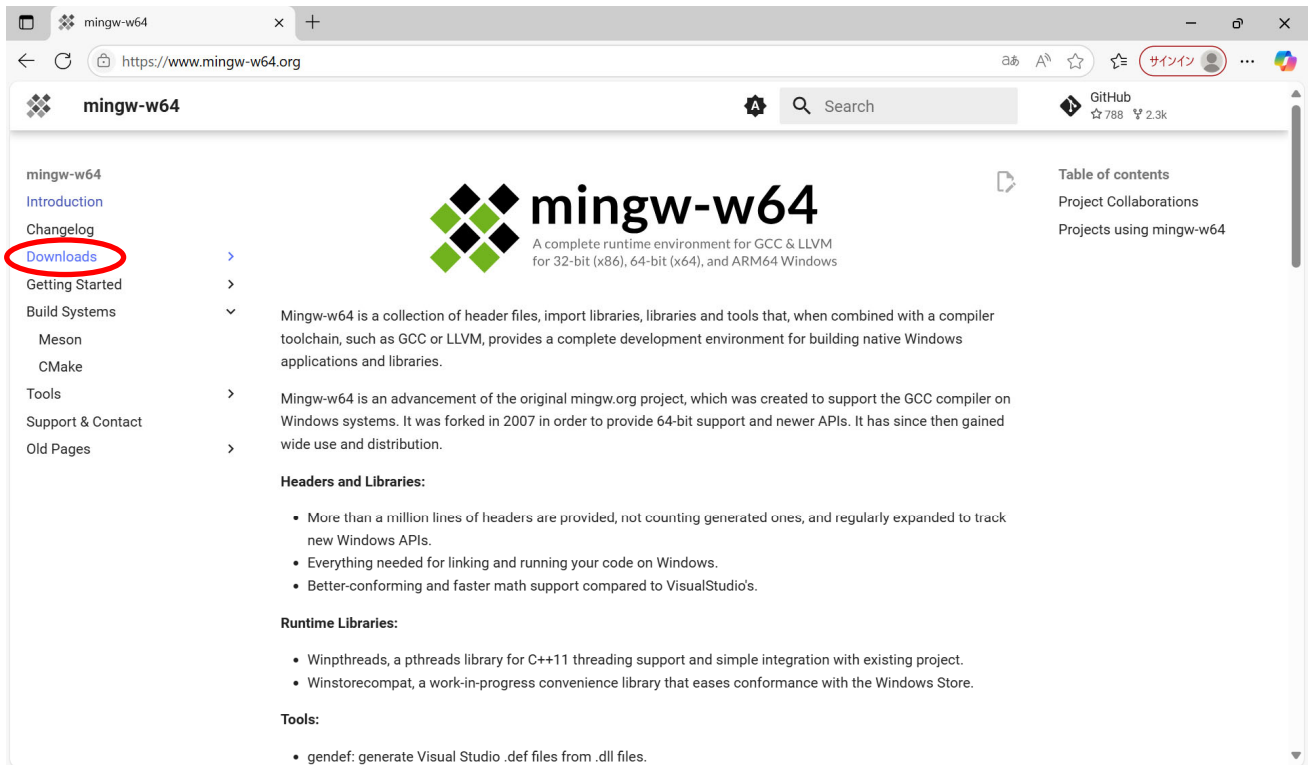


## 1. MinGW-w64 のダウンロード

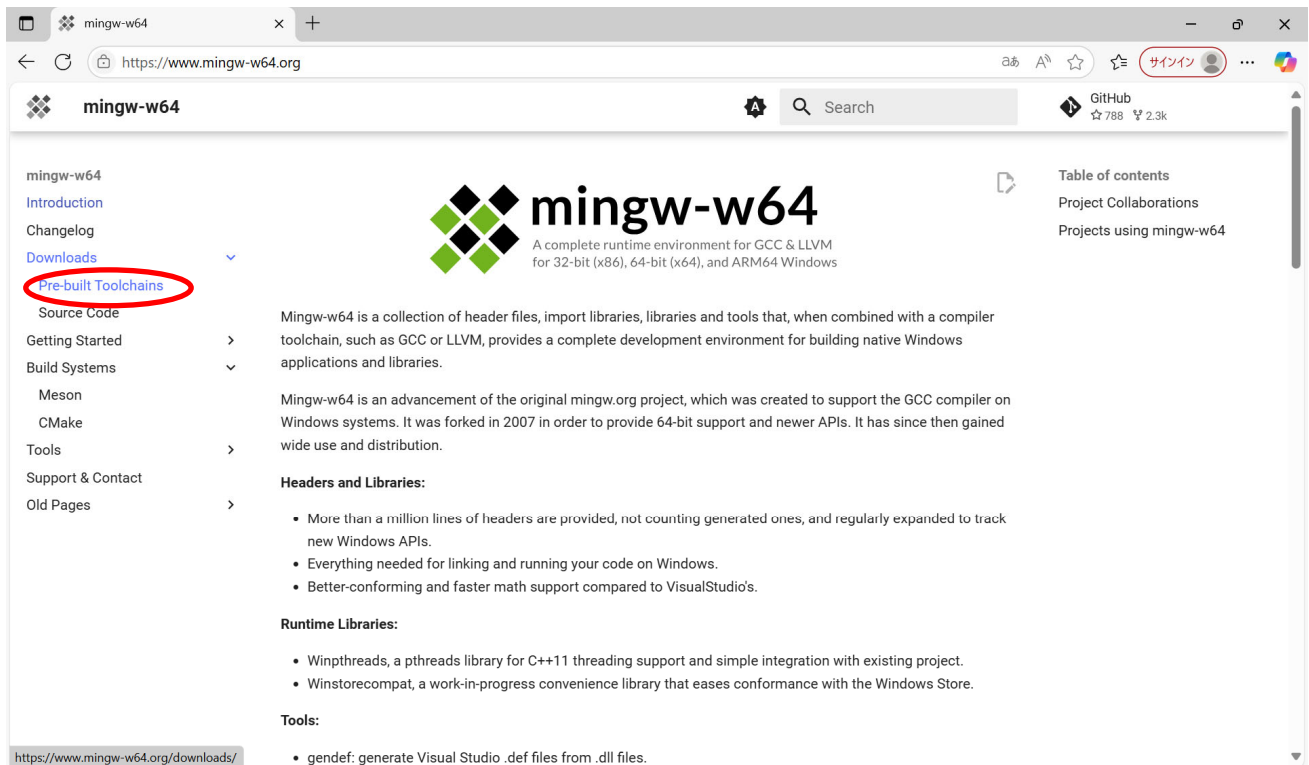
(1) web ブラウザで下記サイトをアクセスします。以下は、Edge の例です。

