

# Basics-SMTS-SMD

## 🔗 Как зовут преподавателя?

Федотов Илья Алексеевич

Основная коммуникация через telegram/vk (только через старосту)

Слайды: <https://github.com/lfedotov73/smts-smd/tree/main/coursework/slides>

Исходный код: <https://github.com/lfedotov73/smts-smd>)

## Как будет происходить курс по TSM и DSM

**SMTS** (Software Module Testing and Support, в учебном плане это "Поддержка и тестирование программных модулей") и **SMD**(Software Module Development, в учебном плане это "Разработка программных модулей").

Оба этих предмета очень связаны, поэтому мы попытаемся стереть грань между этими двумя дисциплинами и попробуем научиться писать программы так, чтобы они соответствовали и первой и второй дисциплине. Из чего следует то, что перечень лабораторных работ будет меньше (одна и также лабораторная работа должна быть защищена дважды), но при этом, сами лабораторные будут немного больше тех, как если бы второй дисциплины не существовало.

| Коротко: один список лабораторных работ на две дисциплины.

Всего на курсе будет около **8 лекций и 8 лабораторных работ**. Также после каждой лекции будет проводится небольшая самостоятельная работа, чтобы оценить качество восприятия материала.

## Что нужно сделать до начала лабораторных?

1. Зарегистрироваться (если еще нет) на GitHub: <https://github.com/>
2. Установить Git для своей системы по ссылке: <https://git-scm.com/install/>
3. Найти мой репозиторий на GitHub по ссылке <https://github.com/lfedotov73/smts-smd>
4. Установить WSL Ubuntu

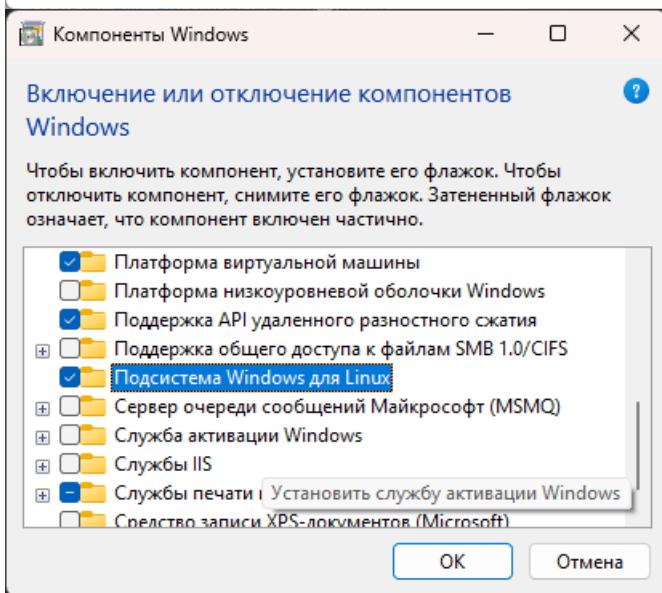
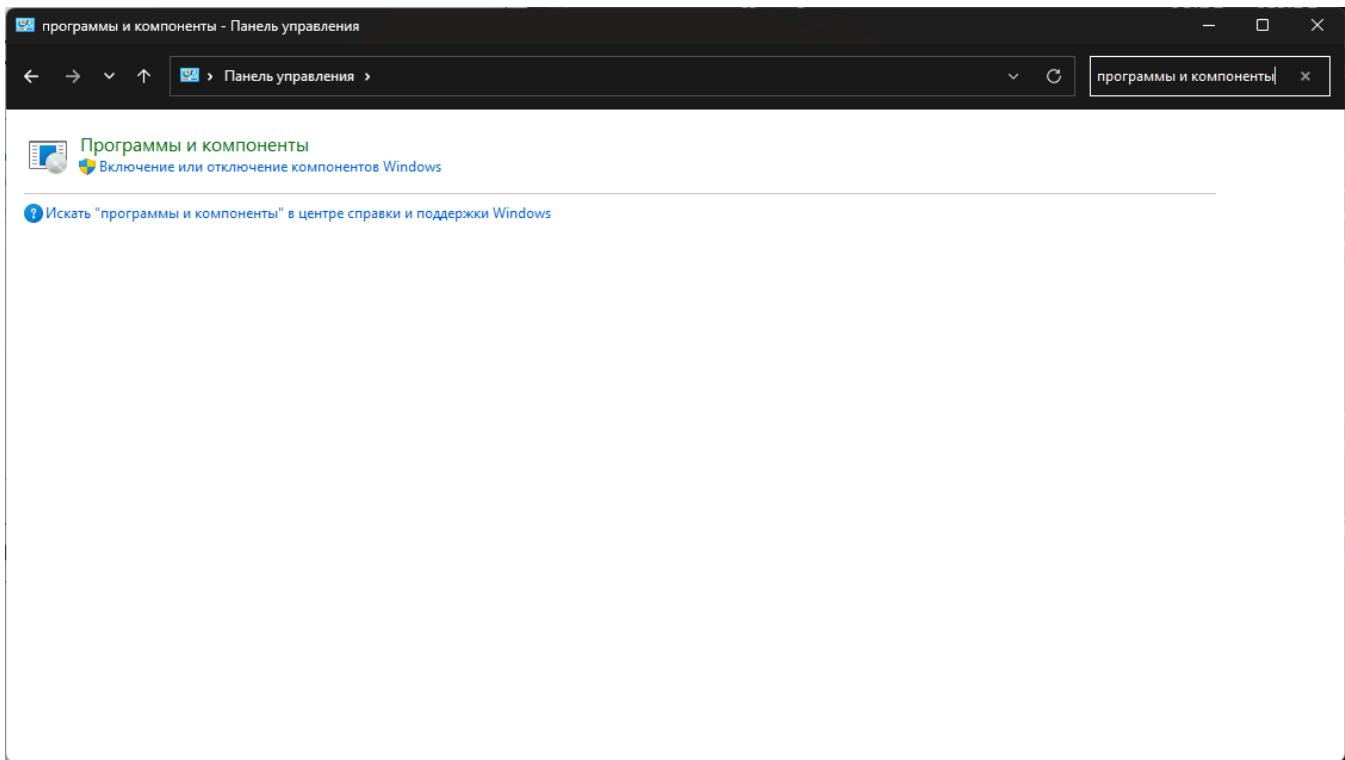
| На данный момент это все, что нам нужно для начала работы.

