Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Тихоокеанский государственный университет»

Высшая школа кибернетики и цифровых технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

ПО КУРСУ «ОСНОВЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ»

«MAKEFILE ДЛЯ ПРОГРАММНОГО КОМПЛЕКСА»

Выполнил: студент 2 курса группы ПО(б)-31

Магонов Александр Александрович

Проверила: ассистент ВШ КЦТ

Балабасова Алиса Дмитриевна

Хабаровск 2024 г.

Цель работы: написать Makefile-ы для программного комплекса, разработанного в лабораторной работе №1.

Задание:

Для библиотеки и основного приложения файлы пишутся отдельно. В

каждом Makefile предусмотреть следующие правила:

1.complie. Цель по умолчанию. Выполняет компиляцию части программного комплекса.

2. install. Установка части программного комплекса в заданную папку. Папка задаётся

переменной в Makefile. Для библиотеки установка двоичного файла производится в папку

lib основной папки, заголовочного — в папку include.

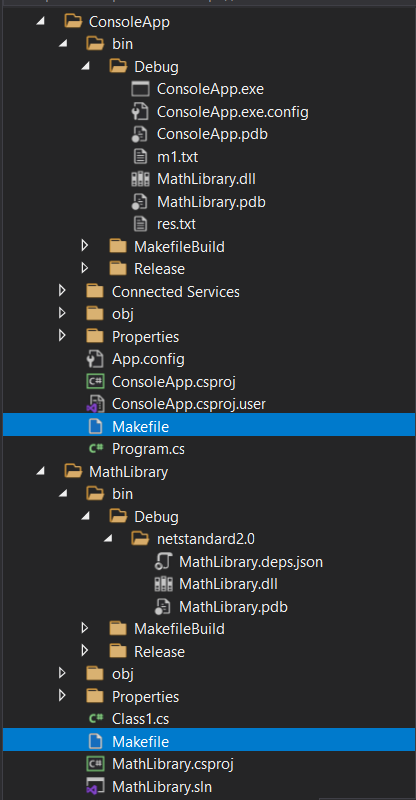
3. clean. Очистка папки от ранее откомпилированных двоичных файлов.

4. uninstall. Удаление ранее установленной части программного комплекса.

Для Makefile основной программы расположение библиотеки задаётся дополнительной

переменной.

Структура комплексного приложения с консольным приложением (ConsoleApp.exe) и динамической библиотекой (MathLibrary.dll), а также Makefile для приложения и библиотеки.

  
Рисунок 1 – структура комплексного приложения с Makefile

Код файла Makefile с инструкцией для сборки dll-библиотеки MathLibrary на рисунке 2.