<http://mingw-w64.org/>

(2) 「Downloads」をクリックします。



(3) 「Pre-built Toolchain」をクリックします。



(4) 下にスクロールし「MinGW-W64-builds」をクリックします。

The screenshot shows the 'Pre-built Toolchains' page on the MinGW-w64 website. The page lists various pre-built toolchains for different operating systems and architectures. The 'MinGW-W64-builds' link is circled in red.

| Toolchain        | OS                    | Architecture | Version           | Supported Languages                         | Package Manager                                                                     |
|------------------|-----------------------|--------------|-------------------|---------------------------------------------|-------------------------------------------------------------------------------------|
| Fedora 40        | Fedora                | x86_64       | 14.1.1/11.0.1     | Ada, C, C++, Fortran, Obj-C, Obj-C++        | many                                                                                |
| Fedora 41        | Fedora                | x86_64       | 14.2.1/12.0.0     | Ada, C, C++, Fortran, Obj-C, Obj-C++        | many                                                                                |
| Homebrew         | macOS                 | x86_64       | 15.1.0/13.0.0     | C, C++, Fortran, Obj-C, Obj-C++             | 1 (nsis)                                                                            |
| LLVM-MinGW       | Windows, Linux, macOS | x86_64       | 20250613          | C, C++                                      | make, Python                                                                        |
| MacPorts         | macOS                 | x86_64       | 15.1.0/12.0.0     | C, C++, Fortran, Obj-C, Obj-C++             | 1 (nsis)                                                                            |
| MinGW-W64-builds | Windows               | x86_64       | 15.1.0/12.0.0     | C, C++, Fortran                             | 4 (gdb, libconf, python, zlib)                                                      |
| MSYS2            | Windows               | x86_64       | 15.1.0/trunk      | Ada, C, C++, Fortran, Obj-C, Obj-C++, OCaml | many                                                                                |
| Ubuntu           | Ubuntu                | x86_64       | 20.04 Focal Fossa | Ada, C, C++, Fortran, Obj-C, Obj-C++        | 9 (gdb, libassuan, libcrypt, libgpg-error, libksba, libnpth, nsis, win-iconv, zlib) |

(5) 「GitHub」をクリックします。

The screenshot shows the 'MinGW-W64-builds' page on the MinGW-w64 website. The page provides information about the MinGW-W64-builds toolchain, including its installation instructions and supported languages. The 'Installation: GitHub' link is circled in red.

**MinGW-W64-builds**

Installation: [GitHub](#)

**MSYS2**

Installation: [GitHub](#)

**Ubuntu**

Installation: through integrated package manager.

[mingw-w64 packages on Ubuntu](#)

**w64devkit**

[w64devkit](#) is a portable C and C++ development kit for x64 (and x86) Windows.

Included tools:

- mingw-w64 GCC : compilers, linker, assembler
- GDB : debugger
- GNU Make : standard build tool
- busybox-w32 : standard unix utilities, including sh
- Vim : powerful text editor
- Universal Ctags : source navigation

The toolchain includes pthreads, C++11 threads, and OpenMP. All included runtime components are static.

Installation: [GitHub](#)

[WinLibs.com](#)

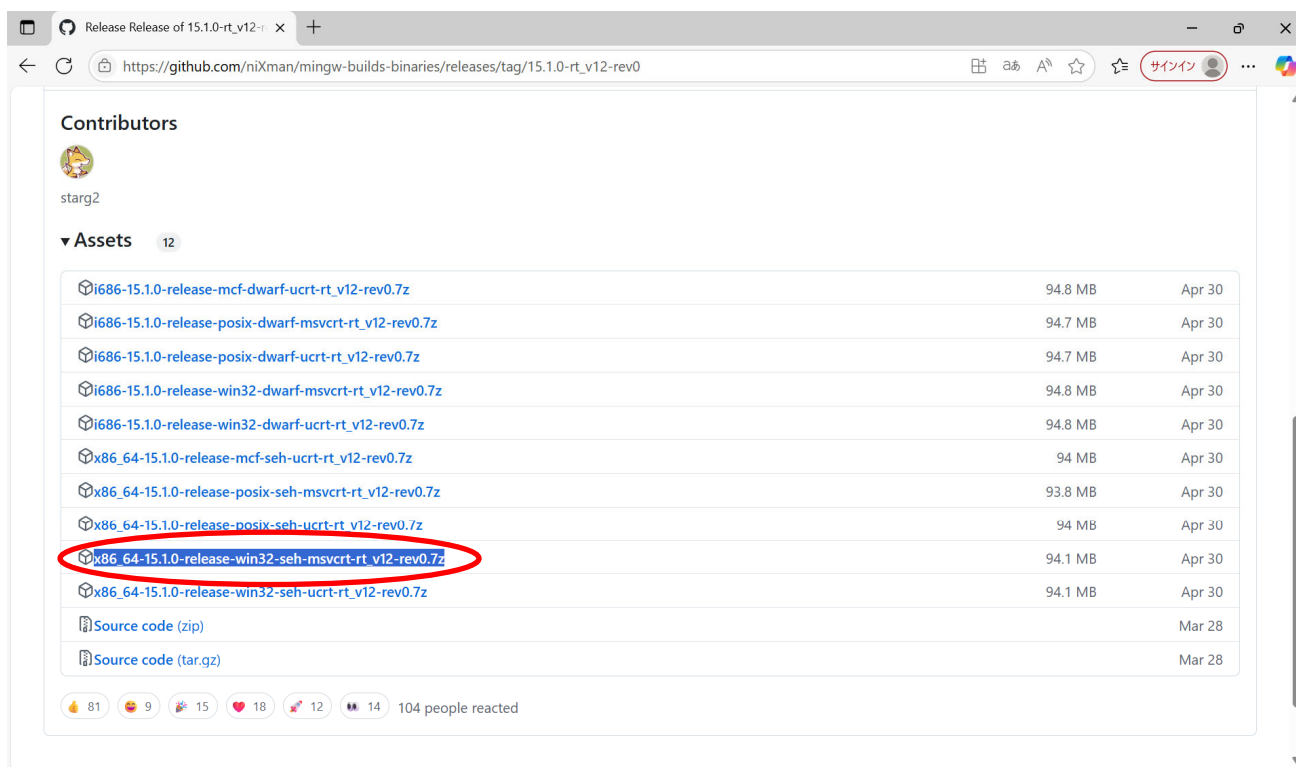
(6) 「Release of 15.1.0-rt\_v12-rev0」をクリックします。

The screenshot shows the GitHub interface for the repository `niXman / mingw-builds-binaries`. The page is displaying the 'Releases' tab. The latest release, 'Release of 15.1.0-rt\_v12-rev0', is highlighted with a red circle. The release was published on April 29 by user `niXman`. The release notes list several updated dependencies:

