (назва, опис, ціна).

**Завдання 1.**

***fly.h:***

#ifndef FLY\_H

#define FLY\_H

#include <QLabel>

#include <QString>

class Fly : public QLabel {

Q\_OBJECT

public:

explicit Fly(QWidget \*parent = nullptr, const QString &imagePath = "");

virtual ~Fly();

virtual void moveFly() = 0;

};

#endif // FLY\_H

***fly.cpp:***

#include "fly.h"

#include <QPixmap>

Fly::Fly(QWidget \*parent, const QString &imagePath) : QLabel(parent) {

setPixmap(QPixmap(imagePath).scaled(80, 80, Qt::KeepAspectRatio));

setAlignment(Qt::AlignCenter);

}

Fly::~Fly() {}

***frog.h:***

#ifndef FROG\_H

#define FROG\_H

#include <QLabel>

#include <QString>

class Frog : public QLabel {

Q\_OBJECT

public:

explicit Frog(QWidget \*parent = nullptr, const QString &imagePath = "");

virtual ~Frog();

void moveFrog(const QPoint &targetPosition);

};

#endif // FROG\_H

***frog.cpp:***

#include "frog.h"

#include <QPixmap>

Frog::Frog(QWidget \*parent, const QString &imagePath) : QLabel(parent) {

setPixmap(QPixmap(imagePath).scaled(80, 80, Qt::KeepAspectRatio));

setAlignment(Qt::AlignCenter);

}

Frog::~Frog() {}

void Frog::moveFrog(const QPoint &targetPosition) {

int x = this->x();

int y = this->y();

int dx = (targetPosition.x() > x) ? 5 : -5;

int dy = (targetPosition.y() > y) ? 5 : -5;

x += dx;

y += dy;

this->move(x, y);

}

***level.h:***

#ifndef LEVEL\_H

#define LEVEL\_H

#include "fly.h"

class Level : public Fly {

Q\_OBJECT

public:

explicit Level(QWidget \*parent = nullptr, const QString &imagePath = "", int level = 1);

void moveFly() override;

private:

int currentLevel;

};

#endif // LEVEL\_H

***level.cpp:***

#include "level.h"

#include <QCursor>

Level::Level(QWidget \*parent, const QString &imagePath, int level)

: Fly(parent, imagePath), currentLevel(level) {}

void Level::moveFly() {

static QPoint prevPos;

QPoint cursorPos = parentWidget()->mapFromGlobal(QCursor::pos());

if (prevPos == cursorPos) {

return;

}

prevPos = cursorPos;

int x = this->x();

int y = this->y();

int dx = (x > cursorPos.x()) ? 10 : -10;

int dy = (y > cursorPos.y()) ? 10 : -10;

switch (currentLevel) {

case 1:

break; // Вправо-вниз

case 2:

dx = -dx;

break; // Вліво-вниз

case 3:

dx = -dx;

dy = -dy;

break; // Вліво-вгору

case 4:

dy = -dy;

break; // Вправо-вгору

case 5:

break;

default:

break;

}

x = std::clamp(x + dx, 0, parentWidget()->width() - this->width());

y = std::clamp(y + dy, 0, parentWidget()->height() - this->height());

this->move(x, y);

}

***Mainwindow.h:***

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QTimer>

#include <QLabel>

#include <QMouseEvent>

#include "level.h"

#include "frog.h"

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow {

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private:

Ui::MainWindow \*ui;

Level \*currentFly;

Level \*secondFly;

Frog \*currentFrog;

QTimer \*moveTimer;

QTimer \*secondMoveTimer;

QLabel \*trap;

QLabel \*window;

int currentLevel;

void moveFly();

void moveFly2();

bool isCaught();

void nextLevel();

protected:

void mouseMoveEvent(QMouseEvent \*event) override;

};

#endif // MAINWINDOW\_H

***Mainwindow.cpp:***

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QMessageBox>

#include <QPixmap>

#include <QCursor>

#include <QTimer>

#include <QDebug>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent), ui(new Ui::MainWindow), currentLevel(1), currentFrog(nullptr), secondFly(nullptr) {

ui->setupUi(this);

setFixedSize(800, 600);

QLabel \*background = new QLabel(this);

background->setGeometry(0, 0, 800, 600);

QPixmap roomPixmap("D:/Стас/Универ/OOP/OOP\_Lab6/Task1/room.jpg");

background->setPixmap(roomPixmap.scaled(800, 600, Qt::IgnoreAspectRatio));

background->lower();

trap = new QLabel(this);

trap->setGeometry(650, 400, 120, 80);

trap->setAlignment(Qt::AlignCenter);

