



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ
КАФЕДРА

«Информатика и системы управления» (ИУ)
«Информационная безопасность» (ИУ8)

Домашняя работа № 2
ПО КУРСУ
«Алгоритмические языки»

Студенты

ИУ8-13

(Группа)

В. С. Ажгирей,
С. Е. Матушин,
Е. М. Грязнов
(И. О. Фамилия)

Преподаватель:

М. В. Малахов
(И.О. Фамилия)

Введение

Цели и задачи работы

Цель работы состоит в овладении навыками разработки архитектуры ПО и сборки ПО. Для достижения цели необходимо выполнить следующие задачи:

1. Освоение сборки ПО с использованием `stake` через командную строку.
2. Разработка архитектуры целевого ПО, включающая в себя:
 - согласование с преподавателем окончательного условия задачи (ТЗ);
 - согласование форматов входных и выходных данных;
 - согласование ограничений на входные данные;
 - согласование с преподавателем разбиения программы на модули (различные `.h` и `.cpp` файлы) и назначение модулей;
 - разработка и ручной расчёт нескольких тестовых примеров.

Реализация

Входные данные располагаются в файле, значения каждого байта меньше `0x80`. Файл архивируется (сжимается) с помощью разработанной программы: если встречаются несколько подряд повторяющихся байт, то они заменяются на 2 байта, первый из которых байт `0x80 + количества повторений данного байта`, а второй - сам повторяющийся байт (например, если идут четыре подряд одинаковых байта `0x70`, то есть "`0x70 0x70 0x70 0x70`", то они заменяются на "`0x84 0x70`"). На выходе будет получаться сжатый файл, изменённый таким образом. Так же, если надо разархивировать (разжать) файл, то выполняются обратные операции (например, набор байт "`0x84 0x70`" заменяется на "`0x70 0x70 0x70 0x70`"). Соответственно, получается разархивированный файл.

Условия для 1 варианта

Разработайте приложение-архиватор, предназначенное для сжатия файлов, в которых значение каждого байта меньше 0x80. Сжатие производить следующим методом: если подряд встречается несколько одинаковых байт (например 0x70 0x70 0x70), то удалять все дубли, а вместо них добавлять их количество + 0x80 (то есть для приведенного случая должно получиться 0x83 0x70). Приложение должно уметь как архивировать, так и разархивировать файлы указанным методом.

Основная часть

В ходе выполнения задания программа была разбита на несколько файлов: main.cpp, source.cpp и source.h. Программа на вход будет принимать 2 аргумента: режим работы архиватора (“-c” - сжатие/ “-d” - разжатие) и путь к файлу, с которым необходимо выполнить действие. С помощью стейк была проведена сборка этих файлов в файл myapp.exe.

Снимки выполнения работы



The image shows a Notepad window titled "CMakeLists.txt - Блокнот". The menu bar includes "Файл", "Правка", "Формат", "Вид", and "Справка". The text content of the file is as follows:

```
cmake_minimum_required(VERSION 3.28)

set(CMAKE_CXX_STANDARD 17)

project(My_homework VERSION 1.0 LANGUAGES CXX)

add_executable(myapp main.cpp functions.cpp)
```

The status bar at the bottom indicates "Стр 1, столб 1", "100%", "Windows (CRLF)", and "UTF-8".

Рисунок 1 – файл CMakeLists.txt

```
MINGW64:/e/LaboratoryWorks/homework_2/build

sergo@DESKTOP-AMLL53A MINGW64 /e/LaboratoryWorks/homework_2 (homework)
$ cd build/

sergo@DESKTOP-AMLL53A MINGW64 /e/LaboratoryWorks/homework_2/build (homework)
$ cmake ..
-- Building for: Visual Studio 17 2022
-- Selecting Windows SDK version 10.0.22621.0 to target windows 10.0.19045.
-- The CXX compiler identification is MSVC 19.37.32825.0
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files/Microsoft Visual Studio/2022/Community/VC/Tools/MSVC/14.37.32822/bin/Hostx64/x64/cl.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done (4.2s)
-- Generating done (0.0s)
-- Build files have been written to: E:/LaboratoryWorks/homework_2/build

sergo@DESKTOP-AMLL53A MINGW64 /e/LaboratoryWorks/homework_2/build (homework)
$ cmake --build . --config Release
Версия MSBuild 17.7.2+d6990bcfa для .NET Framework

1>Checking Build System
Building Custom Rule E:/LaboratoryWorks/homework_2/CMakeLists.txt
main.cpp
functions.cpp
Создание кода...
myapp.vcxproj -> E:\LaboratoryWorks\homework_2\build\Release\myapp.exe
Building Custom Rule E:/LaboratoryWorks/homework_2/CMakeLists.txt

sergo@DESKTOP-AMLL53A MINGW64 /e/LaboratoryWorks/homework_2/build (homework)
$ |
```

