

Репорт о проекте на C++

Выполнила Иванилова Полина

Общая структура проекта

Проект реализует базовый функционал работы с матрицами и фигурами. Система включает в себя поддержку черно-белых (BW) и цветных (RGB) изображений, а также функции для рисования фигур (прямоугольники, круги и т.д.) на этих изображениях.

Ключевые особенности:

- Использование шаблонной функции `draw()` для работы с разными типами фигур.
- Поддержка динамической памяти через `std::unique_ptr` для управления объектами типа `Color`.
- Работа с графическими библиотеками OpenCV для отображения и сохранения изображений.

Реализация функции `draw()`

Функция `draw()` реализована в заголовочных файлах (`bw_matrix.h` и `rgb_matrix.h`):

```
template<typename T>
void draw(T& shape) {}
```

Преимущества:

- За счет использования шаблонов, `draw()` может принимать любые фигуры (прямоугольники, круги и т.д.), как с черно-белыми (`BWColor`), так и с цветными (`RGBColor`) цветами.

Ограничения:

1. Отсутствие виртуальности: Шаблонные функции не поддерживают виртуальные методы. Это связано с тем, что шаблонная функция должна быть определена на этапе компиляции, в то время как виртуальная функция требует динамического полиморфизма (определяется в runtime).
2. Расположение кода: Шаблонная функция должна быть реализована в заголовочном файле (`.h`), иначе она не будет видна при компиляции других файлов.

Работа с цветами (`Color.h`, `BWColor.cpp`, `RGBColor.cpp`)

Фигуры наследуют `std::unique_ptr<Color>` через конструктор. Это обеспечивает безопасное управление динамически выделенной памятью для объектов цвета.. Пример конструкции:

```
this->color = std::move(color);
```

Использование `std::unique_ptr` исключает вероятность утечек памяти и упрощает работу с объектами `Color`. Что позволяет не использовать деструктор.

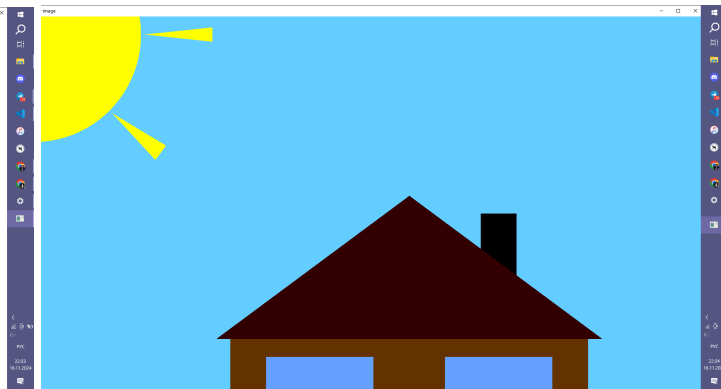
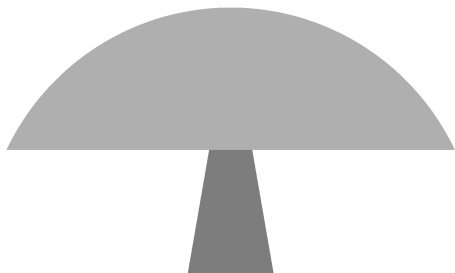
Замечания по производительности

Время выполнения программы зависит от скорости закрытия графического окна, вызванного функцией `matrix.display()`. Если убрать отображение, программа будет выполняться значительно быстрее.