QPixmap trapPixmap("D:/Стас/Универ/OOP/OOP\_Lab6/Task1/trap.jfif");

trapPixmap = trapPixmap.scaled(120, 120, Qt::KeepAspectRatio);

trap->setPixmap(trapPixmap);

window = new QLabel(this);

window->setGeometry(650, 150, 120, 80);

window->setAlignment(Qt::AlignCenter);

QPixmap windowPixmap("D:/Стас/Универ/OOP/OOP\_Lab6/Task1/window.jpg");

windowPixmap = windowPixmap.scaled(120, 120, Qt::KeepAspectRatio);

window->setPixmap(windowPixmap);

moveTimer = new QTimer(this);

connect(moveTimer, &QTimer::timeout, this, &MainWindow::moveFly);

moveTimer->start(90);

secondMoveTimer = new QTimer(this);

connect(secondMoveTimer, &QTimer::timeout, this, &MainWindow::moveFly2);

setMouseTracking(true);

currentFly = new Level(this, "D:/Стас/Универ/OOP/OOP\_Lab6/Task1/fly.png", currentLevel);

currentFly->setGeometry(100, 100, 80, 80);

currentFly->show();

if (currentLevel == 2) {

secondFly = new Level(this, "D:/Стас/Универ/OOP/OOP\_Lab6/Task1/fly2.jpg", currentLevel);

secondFly->setGeometry(200, 100, 80, 80);

secondFly->show();

secondMoveTimer->start(90);

}

}

MainWindow::~MainWindow() {

delete ui;

delete trap;

delete window;

delete currentFly;

delete currentFrog;

delete secondFly;

delete moveTimer;

delete secondMoveTimer;

}

void MainWindow::mouseMoveEvent(QMouseEvent \*event) {

QMainWindow::mouseMoveEvent(event);

}

void MainWindow::moveFly() {

if (currentFly) {

currentFly->moveFly();

}

if (currentFrog) {

currentFrog->moveFrog(currentFly->pos());

}

if (currentFly && currentFly->geometry().intersects(window->geometry())) {

moveTimer->stop();

QMessageBox::information(this, "Гра завершена", "Ви програли! Муха вилетіла в вікно.");

close();

return;

}

if (isCaught()) {

moveTimer->stop();

QMessageBox::information(this, "Гра завершена", "Ви піймали муху!");

nextLevel();

}

if (currentFrog && currentFrog->geometry().intersects(currentFly->geometry())) {

moveTimer->stop();

QMessageBox::information(this, "Гра завершена", "Жаба спіймала муху!");

window->hide();

trap->hide();

close();

}

}

void MainWindow::moveFly2() {

if (secondFly) {

QPoint cursorPos = QCursor::pos();

int dx = (secondFly->x() > cursorPos.x()) ? -10 : 10;

int dy = (secondFly->y() > cursorPos.y()) ? -10 : 10;

int x = std::clamp(secondFly->x() + dx, 0, width() - secondFly->width());

int y = std::clamp(secondFly->y() + dy, 0, height() - secondFly->height());

secondFly->move(x, y);

if (secondFly->geometry().intersects(window->geometry())) {

secondMoveTimer->stop();

QMessageBox::information(this, "Гра завершена", "Друга муха вилетіла в вікно!");

close();

return;

}

if (secondFly->geometry().intersects(trap->geometry())) {

secondMoveTimer->stop();

QMessageBox::information(this, "Гра завершена", "Друга муха потрапила в пастку!");

nextLevel();

return;

}

}

}

bool MainWindow::isCaught() {

return currentFly && currentFly->geometry().intersects(trap->geometry());

}

void MainWindow::nextLevel() {

delete currentFly;

delete currentFrog;

delete secondFly;

currentFrog = nullptr;

secondFly = nullptr;

currentLevel++;

if (currentLevel > 5) {

QMessageBox::information(this, "Гра завершена", "Всі рівні пройдено!");

close();

return;

}

if (currentLevel == 3) {

currentFrog = new Frog(this, "D:/Стас/Универ/OOP/OOP\_Lab6/Task1/frog.jpg");

if (currentFrog) {

currentFrog->setGeometry(650, 300, 80, 80);

currentFrog->show();

} else {

qDebug() << "Failed to create frog object.";

}

}

QString flyImage = QString("D:/Стас/Универ/OOP/OOP\_Lab6/Task1/fly%1.jpg").arg(currentLevel);

currentFly = new Level(this, flyImage, currentLevel);

currentFly->setGeometry(100, 100, 80, 80);

currentFly->show();

if (currentLevel == 2) {

secondFly = new Level(this, "D:/Стас/Универ/OOP/OOP\_Lab6/Task1/fly2.jpg", currentLevel);

secondFly->setGeometry(200, 100, 80, 80);

secondFly->show();

secondMoveTimer->start(90);

}

moveTimer->start(90);

}

***Main.cpp:***

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[]) {

QApplication a(argc, argv);

