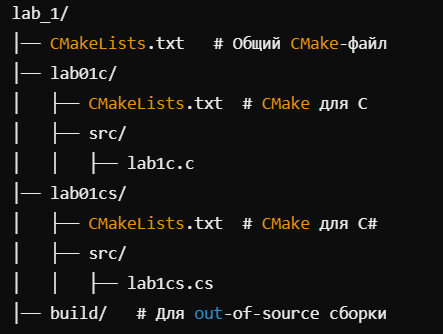
Для Windows:  
Создаем файловую структуру:  
  
  
Общий cmake-файл:

|  |
| --- |
| cmake\_minimum\_required(VERSION 3.30)  project(labtest)  set(EXECUTABLE\_OUTPUT\_PATH "${CMAKE\_BINARY\_DIR}")  # Определение ОС  if(WIN32)  message("Hello, I am Windows!")  elseif(APPLE)  message("Hello, I am macOS!")  elseif(UNIX)  message("Hello, I am Linux!")  endif()  add\_subdirectory(lab1c)  include(CheckLanguage)  check\_language(CSharp)  if(CMAKE\_CSharp\_COMPILER)  enable\_language(CSharp)  message(STATUS "Yes C# support")  add\_subdirectory(lab1cs)  else()  message(STATUS "No c# support")  endif() |

cmake для c:

|  |
| --- |
| cmake\_minimum\_required(VERSION 3.30)  project(CWorld C)  add\_executable(CWorld ./src/lab1c.c) |

cmake для cs:

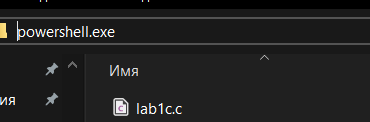
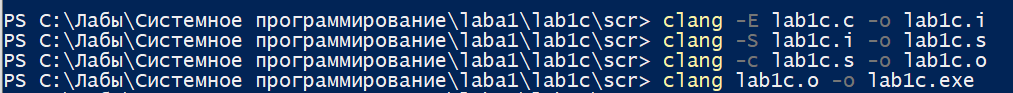
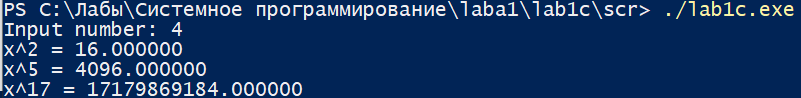
|  |
| --- |
| cmake\_minimum\_required(VERSION 3.30)  project(CSWorld CSharp)  add\_executable(CSWorld ./src/lab1cs.cs) |

lab1c.c:  
в папке создаем текстовый документ, меняем его название и формат на «lab1c.c» и код по варианту:

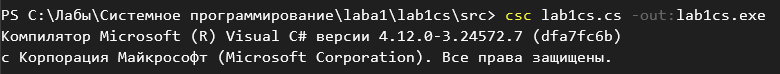
|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  int main() {  double x;  printf("Chislo: ");  scanf("%lf", &x);  printf("x^21 = %lf\n", x \* x);  printf("\nPress Enter");  getchar();  getchar(); // Нужно дважды, если перед этим использовался `scanf`  return 0;  } |

lab1cs.cs:

|  |
| --- |
| using System;  class Program  {  static double Power21(double x)  {  double x2 = x \* x;  double x4 = x2 \* x2;  double x8 = x4 \* x4;  double x16 = x8 \* x8;  return x16 \* x4 \* x;  }  static void Main()  {  Console.Write("Chislo: ");  double x = Convert.ToDouble(Console.ReadLine());  Console.WriteLine($"x^21 = {Power21(x)}");  Console.WriteLine("Press Enter");  Console.ReadLine();  }  } |

далее в строку пути к файлу пишем «powershell.exe»  
  
  
в powershell прописываем команды для clang:  
  
для запуска «./lab1c.exe»  


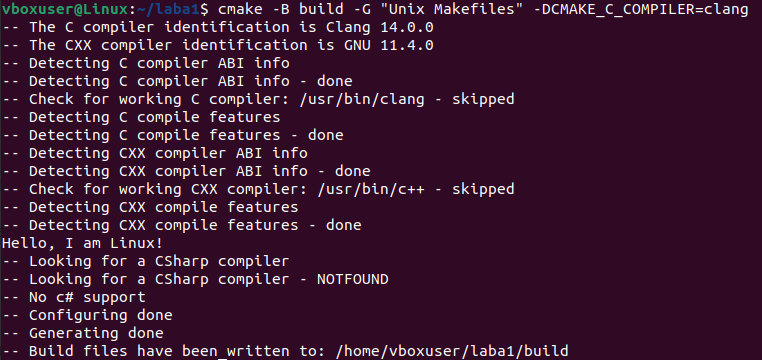
для cs на Winde:

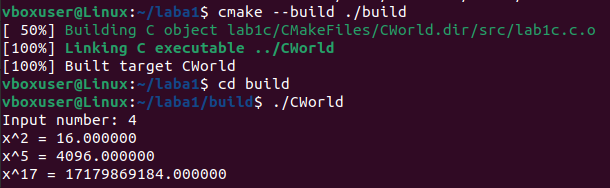


Заходим в папку build открываем cmd:  
«cmake -B ./build -G “Visual Studio 17 2022”» -T ClangCL .   
«cmake --build ./build»  
  
  
  
тоже самое по файлам как в windows но главный CMakeLists другой:

|  |
| --- |
| cmake\_minimum\_required(VERSION 3.22)  project(lab1)  set(EXECUTABLE\_OUTPUT\_PATH "${CMAKE\_BINARY\_DIR}")  # Определение ОС  if(WIN32)  message("Hello, I am Windows!")  elseif(APPLE)  message("Hello, I am macOS!")  elseif(UNIX)  message("Hello, I am Linux!")  endif()  # Добавляем C-программу всегда  add\_subdirectory(lab1c)  # Проверяем, можно ли собрать C#  if(WIN32) # Только Windows поддерживает C#  include(CheckLanguage)  check\_language(CSharp)    if(CMAKE\_CSharp\_COMPILER)  enable\_language(CSharp)  message(STATUS "C# поддерживается, добавляем в сборку")  add\_subdirectory(lab1cs)  else()  message(STATUS "C# не поддерживается")  endif()  else()  message(STATUS "C# отключен для Linux/macOS")  endif() |

Для того чтобы отключить c#  
  
последовательно:



  
  
  
  
clang –E lab1c.c –o lab1c.i

Clang –S lab1c.i –o lab1c.s

Clang –c lab1c.s –o lab1c.o

Clang lab1c.o –o lab1c.exe

Cmake –B build –G ”Visual Studio 17 2022” –T ClangCL .

Cmake –B build –G “Unix Makefiles” –DCMake\_C\_COMPILER=clang

Cmake –build ./build