

本講義では、コンピュータ基礎で構築した Linux 環境を使用する。

プログラムの作成・編集は、【emacs】を利用することを推奨する。

C 言語プログラムを実行するために、【gcc】コマンドのインストールを行う。【gcc】については、第 2 回講義で説明する。

使用する OS (Windows / Mac / Linux) に応じて、手順に従い、環境を構築すること。

1. 【Windows 版】WSL 環境の利用 (【】は入力コマンドを示す。【】を除いて実行すること。)

1-1. Ubuntu を起動後、以下のコマンド (ソフトウェア更新のため) を実行する。

【sudo apt update】

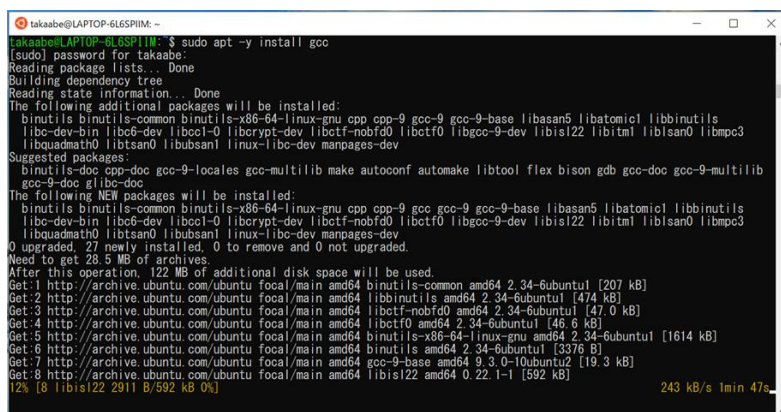
【sudo apt upgrade】

1-2. 上記コマンドを実行時に、管理者のパスワード入力を行う。

1-3. **【sudo apt -y install gcc】** とコマンド入力する。

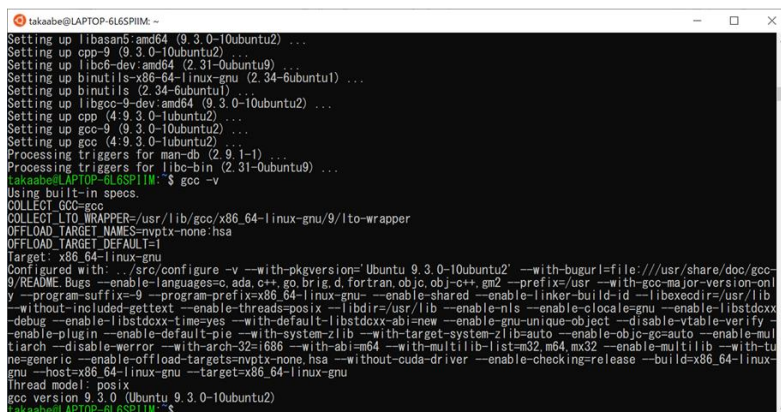
1-4. インストール完了後、**【gcc -v】** とコマンド入力する。

コマンドの実行時の画面例、入力完了と確認画面例を図 1, 2 に示す。



```
takaabe@LAPTOP-6L6SP1M: ~$ sudo apt -y install gcc
[sudo] password for takaabe:
Reading state information... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 gcc-9 gcc-9-base libasan5 libatomic1 libbinutils
  libc-dev-bin libc6-dev libcbl-0 libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev libisl22 libitm1 liblsan0 libmpc3
  libquadmath0 libtsan0 libubsan1 linux-libc-dev manpages-dev
Suggested packages:
  binutils-doc cpp-doc gcc-9-locales gcc-multilib make autoconf automake libtool flex bison gdb gcc-doc gcc-9-multilib
  gcc-9-doc glibc-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 gcc gcc-9 gcc-9-base libasan5 libatomic1 libbinutils
  libc-dev-bin libc6-dev libcbl-0 libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev libisl22 libitm1 liblsan0 libmpc3
  libquadmath0 libtsan0 libubsan1 linux-libc-dev manpages-dev
0 upgraded, 27 newly installed, 0 to remove and 0 not upgraded.
Need to get 28.5 MB of archives.
After this operation, 122 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/main amd64 binutils-common amd64 2.34-6ubuntu1 [207 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal/main amd64 libbinutils amd64 2.34-6ubuntu1 [474 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal/main amd64 libctf-nobfd0 amd64 2.34-6ubuntu1 [47.0 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal/main amd64 libctf0 amd64 2.34-6ubuntu1 [46.6 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal/main amd64 binutils-x86-64-linux-gnu amd64 2.34-6ubuntu1 [1614 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal/main amd64 binutils amd64 2.34-6ubuntu1 [1376 B]
Get:7 http://archive.ubuntu.com/ubuntu focal/main amd64 gcc-9-base amd64 9.3.0-10ubuntu2 [19.3 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal/main amd64 libisl22 amd64 0.22.1-1 [592 kB]
12% [8 libisl22 2911 B/592 kB 0%] 243 kB/s 1min 47s
```

図 1. インストールコマンド実行時画面



```
takaabe@LAPTOP-6L6SP1M: ~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/9/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:hsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 9.3.0-10ubuntu2' --with-bugurl=file:///usr/share/doc/gcc-9/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++,gm2 --prefix=/usr --with-gcc-major-version-only --program-suffix=-9 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-objc-gate --enable-multilib --enable-lto --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-plugin --enable-default-pie --with-system-zlib --with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch --disable-werror --with-arch=32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none:hsa --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 9.3.0 (Ubuntu 9.3.0-10ubuntu2)
```

図 2. インストール完了と確認画面

2. 【Mac 版】

2-1. アップルストアにて、【xcodes】アプリをインストールする (図1).