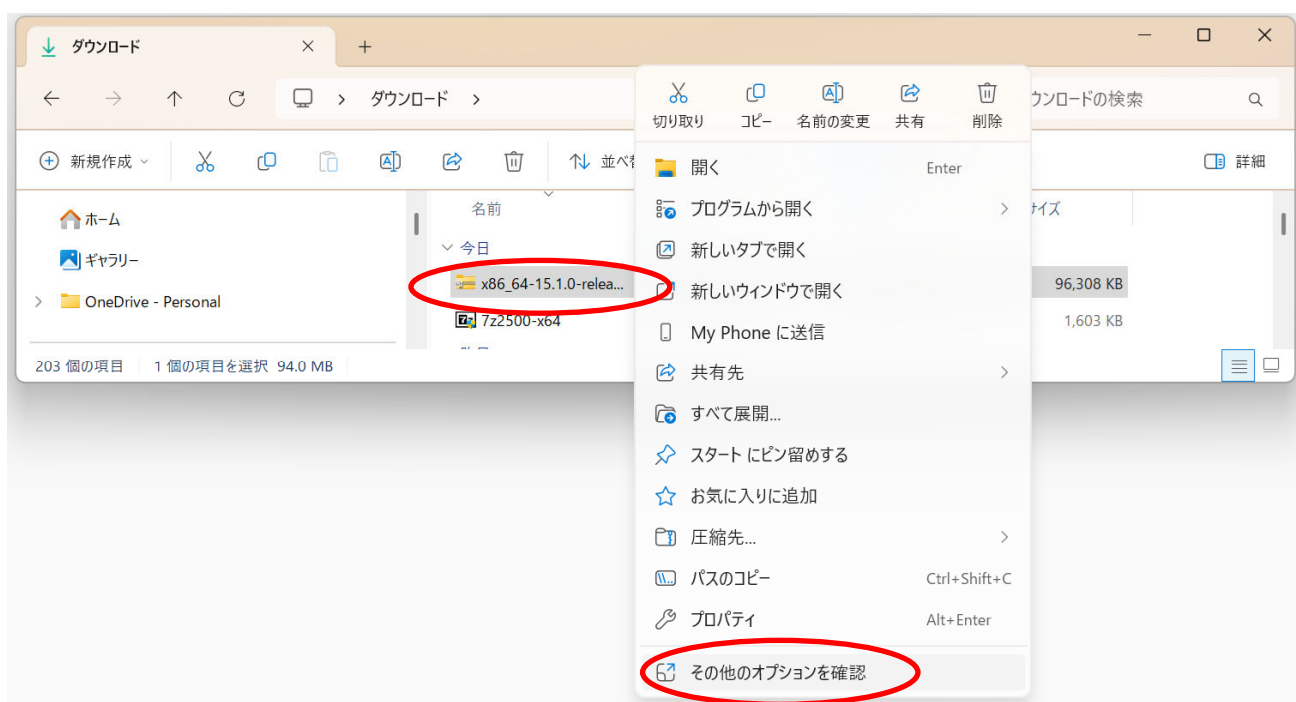
- `isl` updated to `0.27`
- `expat` updated to `2.7.1`
- `xz-utils` updated to `5.8.1`
- `python-3` updated to `3.12.10`
- `libffi` updated to `3.4.8`
- `sqlite` updated to `3.49.1`
- `zlib` updated to `1.3.1`
- `gcc` updated to `15.1.0`

At the bottom of the release notes, it says: 'Many thanks for this release to `@starg2`'.

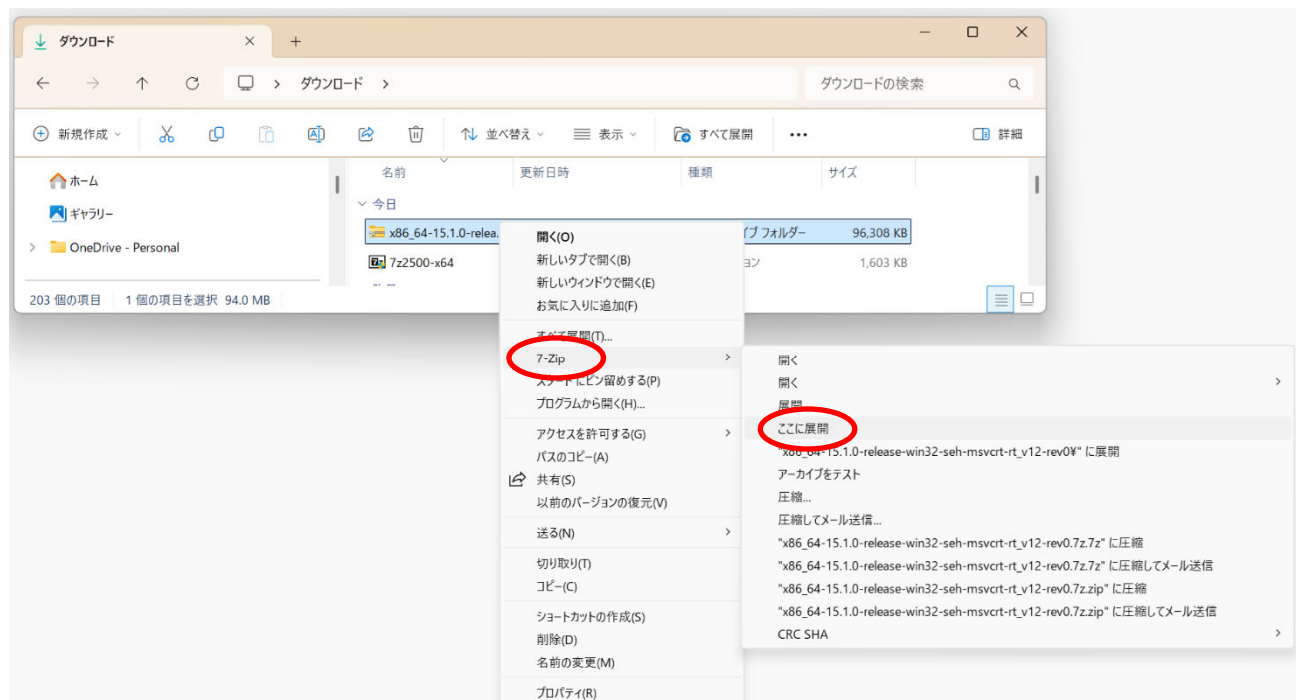
(7) 「x86\_64-15.1.0-release-win32-seh-msvcrt-rt\_v12-rev0.7z」をクリックします。