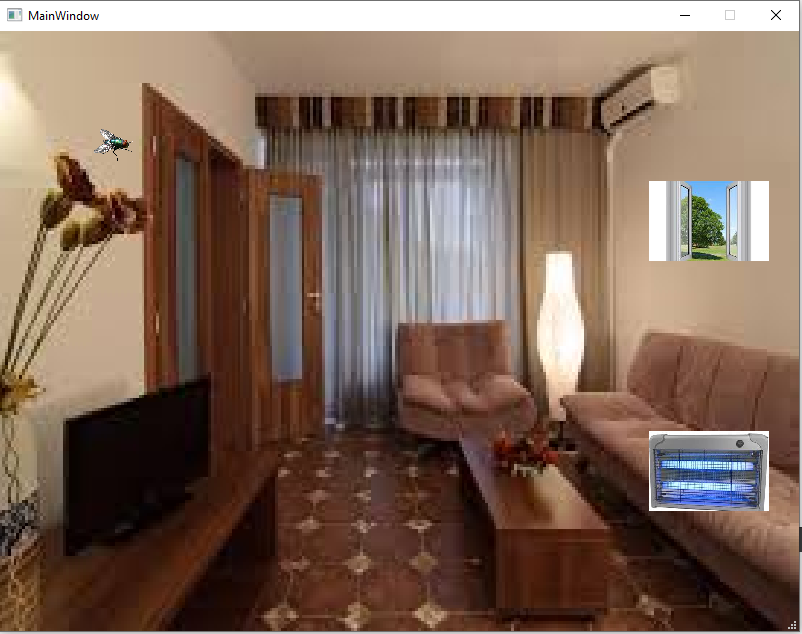
MainWindow w;

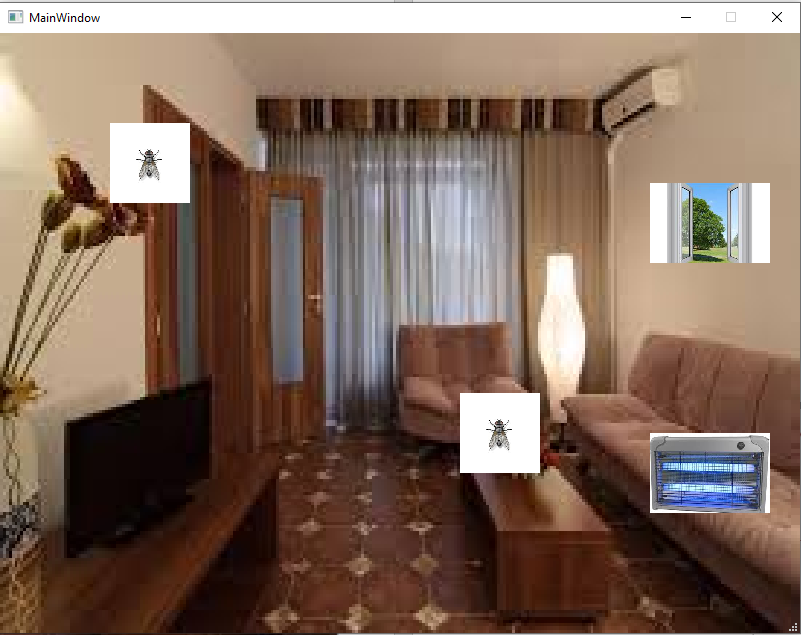
w.show();

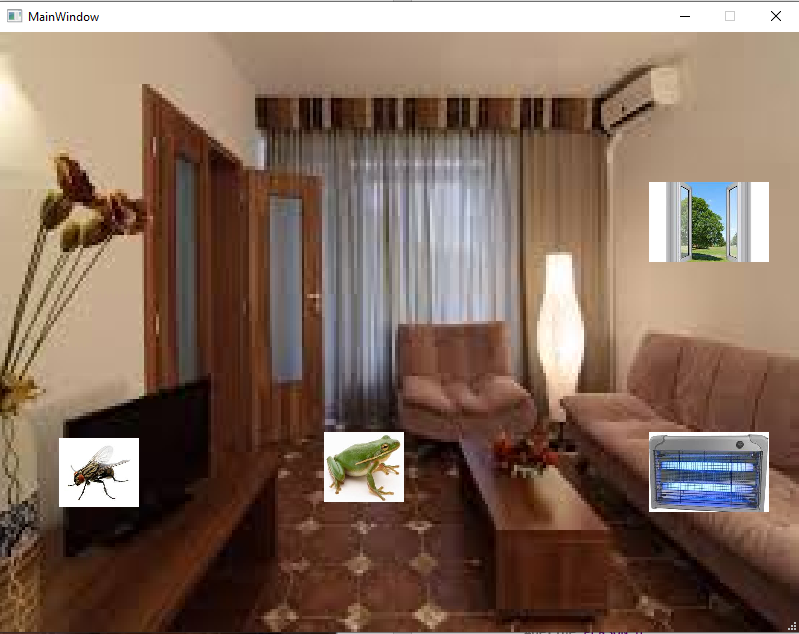
return a.exec();