Рисунок 2 – сборка ПО с помощью cmake через командную строку

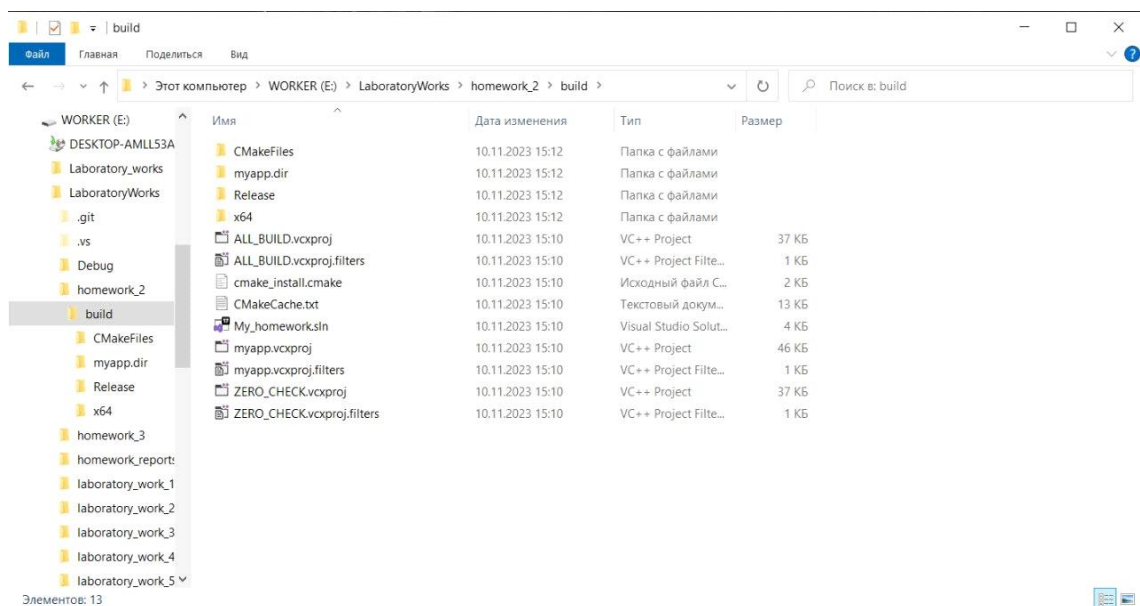


Рисунок 3 – файлы, созданные в процессе сборки через командную строку