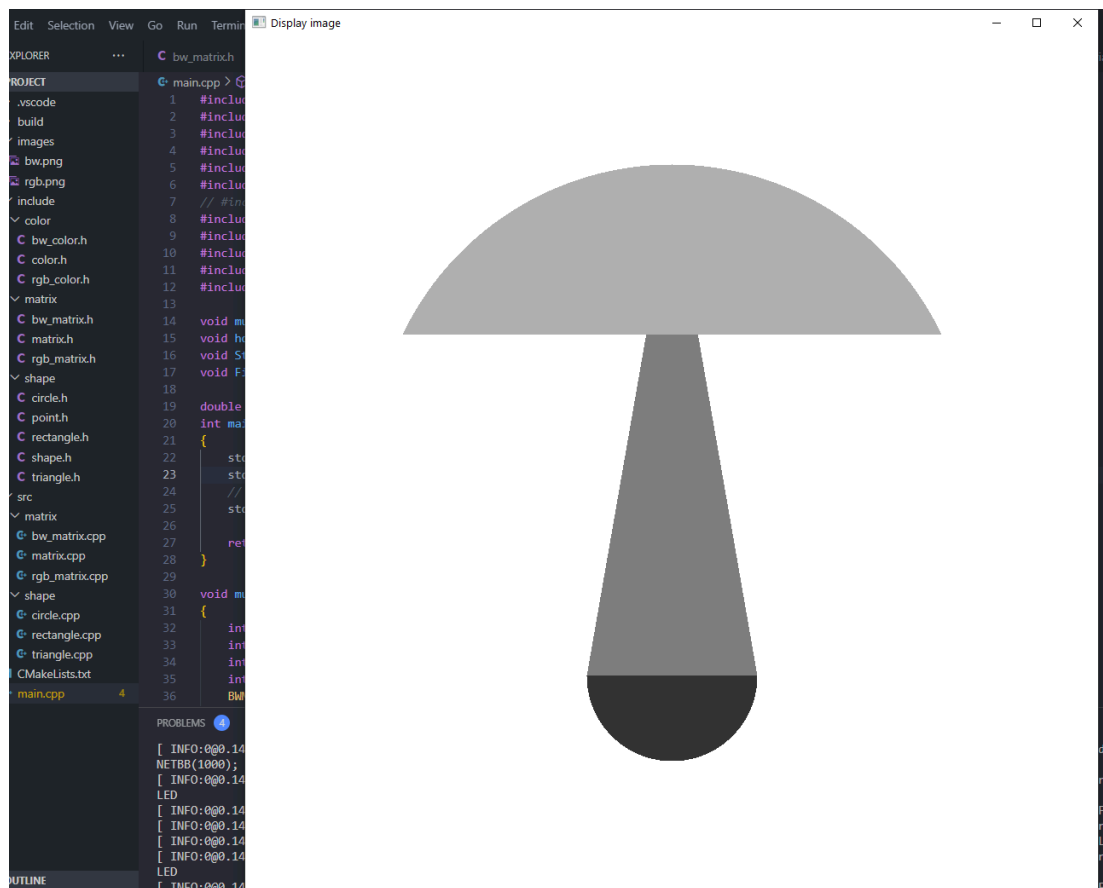
```
Mushroom small time: 105.503ms
House small time: 216.068ms
Mushroom time: 405.499ms
House time: 854.95ms
Fish time: 137.168ms
```