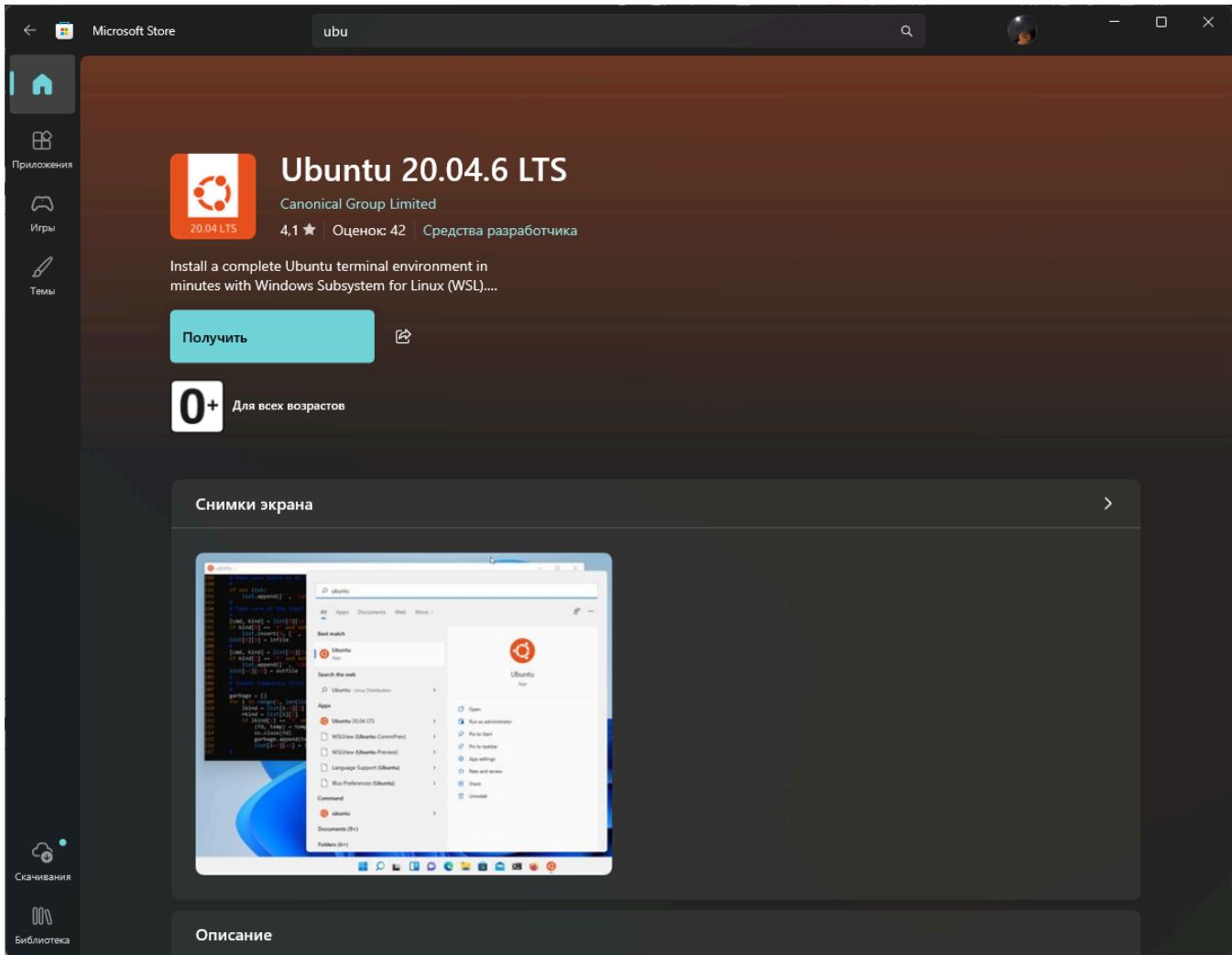
## Как начать в WSL под Windows

### 🔗 Почему WSL

Мы начнем курс так, будто мы мало чего знаем о программировании и совершенно не хотим перегружать свою голову особенностями различных IDE, систем сборок и т.д. В дальнейшем мы будем переходить на что-то новое, но сейчас нам достаточно создать файл на языке программирования C, скомпилировать его, и запустить.

| Также я надеюсь, что у всех установлена операционная система Windows не старше Windows 10, если это не так, то устанавливайте Ubuntu или более новую версию Windows.





## Ubuntu - быстрый старт.

Работа с пакетами требует привилегий sudo.

1. sudo apt update обновляет эту базу данных, чтобы твой компьютер знал обо всех последних изменениях в репозиториях, например, о новых пакетах или новых версиях старых пакетов.
2. sudo apt upgrade сравнивает твою локальную базу данных со списком пакетов, которые у тебя уже установлены, и обновляет установленные пакеты, которые старше, чем те, что есть в репозиториях.

Для установки программы используется sudo apt install .

Необходимо ввести поочередно в Ваш терминал Ubuntu.

```
sudo apt update // обновляем базу данных  
sudo apt install gcc // устанавливаем компилятор gcc
```

