

For Lab 1, we require

1. A compiler for our C code (GCC)
2. A debugger tool (GDB/LLDB)

Instructions in Lab1 document will assume you are using GCC for the compiler and GDB for debugger. However, if you are using a similar compiler & debugger, you should still be able to do the lab.

1. GNU Compiler Collection (GCC)

GCC is an open source compiler system used to compile C/C++ programs: <https://gcc.gnu.org/>

Mac users should already have GCC, otherwise installing it should be simple anyway.

For Windows users, there are many ways to install GCC on your device. I would strongly recommend getting WSL <https://learn.microsoft.com/en-us/windows/wsl/install> first, which installs an Ubuntu distribution of Linux on your Windows on which you can install GCC (and future Linux packages) much more easily. Effectively, you are now running Linux while in WSL, hence Linux install instructions can be followed.

Alternatively, you can use Cygwin (mentioned below) or MinGW.

2. GNU Debugger (GDB) <https://www.gnu.org/software/gdb/>

A debugger is used to analyze program execution in a step-by-step and detailed manner. It is used to find bugs in a program. Using a debugger, we can execute a program partially and view the status of the variables and resources being used the program to identify any discrepancies. **GDB** is an open source, freely available debugger which can be used for multiple languages.

GDB can do four main kinds of things (plus other things in support of these) to help you catch bugs in the act:

3. Start your program, specifying anything that might affect its behaviour.
4. Make your program stop on specified conditions.
5. Examine what has happened, when your program has stopped.
6. Change things in your program; so that you can experiment with correcting the effects of one bug and go on to learn about another.

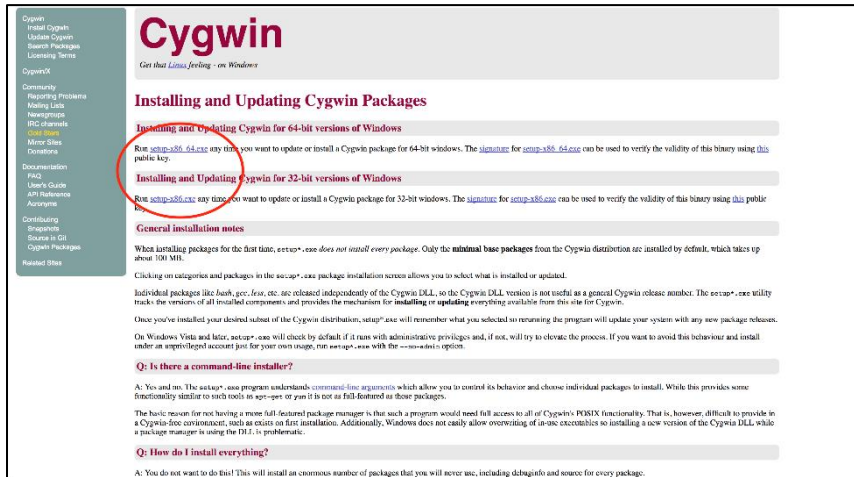
To install GDB:

1. **Ubuntu:** `sudo apt-get install gdb`
2. **OSX:** `brew install gdb`
3. **Windows:**

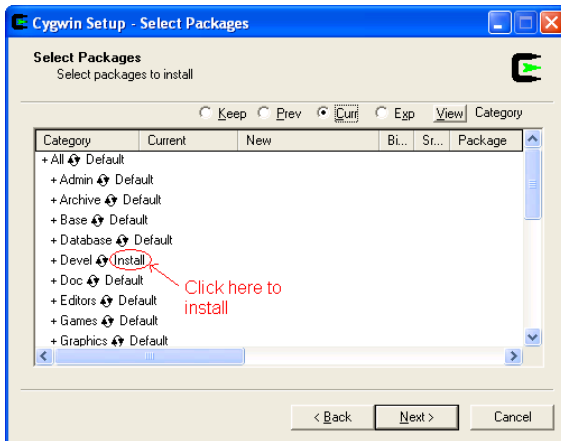
If you are using WSL, simply follow the instruction above for **Ubuntu** (`sudo apt-get install gdb`) after entering WSL.

Alternatively you can use Cygwin as follows:

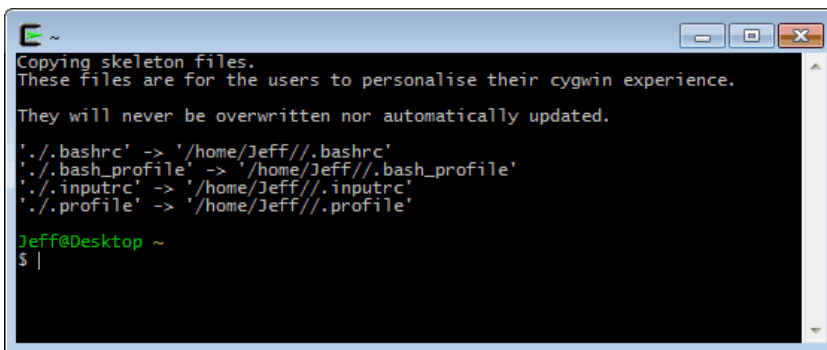
- a. Download Cygwin (<https://cygwin.com/install.html>)