}

***Скріншот:***

****

****

****

**Завдання 2.**

***dish.h:***

#ifndef DISH\_H

#define DISH\_H

#include <string>

#include <iostream>

using namespace std;

class Dish {

private:

string name;

string description;

double price;

public:

Dish(const string& name, const string& description, double price);

string getName() const;

string getDescription() const;

double getPrice() const;

void display() const;

};

#endif // DISH\_H

***dish.cpp:***

#include "dish.h"

Dish::Dish(const string& name, const string& description, double price)

{

this->name = name;

this->description = description;

this->price = price;

}

string Dish::getName() const {

return name;

}

string Dish::getDescription() const {

return description;

}

double Dish::getPrice() const {

return price;

}

void Dish::display() const {

cout << "Name: " << name << "\nDescription: " << description << "\nPrice: " << price << endl;

}

***Menu.h:***

#ifndef MENU\_H

#define MENU\_H

#include <vector>

#include <algorithm>

#include "Dish.h"

using namespace std;

template<typename T>

class Menu {

private:

vector<T> dishes;

public:

void addDish(const T& dish) {

dishes.push\_back(dish);

}

void removeDish(const string& name) {

auto it = remove\_if(dishes.begin(), dishes.end(), [&name](const T& dish) {

return dish.getName() == name;

});

if (it != dishes.end()) {

dishes.erase(it, dishes.end());

cout << "The dish \"" << name << "\" successfully removed." << endl;

}

else {

cout << "The dish \"" << name << "\" was not found in menu." << endl;

}

}

void displayAll() const {

cout << "List of all dishes:" << endl;

for (const T& dish : dishes) {

dish.display();

}

}

void findDishes(double price) const {

cout << "Dishes with the price " << price << ":" << endl;

bool found = false;

for (const T& dish : dishes) {

if (dish.getPrice() == price) {

dish.display();

found = true;

}

}

if (!found) {

cout << "No dishes with price " << price << endl;

}

}

};

#endif // MENU\_H

***Task2.cpp:***

#include <iostream>

#include "dish.h"

#include "Menu.h"

using namespace std;

int main() {

Menu<Dish> restMenu;

restMenu.addDish(Dish("Pizza", "Tomato sauce, mozzarella, meat", 150.0));

restMenu.addDish(Dish("Mushroom Soup", "Mushroom soup with vegetables", 80.0));

restMenu.addDish(Dish("Beef Steak", "Beef steak with vegetables", 300.0));

restMenu.displayAll();

restMenu.findDishes(150.0);

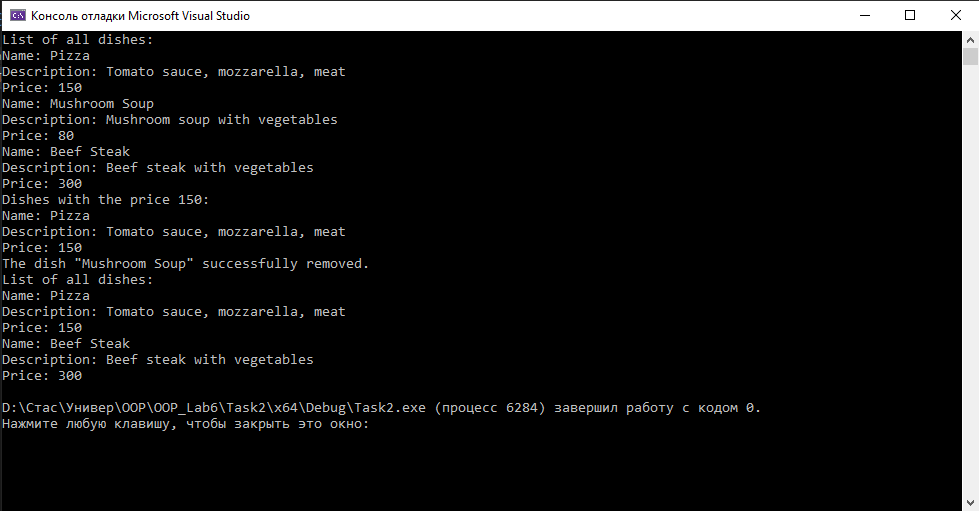
restMenu.removeDish("Mushroom Soup");

restMenu.displayAll();

return 0;

}

***Скріншот:***

****

**Висновок:**

У ході виконання лабораторної роботи №6 “ Шаблони в С++”. Я ознайомився з основними поняттями шаблони та навчився їх програмно реалізовувати мовою С++.