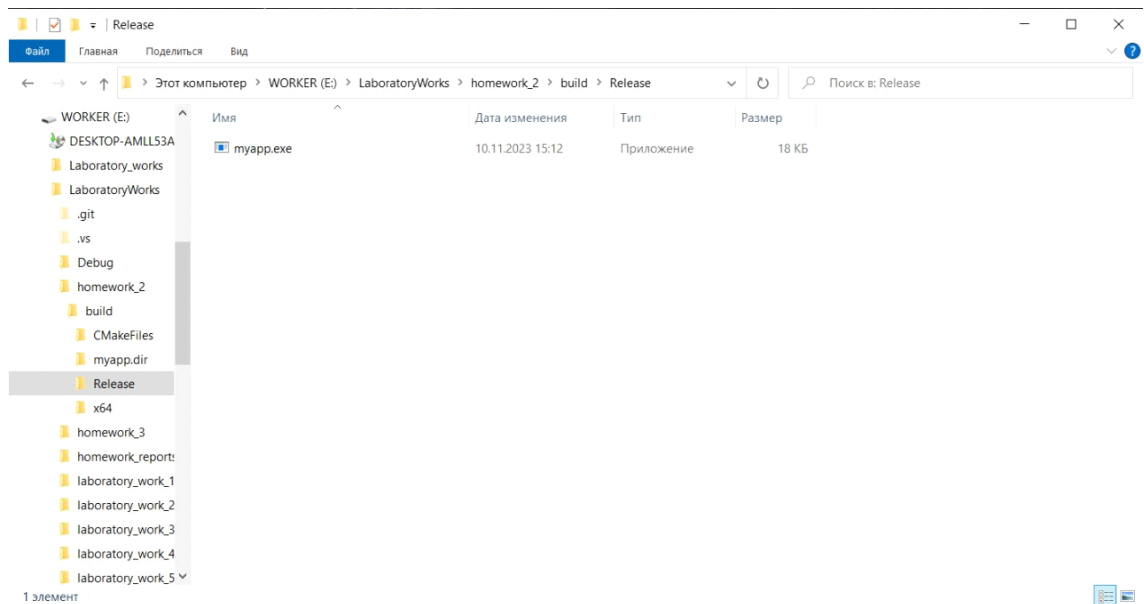


Рисунок 4 – собранная программа

Примеры работы программы

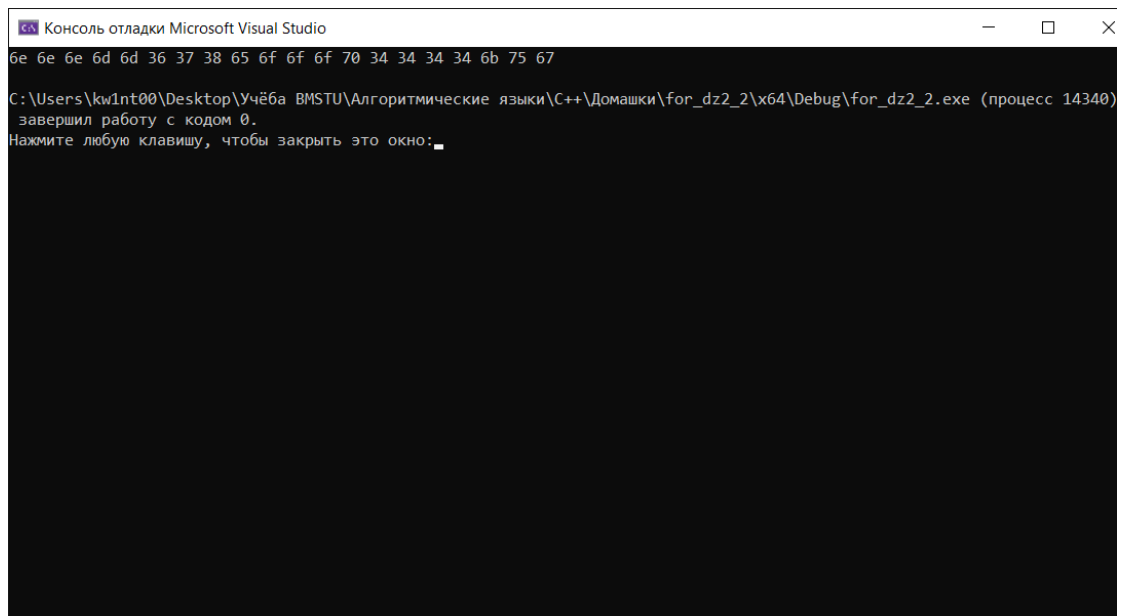
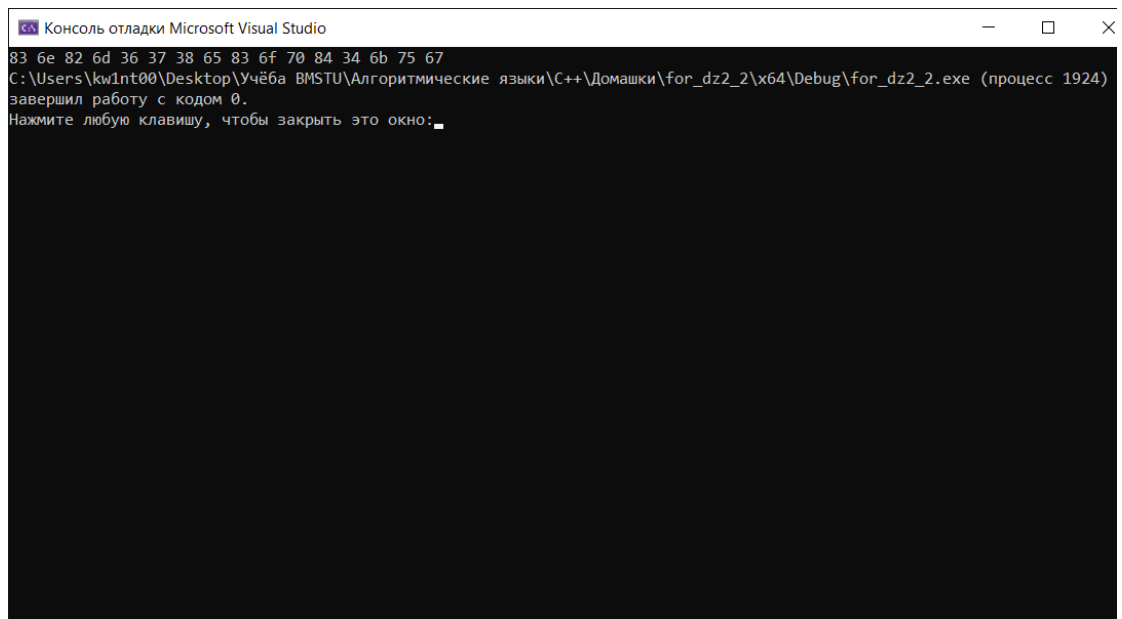