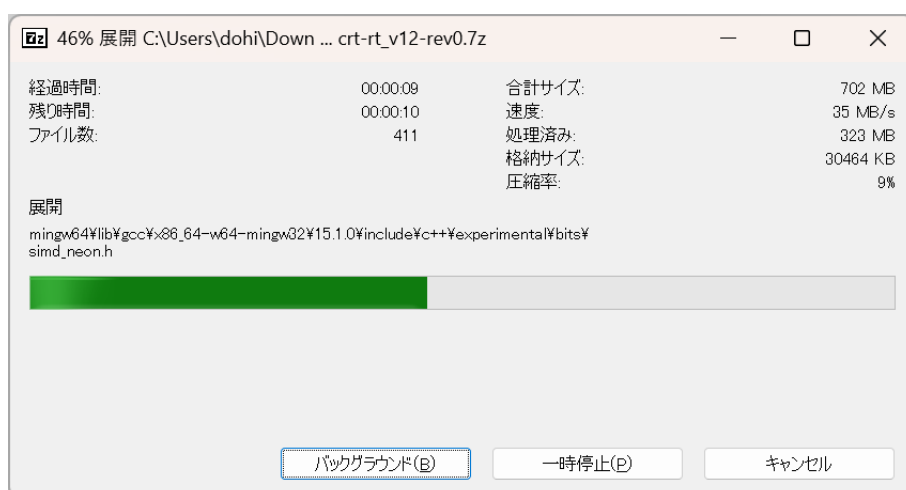
(8) ダウンロードが完了したらダウンロードフォルダの「x86\_64-15.1.0-release-win32-seh-msvcrt-rt\_v12-rev0」を右クリックし、「その他のオプションを確認」をクリックします。