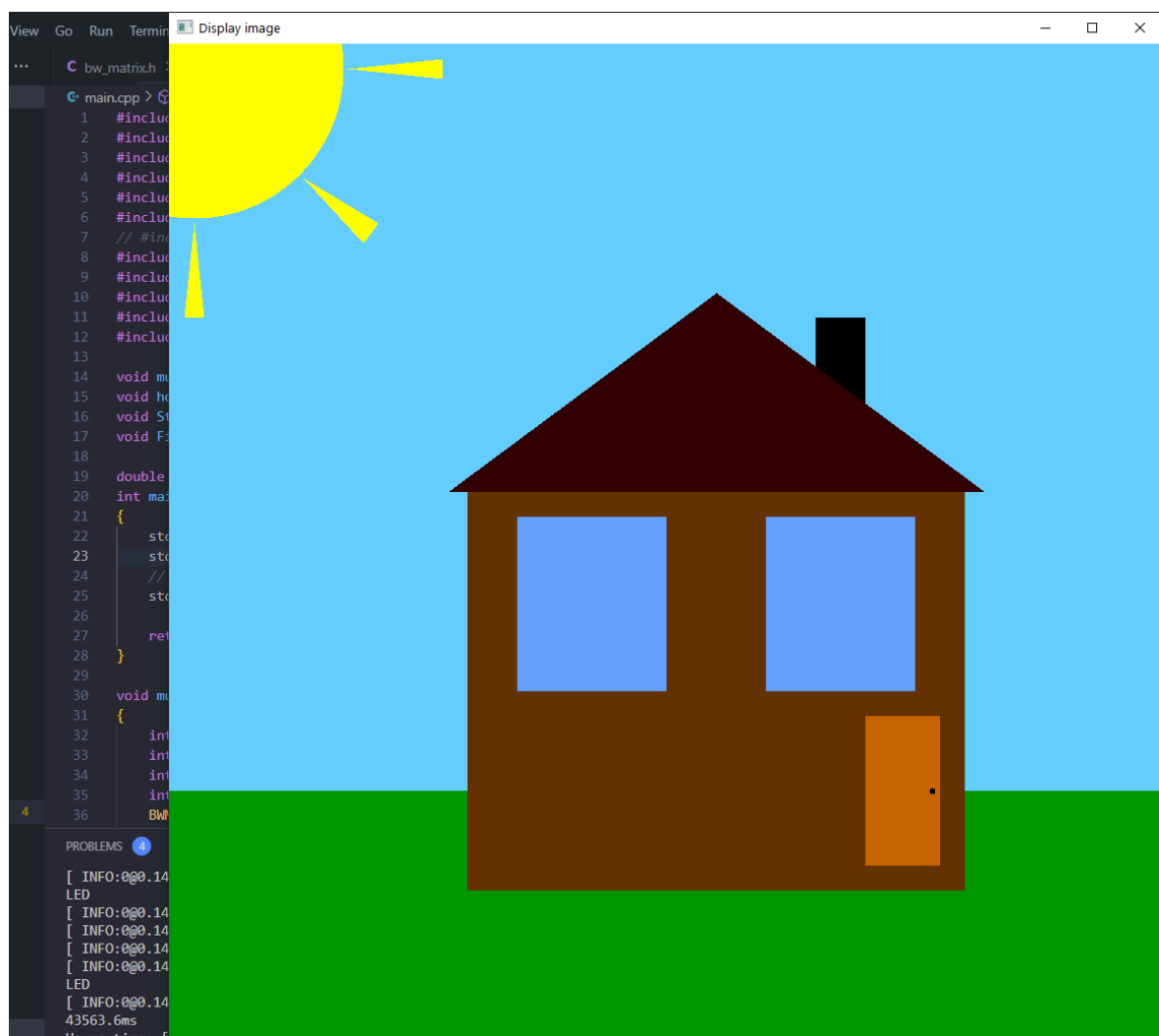
Примеры работы

Проблемы с исходными примерами

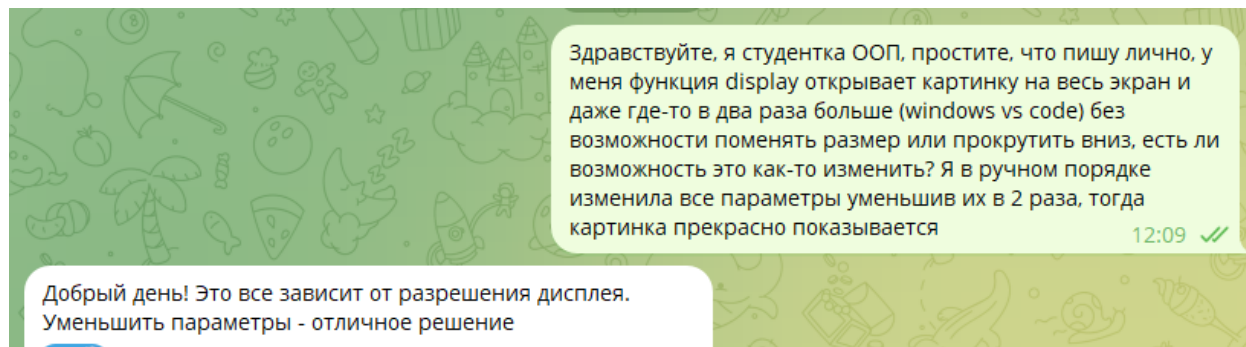


Изначальные примеры (`mushroom()`, `house()`) на моём ноутбуке открывались слишком большими, что делало их неудобными для просмотра. Чтобы это исправить, я уменьшила все координаты и размеры фигур в два раза. Получились версии: `mushroom_small()` и `house_small()`