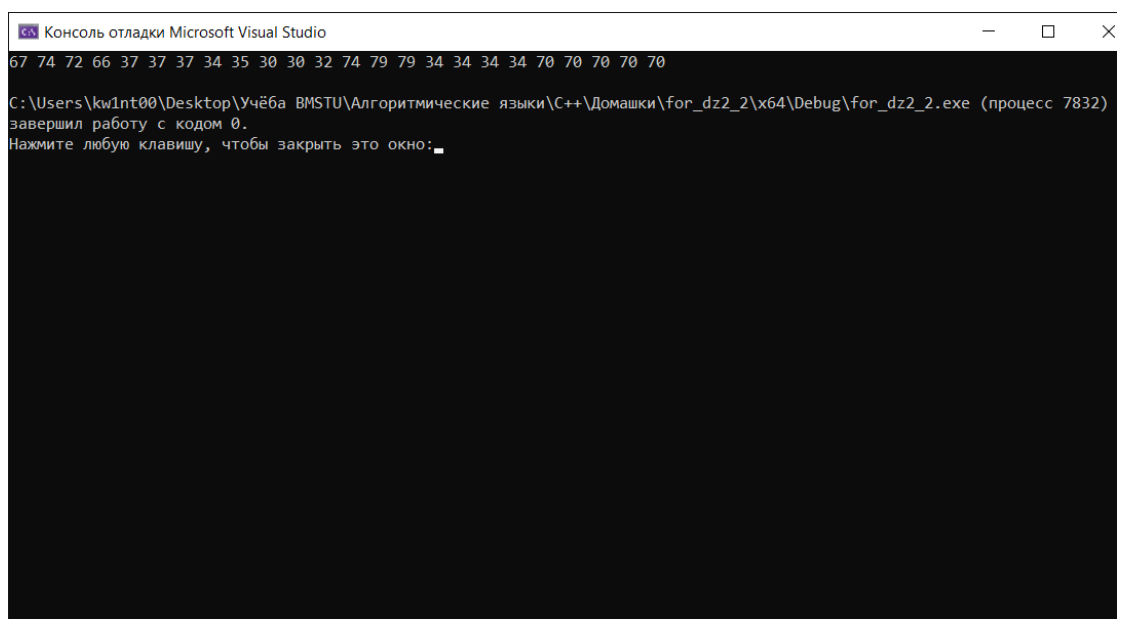
Рисунок 5 – первый исходный файл



Консоль отладки Microsoft Visual Studio

```
83 6e 82 6d 36 37 38 65 83 6f 70 84 34 6b 75 67
C:\Users\kwInt00\Desktop\Учёба ВМSTU\Алгоритмические языки\C++\Домашки\for_dz2_2\x64\Debug\for_dz2_2.exe (процесс 1924)
завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: █
```