サイズが大きいのので、ダウンロード、インストールに時間がかかるので、注意すること.

2-2. インストール完了後、ターミナルを開いて、【gcc-v】をコマンド入力する (図2).

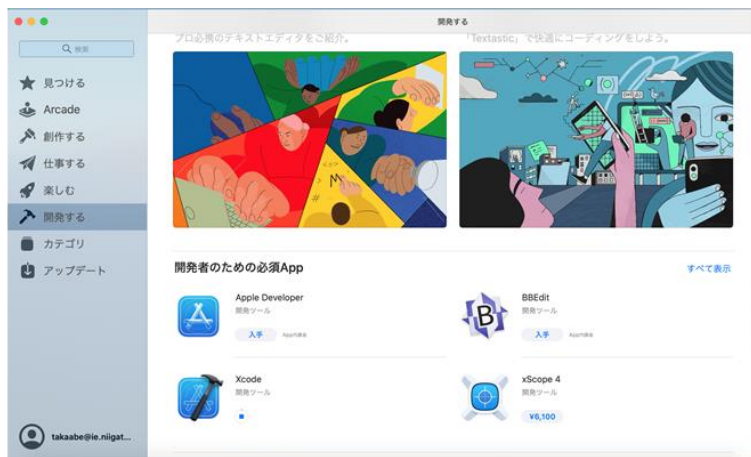


図3. Appleストアでの開発ツールの検索画面

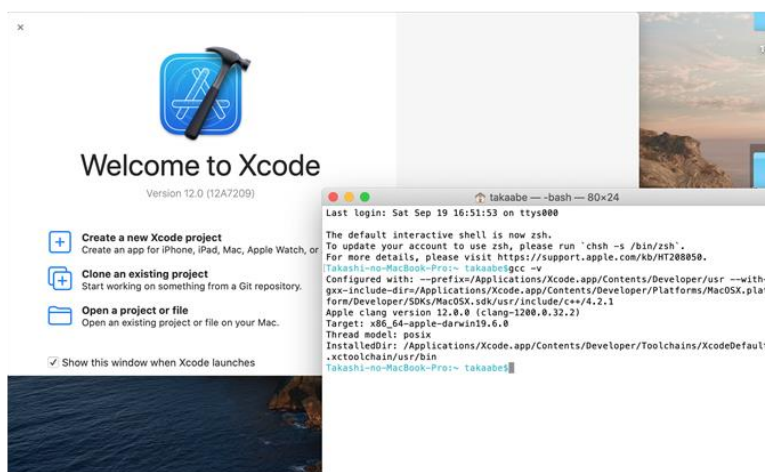


図4. Xcodeの画面とターミナルでの【gcc-v】の実行例

3. 【Linux 版】

基本的には、【gcc】はインストール済みのはずである。

もし【gcc -v】とコマンド入力して、エラーが表示された場合、gcc パッケージをインストールして下さい。

4. その他 : Web ブラウザを介して, 大学の計算機演習環境を利用する.

上記 1 ~ 3 で環境構築を行うことができない場合, 大学の計算機演習室環境を利用できる. 利用方法については, 別途, 個別に説明するので, 講義担当教員へ連絡すること.

5. 【非推奨】 その他 : Web ブラウザを利用したプログラミング

上記 1 ~ 3 以外での C 言語のプログラミング環境として, Web 上に公開された C 言語のオンラインコンパイラを利用することもできる (図 5).

--URL

<https://repl.it/languages/c>

--

大学のメールアドレス (gmail アカウント) でログインすれば, ファイルの保存も可能である. また, C 言語以外のプログラミング言語も利用可能なので, 興味のある方はお試しください (図 6).

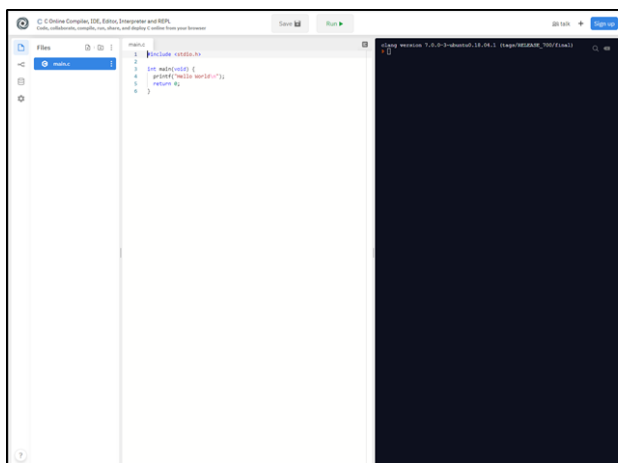


図 5. 参考サイト画面

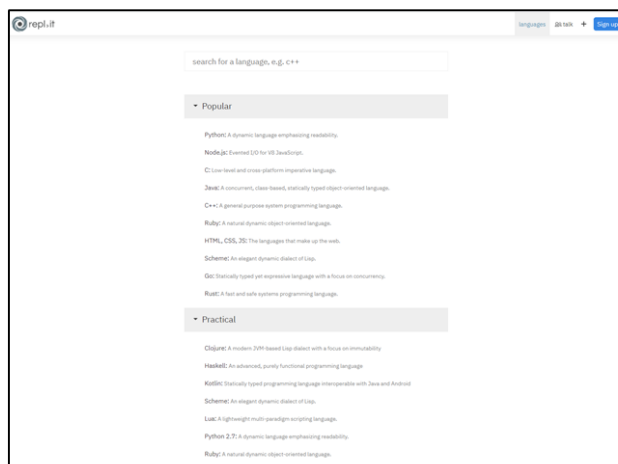


図 6. 利用可能なプログラミング言語