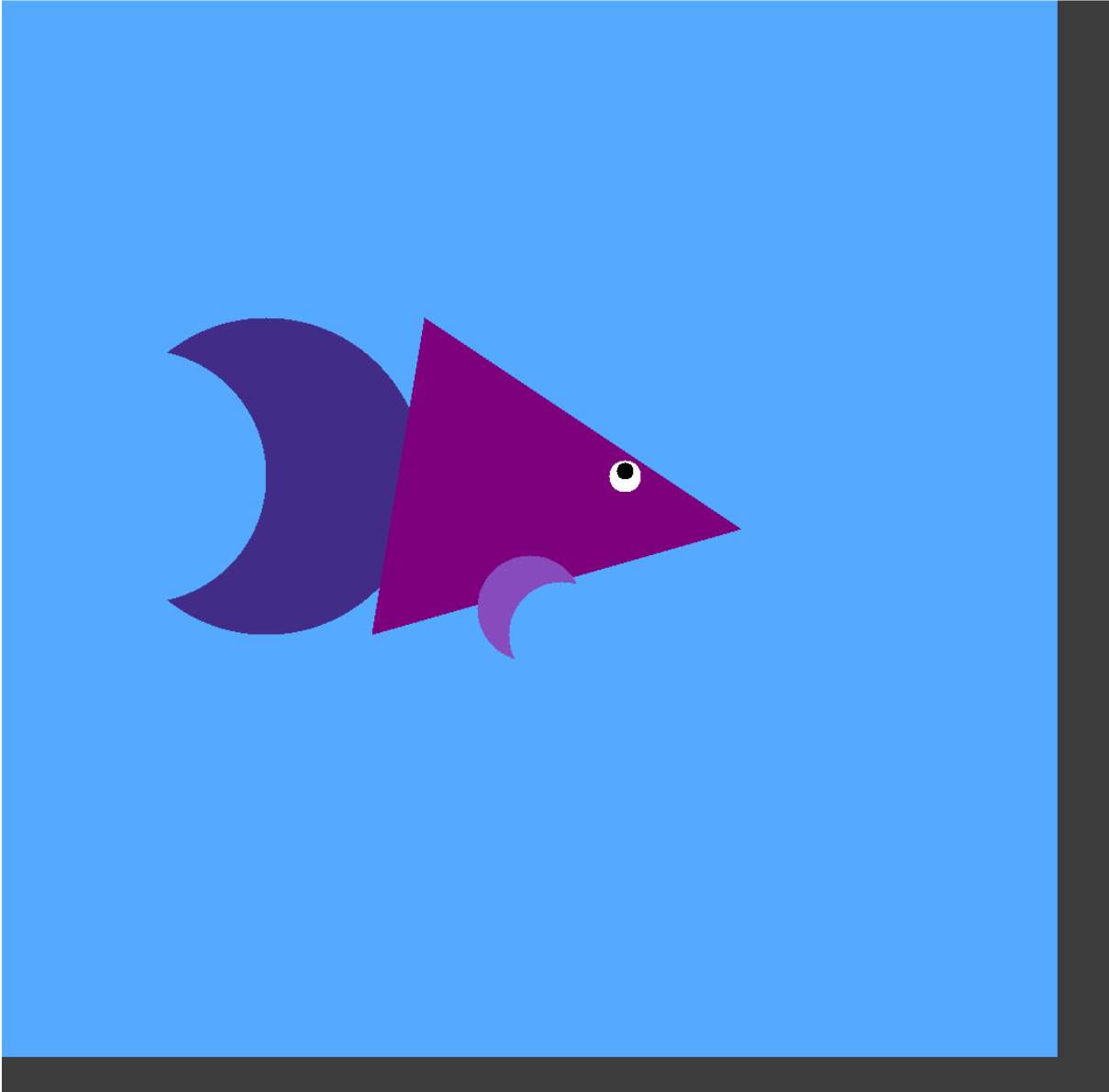
Такой подход был одобрен преподавателем:



Мой пример: Рыбка

Мой был написан пример ``void Fish()``, который рисует фигурами рыбу.

 Display image



Заключение

Проект успешно реализует поставленные задачи:

- Поддержка шаблонных функций для работы с разными типами данных.
- Безопасное управление памятью через ``std::unique_ptr``.
- Взаимодействие с библиотекой OpenCV для отображения и сохранения изображений.

Модификации и улучшения позволили адаптировать программу под реальные условия использования и сделать ее более удобной.