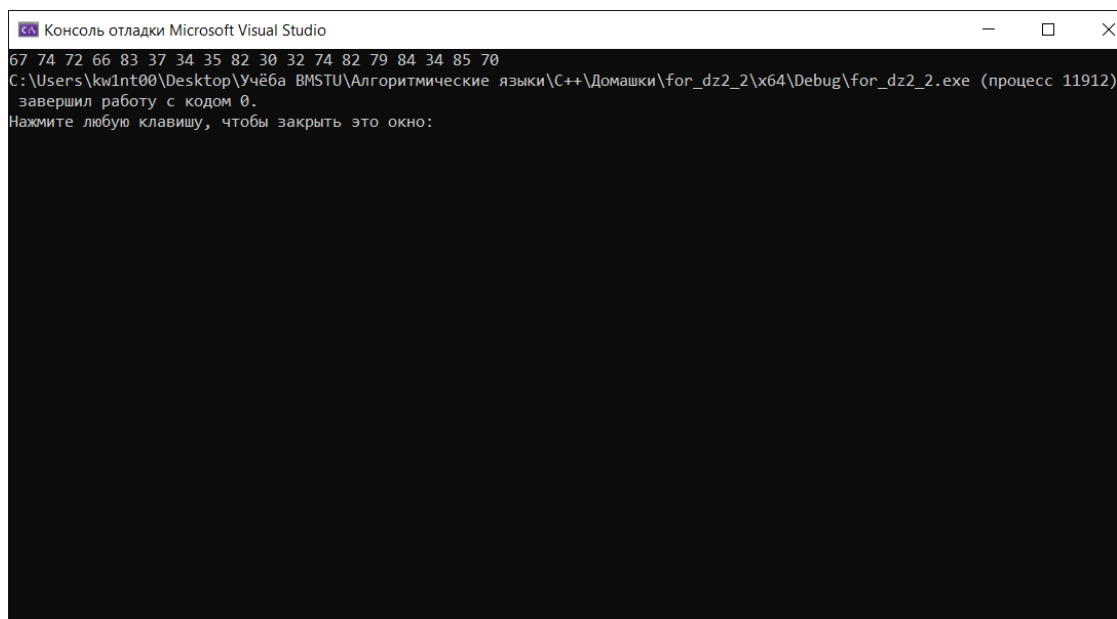
Рисунок 6 – первый сжатый файл



Консоль отладки Microsoft Visual Studio

```
67 74 72 66 37 37 37 34 35 30 30 32 74 79 79 34 34 34 34 70 70 70 70
C:\Users\kwInt00\Desktop\Учёба ВМSTU\Алгоритмические языки\C++\Домашки\for_dz2_2\x64\Debug\for_dz2_2.exe (процесс 7832)
завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: █
```

Рисунок 7 – второй исходный файл



```
Консоль отладки Microsoft Visual Studio
67 74 72 66 83 37 34 35 82 30 32 74 82 79 84 34 85 70
C:\Users\kwInt00\Desktop\Учёба ВМSTU\Алгоритмические языки\C++\Домашки\for_dz2_2\x64\Debug\for_dz2_2.exe (процесс 11912)
завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 8 – второй сжатый файл

Заключение

Задачи домашней работы были решены, результаты проверены. Разработана архитектура ПО (условие задачи и разбиение программы на различные модули были согласованы с преподавателем, также были согласованы форматы входных и выходных данных, ограничения на входные данные, рассчитаны тестовые примеры), а также изучена на практике сборка ПО с помощью стэке через командную строку.