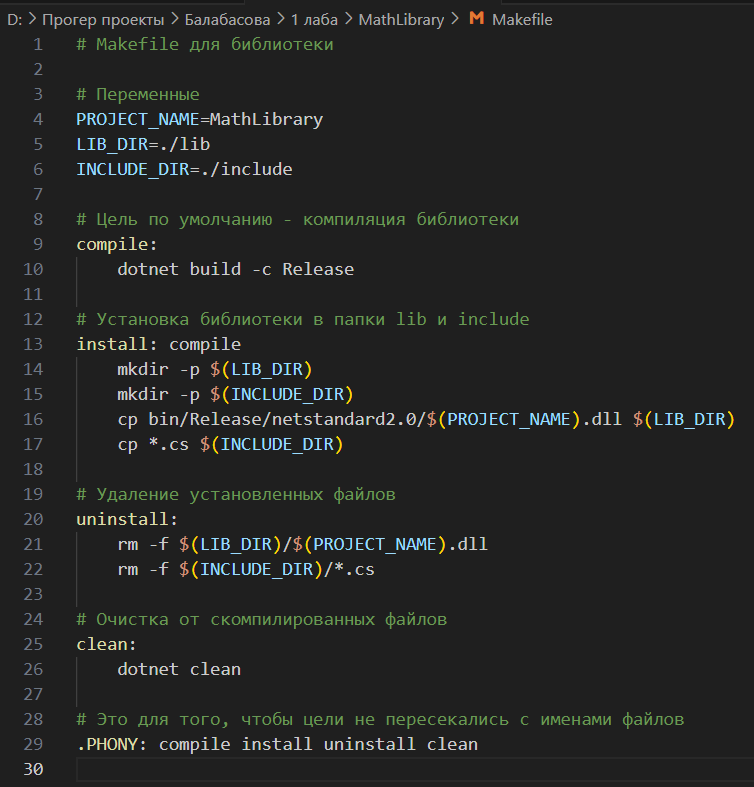


Рисунок 2 – код Makefile для динамической библиотеки

Код файла Makefile с инструкцией для сборки консольного приложения ConsoleApp на рисунке 3.

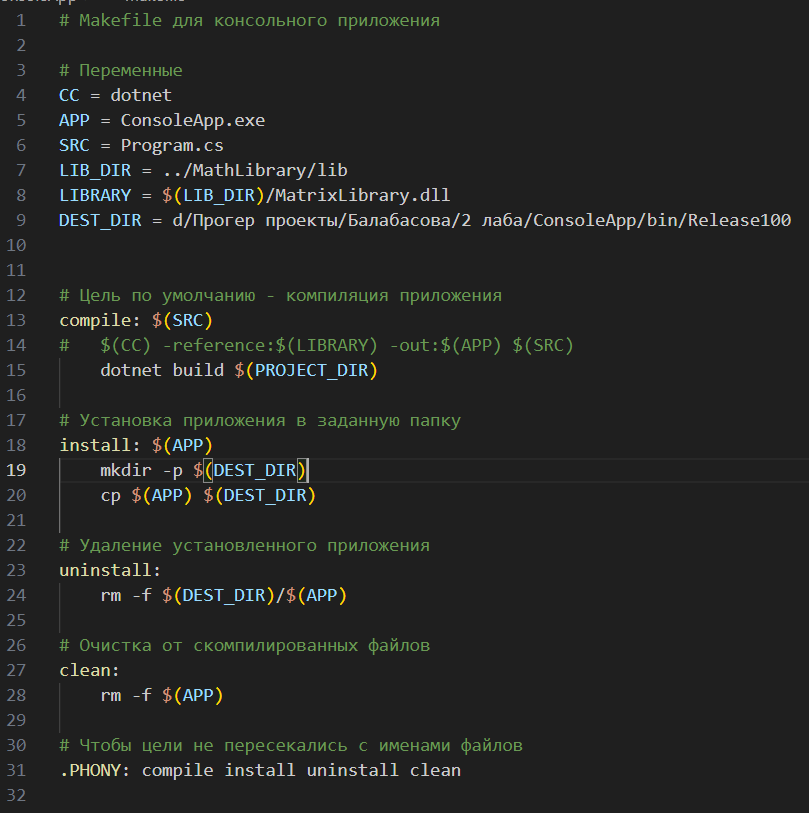


Рисунок 3 – код Makefile для консольного приложения

Демонстрация сборки (компиляции) проекта с помощью Makefile в Visual Studio Code через консоль powershell, запущенной в папке с нашей программой, а также с помощью Git Bash (MINGW64):