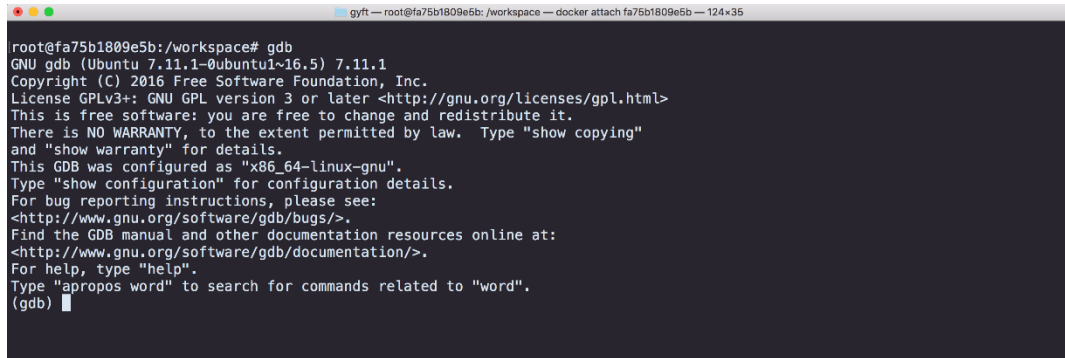
- b. Select **develop** packages during installation:



- c. Start Cygwin terminal from the start menu:



4. Starting GDB (all platforms): Type GDB in the terminal (WSL/Cygwin terminal for windows users). **Help** command lists the help and **quit** command exits GDB.



```

root@fa75b1809e5b:/workspace# gdb
GNU gdb (Ubuntu 7.11.1-0ubuntu1~16.5) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
(gdb)

```

APPLE SILICON MACOS USERS (M1 onwards)

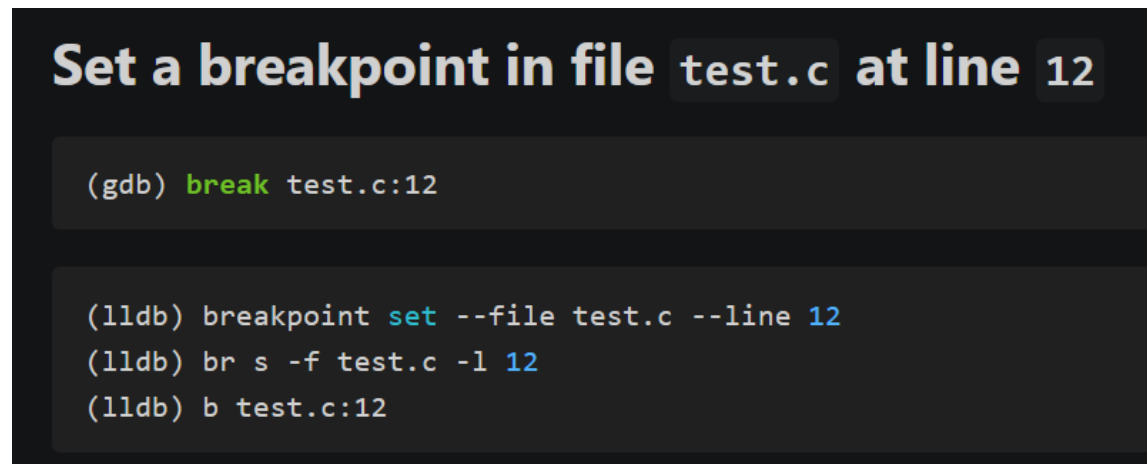
GDB does not work on MacOS on Apple Silicon (M1, M2, etc) devices. If you are using such a system, we can use LLDB instead, the default debugger on Mac.

Both GDB and LLDB are great debugger tools, and the majority of GDB and LLDB commands are the same, just with different names. The lab sheet contains several GDB commands, you will need to find the equivalent command on LLDB.

To do so, refer to this map from GDB commands to LLDB commands:

<https://lldb.lvm.org/use/map.html#breakpoint-commands>

Eg. Setting breakpoint at a specific line:



```

Set a breakpoint in file test.c at line 12

(gdb) break test.c:12

(lldb) breakpoint set --file test.c --line 12
(lldb) br s -f test.c -l 12
(lldb) b test.c:12

```

The above is a screenshot from the GDB-LLDB command map (above link). Here, instead of the gdb command (**break test.c:12**), we use instead the equivalent lldb command (**b test.c:12**).

If you are more interested in how LLDB works, you can find an LLDB tutorial here:

<https://lldb.lvm.org/use/tutorial.html>