Как результат: можно проверить версию установленного компилятора через команду gcc --version .

```
ilfe@DESKTOP-955S1HI:~$ gcc --version  
gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0  
Copyright (C) 2019 Free Software Foundation, Inc.  
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

## Первая программа

Теперь есть практически все, чтобы написать классическую программу в программировании: hello.c.

Но предварительно давайте заведем рабочую директорию Dev по адресу C:/Dev .

В этой директории мы будем заводить наши проекты. Теперь создадим папку проекта HelloWorld по адресу

C:/Dev/HelloWorld .

Теперь необходимо найти эту папку из Windows в нашей Linux-системе, делается это через команду: cd /mnt/c/dev/... , после чего написать, и скомпилировать нашу первую программу через компилятор gcc: gcc hello.c . В итоге в папке с файлом **hello.c** появился скомпилированный файл, который по умолчанию называется **a.out**. Запустим его с помощью следующей команды: ./a.out .

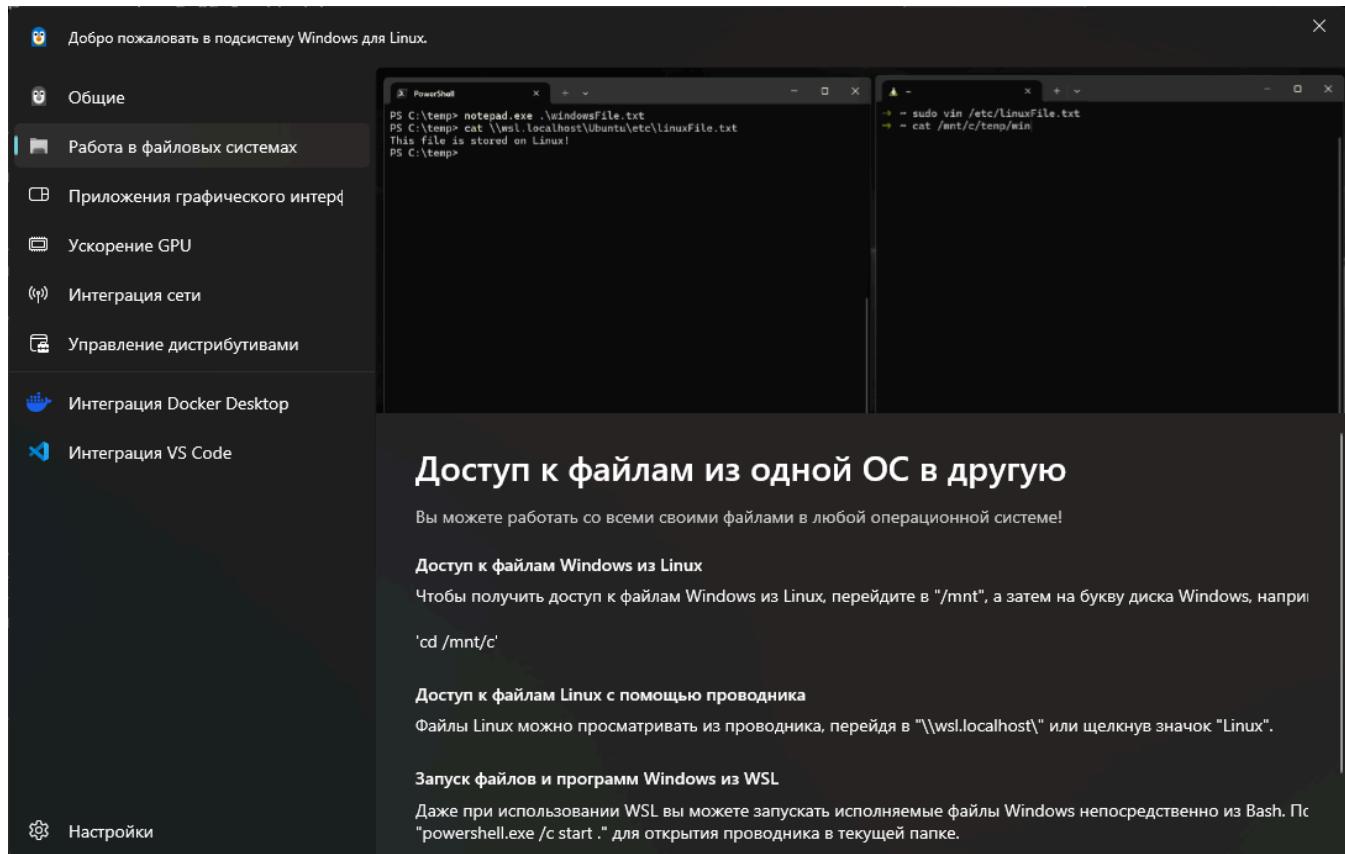
Код нашей первой программы:

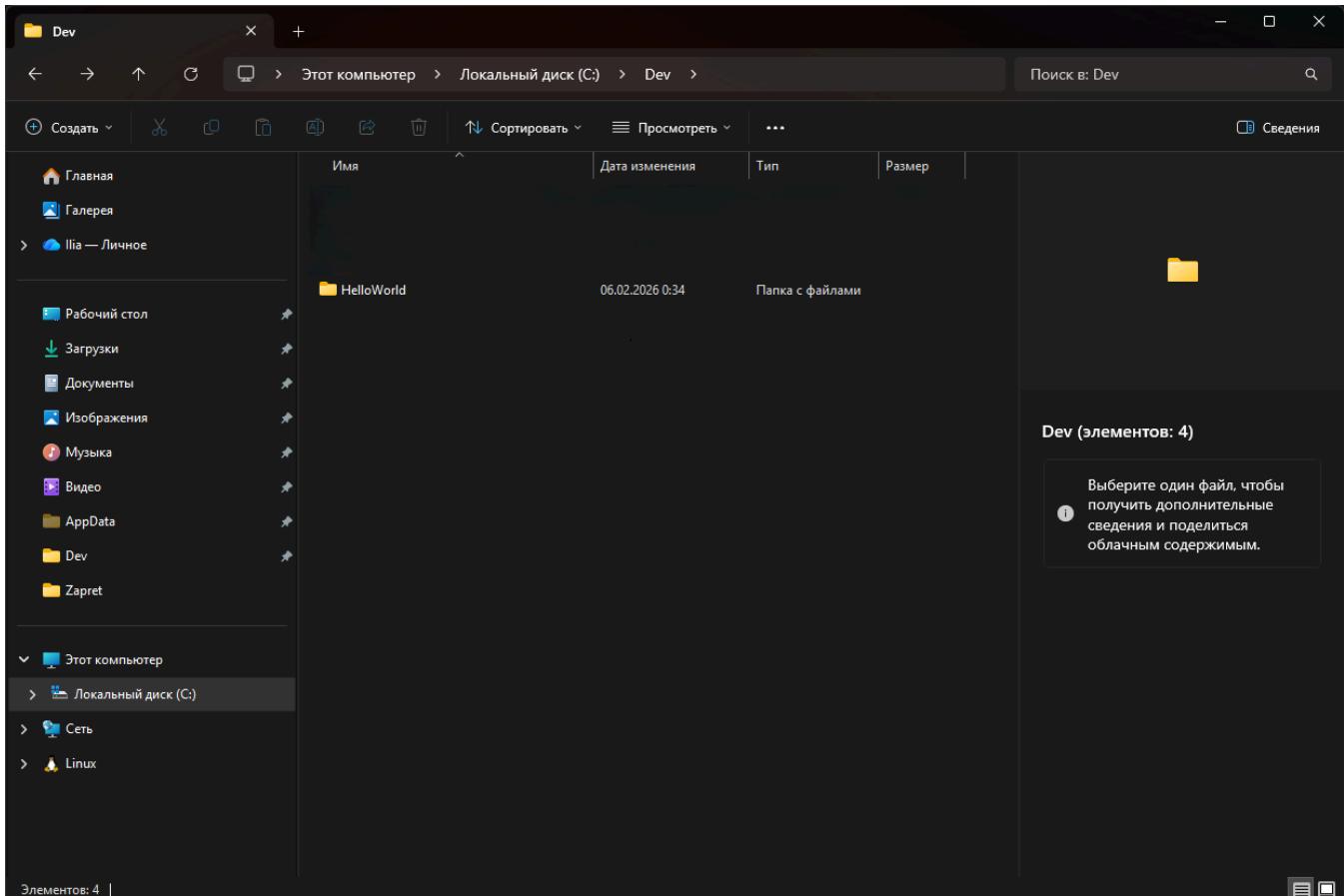
```
#include <stdio.h>