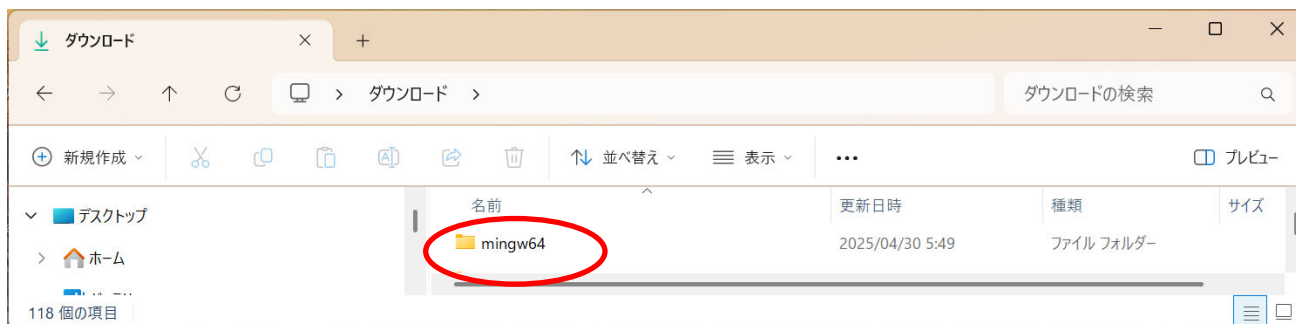
(9) 「7-Zip」「ここに展開」の順にクリックします。



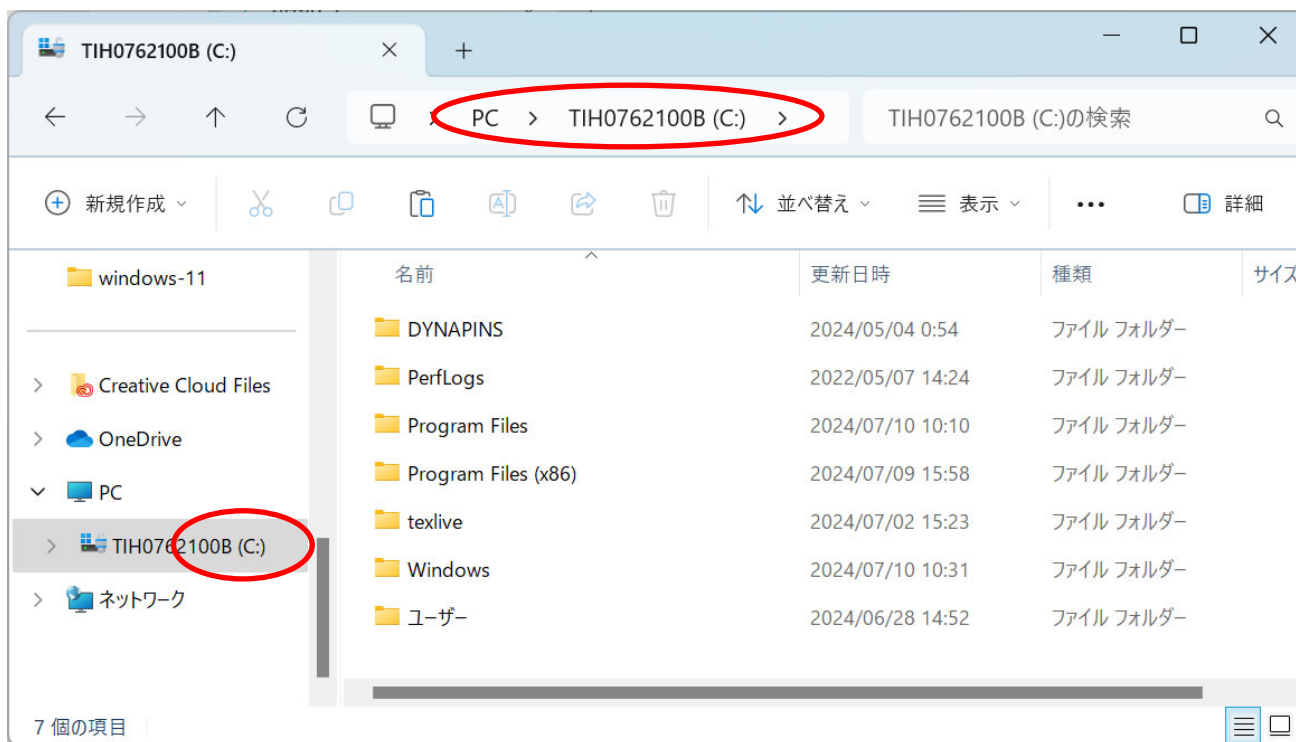
(10) 展開が始まります。展開には5分程度かかります。



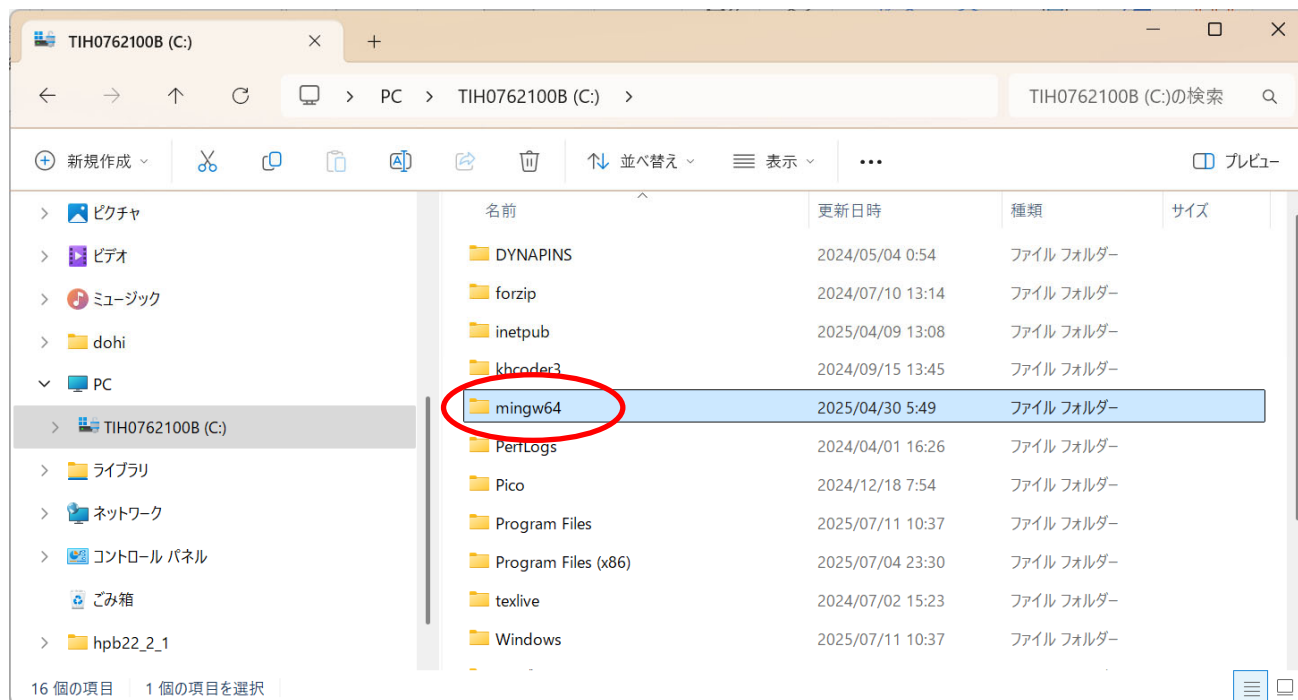
- (11) ダウンロードフォルダに展開された「mingw64」をクリックし、「Ctrl + X」でクリップボードに切り取ります。



