Отчет выполнил: студент ВМК-22, Александров Данил Сергеевич

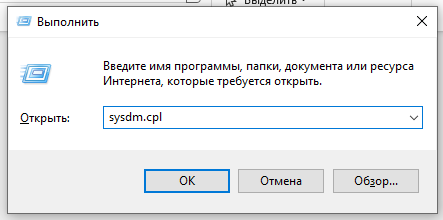
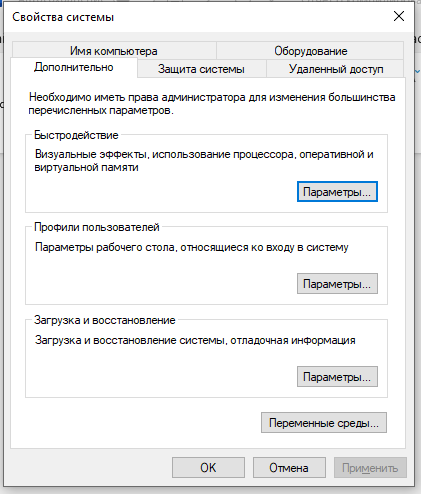
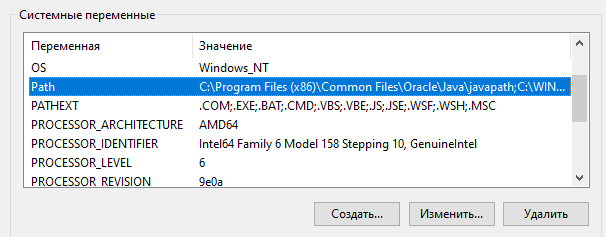
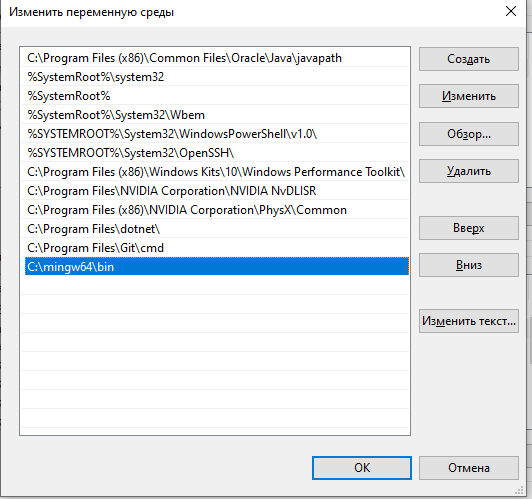
**Отчёт о компилировании программ в консоли**

Добавление пути к компилятору в переменную среды окружения Path

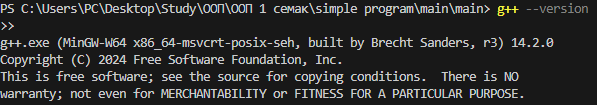
Добавление пути к компилятору в переменную среды окружения Path нужно для того, чтобы можно было компилировать программы из любой директории с помощью командной строки.

Отчет составлен на примере добавления компилятора mingw64

Шаги добавления mingw64 в системные переменные:

1. Нажать Win + R, ввести sysdm.cpl и нажми Enter.  
   
2. Перейти во вкладку Дополнительно и нажать Переменные среды.  
   
3. В разделе Системные переменные найти Path и нажмать Изменить.  
   
4. Нажать Создать и добавить путь к bin папке mingw64  
   
5. Нажать Ок и закрыть все окна.

Проверка установки:

1. Открыть терминал(CMD)
2. Ввести команду g++ --version
3. Если все корректно, то будет следующее сообщение:  
   

Для компиляции программы на C++ с помощью g++ в командной строке (терминале):

1. Компиляция одиночного файлаg++ main.cpp -o program  
Здесь:

* g++ — компилятор C++.
* main.cpp — исходный файл.
* -o program — имя выходного исполняемого файла (program).

Запуск программы:  
program.exe

2. Компиляция нескольких файлов  
Если программа состоит из нескольких файлов, например main.cpp и triangle.cpp:  
g++ main.cpp triangle.cpp -o program

3. Использование стандартов C++   
Чтобы указать стандарт C++, например C++17 или C++20:   
g++ -std=c++17 main.cpp -o program

Самое важное для работы в консоли:

dir – команда просмотра текущей директории(windows аналог команды ls).  
cd – команда смена директории.  
cd .. – команда для выхода к родительской директории.  
cd \ — перемещает в корень диска.

Автодополнение: Tab для автодополнения имен файлов и папок.

Для повторного использования предыдущих команд:

1. Стрелка вверх — вызывает последнюю введенную команду.
2. Стрелка вниз — если использована стрелка вверх, позволяет перемещаться по списку команд.

5. Как уменьшить размер исполняемого файла, создаваемого компилятором MinGW (GCC)

1. Флаг -Os заставляет компилятор оптимизировать код таким образом, чтобы уменьшить размер исполняемого файла.  
   g++ main.cpp triangle.cpp -o program -Os
2. Флаг -s удаляет отладочную информацию и символы, которые не нужны для работы программы, тем самым уменьшая ее размер.  
   g++ main.cpp triangle.cpp -o program -Os -s
3. Флаг -flto включает оптимизацию на этапе линковки, что помогает сократить размер программы.  
   g++ main.cpp triangle.cpp -o program -Os -s -flto

6. Cкомпилируйте cpp файл модуля в статическую библиотеку.

Статическая библиотека — это коллекция объектных файлов, которые компилируются и собираются в один файл с расширением .lib (для Windows).  
Компиляция .cpp файла модуля в статическую библиотеку позволяет повторно использовать код в разных проектах без необходимости перекомпиляции. Статическая библиотека связывается с программой на этапе компиляции и может значительно уменьшить время сборки проекта, а также облегчить управление зависимостями.

Шаги для создания статической библиотеки:

1. Компиляция исходного файла в объектный файл:  
   Нужно скомпилировать файл модуля triangle.cpp в объектный файл с помощью флага -c. Это позволяет избежать линковки в момент компиляции.  
   g++ -c triangle.cpp -o triangle.o  
   В результате создан объектный файл triangle.o
2. Инструмент ar для создания статической библиотеки из объектного файла.   
   ar rcs libtriangle.a triangle.o  
   ar — это утилита для работы с архивами.

rcs — флаги для создания библиотеки:

* 1. r — добавление/обновление объектных файлов в архив,
  2. c — создание архива, если он ещё не существует,
  3. s — создание индекса для библиотеки, что ускоряет поиск в библиотеке

В результате создан файл библиотеки libtriangle.a

1. Компиляция программы с использованием библиотеки:  
   g++ main.cpp -L. -ltriangle -o program  
   -L. — указывает компилятору, что библиотека находится в текущей директории

-ltriangle — указывает компилятору на библиотеку libtriangle.a.

1. Компиляция программы с использованием библиотеки:  
   Запуск программы:  
   ./program