int main()
{
    printf("Hello, World!");
}
```

В качестве редактора для редактирования файла исходного кода я использую **vim**. Чтобы создать файл **hello.c** написать vim hello.c . Файл будет автоматически создан и открыт в редакторе vim. Для редактирования кода в vim необходимо перейти в **insert mode** , для этого необходимо нажать клавишу I (или кнопку с русской Щ ). Как только мы закончили редактировать файл, нажимаем Esc , затем : (русская Ж ), вводим wq (запись и выход) и нажимаем Enter .

Все операции выполняются в английской раскладке!!!.



A screenshot of a terminal window with the title 'ilfe@DESKTOP-955S1HI: /mnt/'. The window contains the following C code:

```
#include <stdio.h>

int main()
{
    printf("Hello, World!");
}
```

A screenshot of a terminal window with the title 'ilfe@DESKTOP-955S1HI: ~\$'. The window shows the following command-line session:

```
ilfe@DESKTOP-955S1HI: ~$ cd /mnt/c/dev/helloworld
ilfe@DESKTOP-955S1HI:/mnt/c/dev/helloworld$ vim hello.c
ilfe@DESKTOP-955S1HI:/mnt/c/dev/helloworld$ gcc hello.c
ilfe@DESKTOP-955S1HI:/mnt/c/dev/helloworld$ ls
a.out  hello.c
ilfe@DESKTOP-955S1HI:/mnt/c/dev/helloworld$ ./a.out
Hello, World! ilfe@DESKTOP-955S1HI:/mnt/c/dev/helloworld$ |
```

## Полезные источники

1. <https://metanit.com/c/> -- руководство по языку программирования С
2. Книга “Язык программирования С”. Кернigan Брайан, Ритчи Деннис
3. <https://metanit.com/cpp/> -- руководство по языку программирования С++

4. <https://metanit.com/sharp/> -- руководство по языку программирования C#