- (12) Cドライブをクリックしルートディレクトリに移動します。

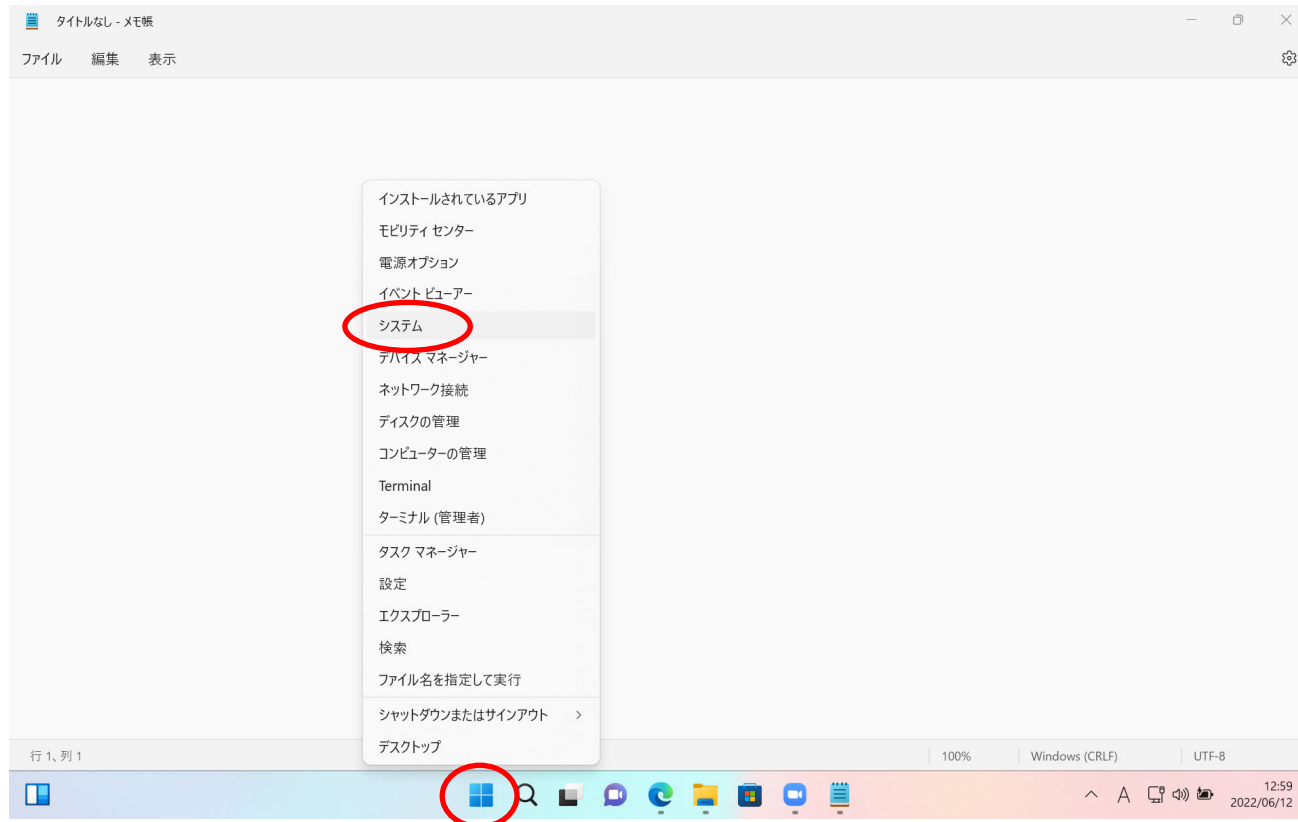


- (13) Cドライブのルートディレクトリに、「Ctrl + V」で貼り付けます。mingw64 のフォルダが表示されたことを確認します。

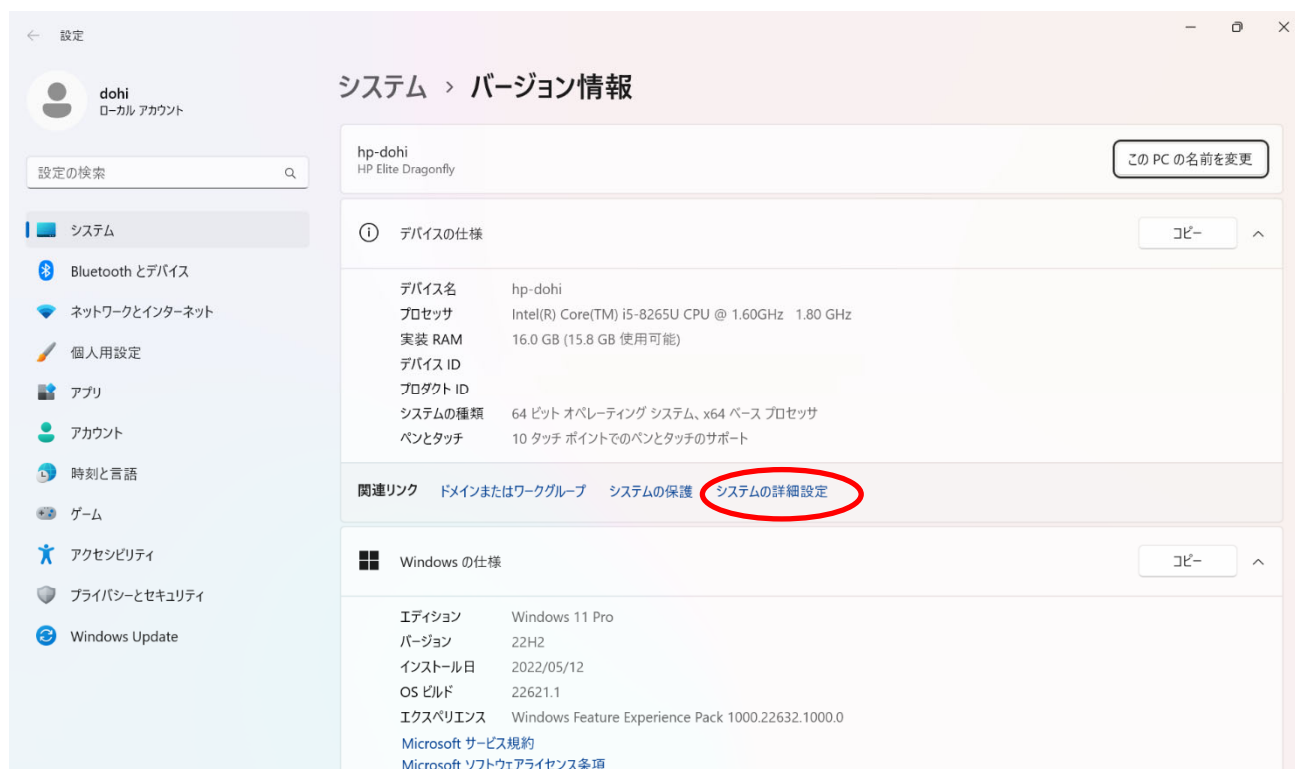


## 2. 環境変数の設定

- (1) 「スタート」を右クリックし「システム」をクリックします。



(2) 「システムの詳細設定」をクリックします。

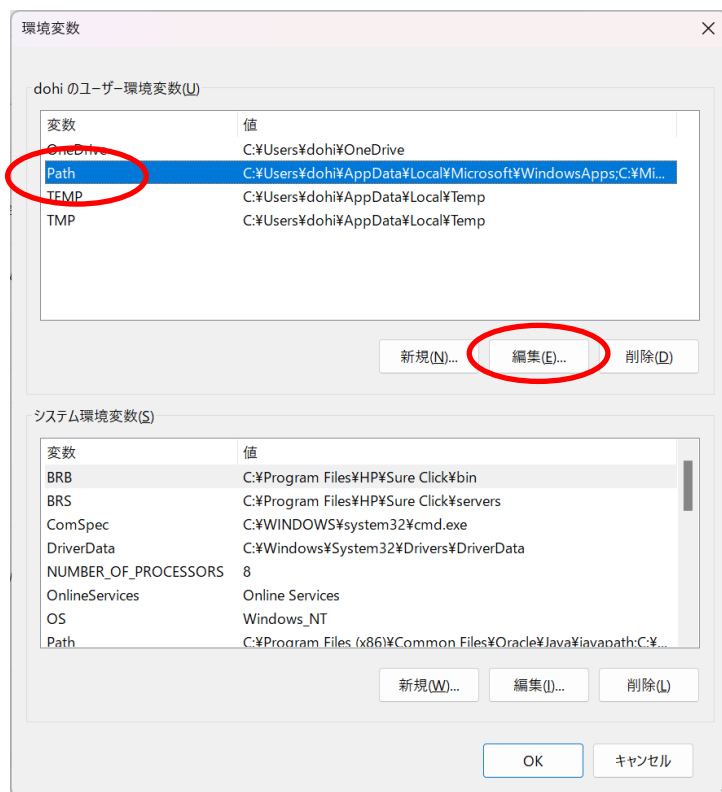


(3) 「環境変数」をクリックします。

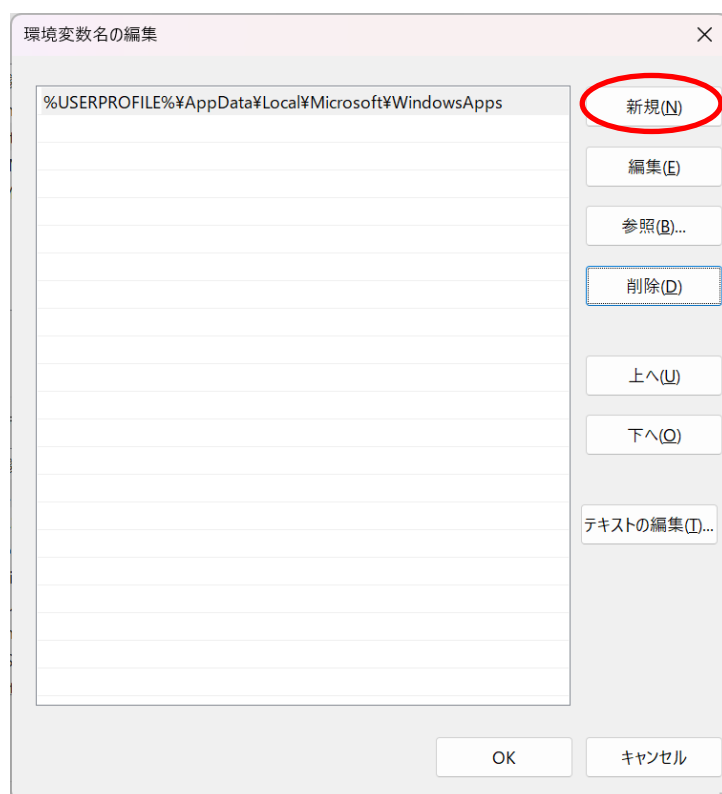




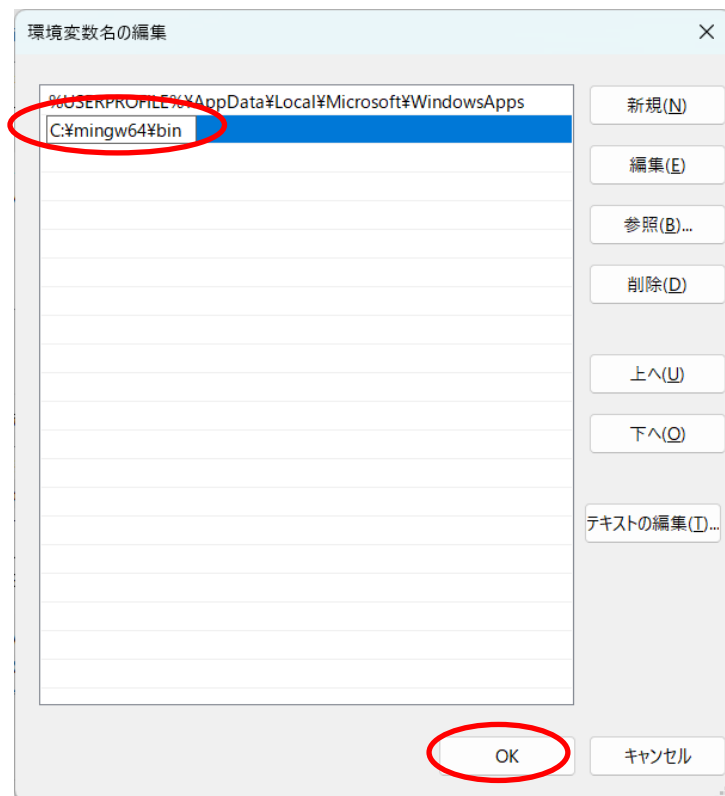
(4) 「Path」をクリックし、「編集」をクリックします。



