**Анализ выполненной задачи**

1. **Функциональные требования**. Программа успешно реализует все основные функции для управления задачами: добавление, удаление, отображение задач (активных и выполненных), пометка задач как выполненных и сохранение/загрузка задач из файла. Эти функции соответствуют требованиям задачи и покрывают базовую логику управления задачами в консольном режиме.
2. **Удобство использования**. Пользовательский интерфейс программы представлен в виде меню, которое позволяет пользователю легко выбирать нужные действия. Ввод происходит через стандартную консоль, что подходит для начального уровня реализации, но ограничивает гибкость интерфейса.
3. **Сохранение данных**. Программа корректно сохраняет и загружает данные из файла tasks.txt, обеспечивая сохранение состояния между сессиями работы пользователя. Однако, формат данных прост и не защищен от потенциальных ошибок чтения в случае, если формат файла изменится или файл будет повреждён.
4. **Качество кода**. Код структурирован на несколько классов и файлов, что повышает читаемость и удобство поддержки. Методы классов Task и TaskList отделены, и каждый класс имеет чёткие обязанности. Однако некоторые функции могли бы быть оптимизированы для улучшения эффективности и обработки ошибок.
5. **Обработка ошибок**. В программе реализована базовая обработка ошибок при открытии файлов, что помогает предотвратить аварийное завершение программы. Тем не менее, можно было бы добавить дополнительные проверки, например, на ввод пользователя и корректность данных в файле.
6. **Комментарии и документация**. Код содержит базовые комментарии, объясняющие назначение классов, методов и отдельных фрагментов кода. Это помогает ориентироваться в структуре программы, особенно новичкам. Но можно улучшить комментарии в сложных местах и добавить документацию к каждому методу и классу.
7. **Распределение на файлы**. Код грамотно распределен на заголовочные и реализационные файлы (.h и .cpp), что делает его более удобным для компиляции и дальнейшего расширения, например, добавления новых функций.

**Рекомендации по устранению выявленных ошибок и улучшению**

1. **Защита от некорректного ввода**. Добавить проверку ввода пользователя для обработки некорректных данных. Например, проверять, что пользователь ввёл корректное числовое значение при выборе пункта меню, а также обрабатывать пустые строки и невалидные символы.
2. **Оптимизация сохранения и загрузки данных**. Рассмотреть использование более структурированного формата данных, например, JSON или XML, для хранения задач, что упростит расширение программы в будущем и добавит гибкость в обработке ошибок при загрузке данных из файла.
3. **Обработка ошибок при работе с файлами**. В дополнение к проверке открытия файла можно добавить проверку корректности содержимого при чтении, чтобы предотвратить ошибки при повреждении файла или некорректной записи данных.
4. **Добавление тестирования**. Создать тесты для каждой функции, особенно для операций сохранения, загрузки и управления задачами. Это позволит автоматизировать проверку работоспособности программы при внесении изменений и упрощает процесс отладки.
5. **Оптимизация методов удаления и поиска**. При большом количестве задач метод удаления задачи, основанный на линейном поиске, может замедлять работу. Возможно использование контейнеров, поддерживающих быстрый поиск, таких как std::unordered\_map, для улучшения производительности.
6. **Обработка некорректного формата файла**. В случае, если формат файла изменится или данные будут повреждены, программа может некорректно загрузить задачи. Рекомендуется добавить проверку валидности каждого поля при загрузке данных.
7. **Добавление функционала редактирования задачи**. Чтобы улучшить функциональность программы, можно добавить возможность редактировать уже существующие задачи, изменяя их название, описание и статус.

Эти улучшения помогут сделать программу более надёжной, удобной и расширяемой, что пригодится как для текущего использования, так и для возможных будущих разработок.