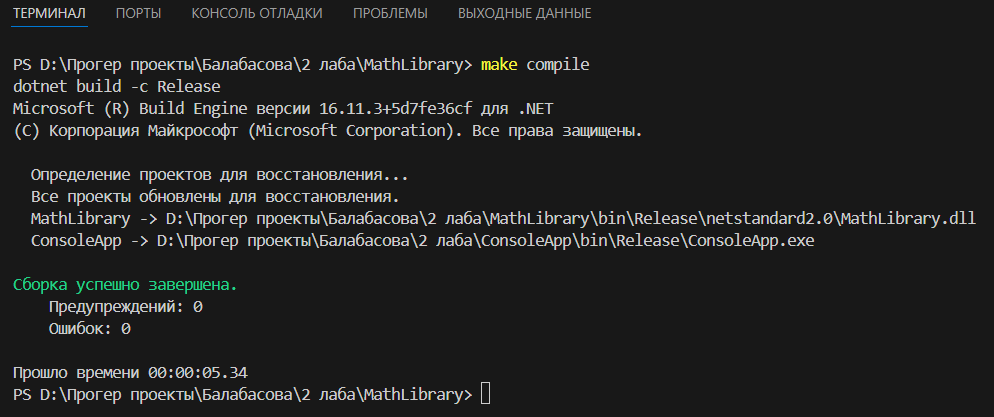


Рисунок 4 – пример выполнения команды компиляции Makefile

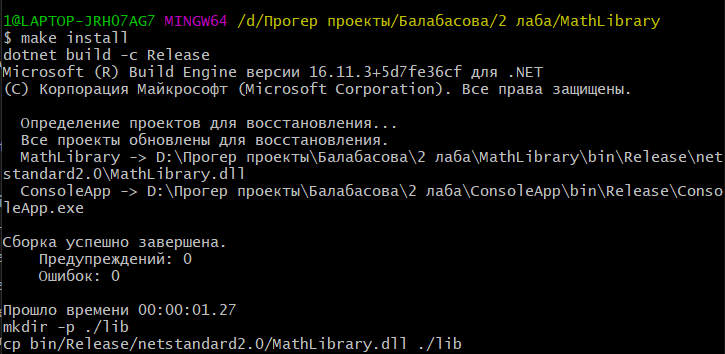


Рисунок 5 – пример выполнения команды компиляции Makefile

Вывод: цель лабораторной работы заключалась в создании программного комплекса, состоящего из консольного приложения и динамической библиотеки для выполнения операций над матрицами размером 4x4. В ходе работы были выполнены следующие задачи:

* Разработаны **Makefile-ы** для автоматизации сборки, установки, очистки и удаления компонентов программного комплекса. Makefile для библиотеки и Makefile для основного приложения созданы отдельно, что позволяет гибко управлять каждой частью проекта.
* Создана **динамическая библиотека MathLibrary**, содержащая функции для выполнения операций над матрицами: сложение, вычитание и умножение матриц. Библиотека разделена на заголовочный файл и файл с реализацией функций.
* Реализованы команды в **Makefile библиотеки**:
  + compile — выполняет компиляцию библиотеки.
  + install — устанавливает библиотеку и заголовочные файлы в указанные папки lib и include.
  + clean — очищает папку от ранее скомпилированных файлов.
  + uninstall — удаляет установленную библиотеку и заголовочные файлы.
* Написано **консольное приложение MatrixApp** с Makefile для основного приложения, который позволяет:
  + Скомпилировать приложение (compile).
  + Установить приложение в указанную папку (install).
  + Очистить скомпилированные файлы (clean).
  + Удалить установленное приложение (uninstall).
* Приложение поддерживает **интерактивный ввод данных** от пользователя, включая ввод команд для выполнения операций над матрицами. Реализован анализ аргументов командной строки с поддержкой операций сложения (add), вычитания (subtract), умножения (multiply), а также вывод справки по командам через ключ --help.
* Обеспечена **обработка ошибок**: проверка корректности входных данных (размер матриц), обработка ошибок при некорректных входных данных и попытках выполнения операций с неинициализированными объектами. Программа протестирована на различных наборах данных для всех поддерживаемых операций, а также на случай некорректного ввода, что позволяет избежать аварийного завершения работы приложения.

В результате выполнения лабораторной работы был создан программный комплекс, соответствующий требованиям задания. Комплекс включает приложение и динамическую библиотеку, обеспечивающих выполнение операций над матрицами в интерактивном режиме с обработкой ошибок и выводом результатов. Лабораторная работа продемонстрировала практическое применение концепций модульной разработки, использования динамических библиотек и автоматизации сборки с помощью Makefile в C#.