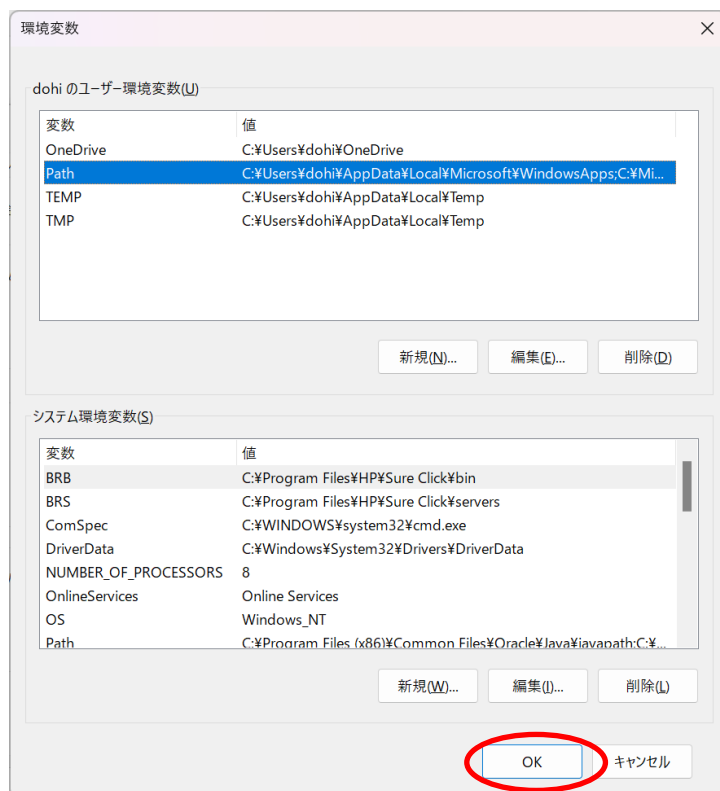
(5) 「新規」をクリックします。



- (6) 「C:¥mingw64¥bin」を半角でキーボードから入力し、「OK」をクリックします。スペルを間違えないように注意しましょう。



- (7) 「OK」をクリックします。

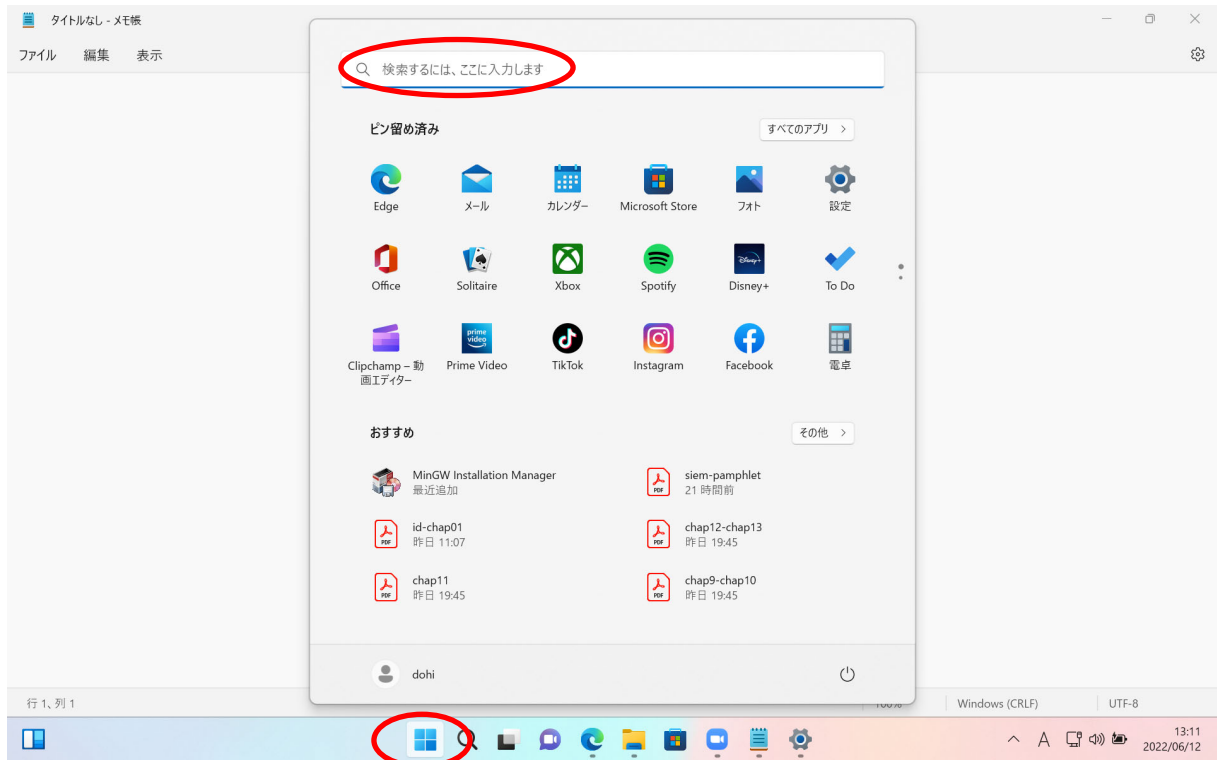


(8) 「OK」をクリックします。

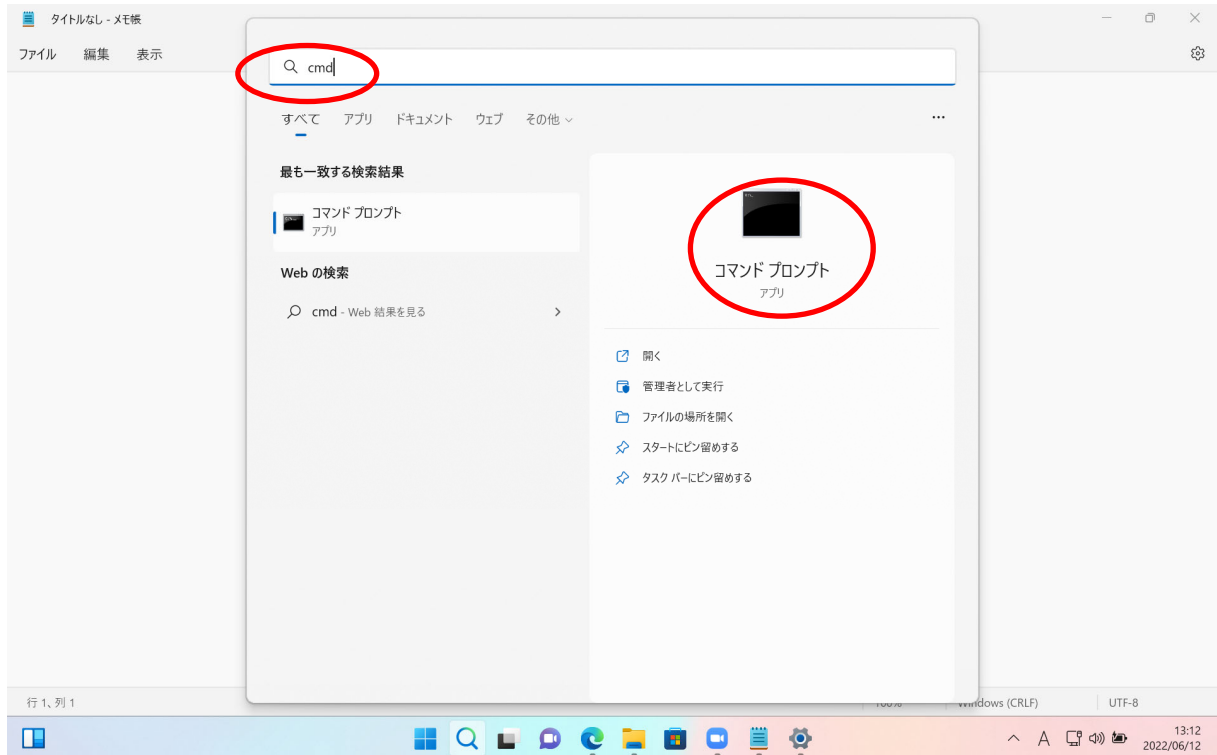


### 3. gcc の動作確認

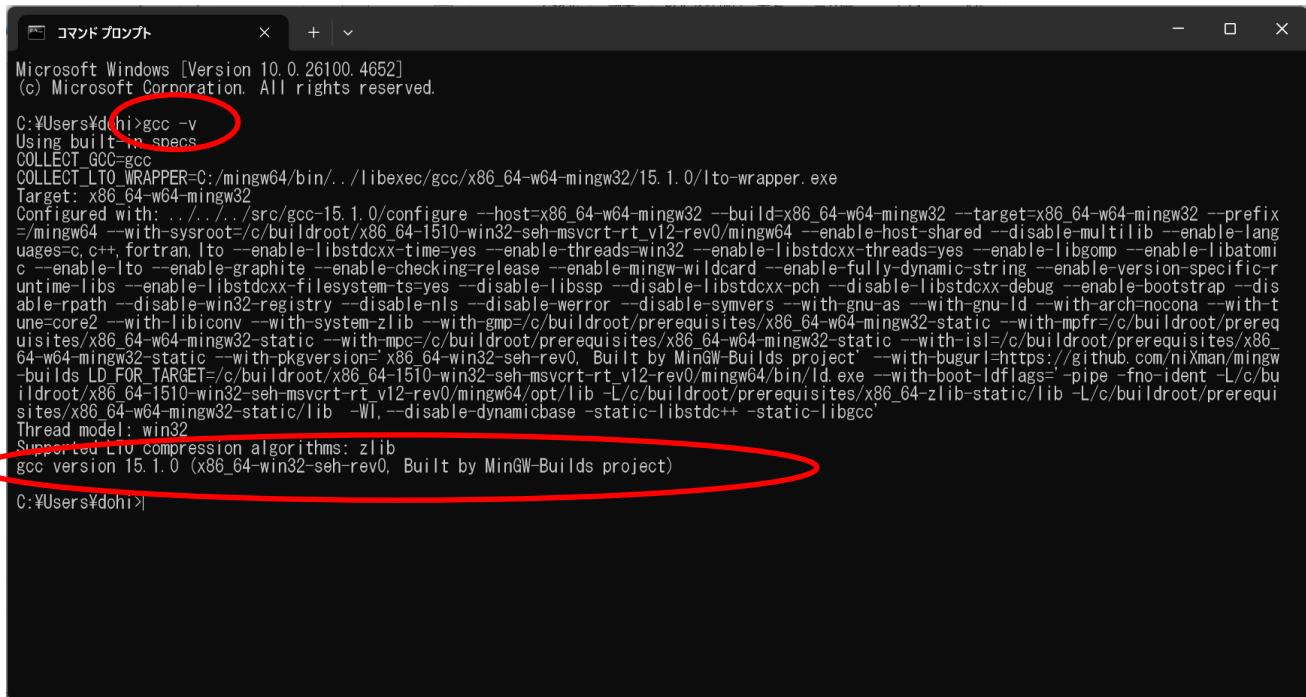
(1) 「スタート」をクリックし、「検索するには、ここをクリックします」をクリックします。



(2) 「cmd」と入力し、「コマンドプロンプト」をクリックします。右側にコマンドプロンプトが表示されていれば、Enter キーを押すこともできます。



(3) 「gcc -v」と入力した後 Enter キーを押し、バージョンの情報が表示されることを確認します。



```
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dohi>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=C:/mingw64/bin/./libexec/gcc/x86_64-w64-mingw32/15.1.0/lto-wrapper.exe
Target: x86_64-w64-mingw32
Configured with: ../../src/gcc-15.1.0/configure --host=x86_64-w64-mingw32 --build=x86_64-w64-mingw32 --target=x86_64-w64-mingw32 --prefix=/mingw64 --with-sysroot=C:/buildroot/x86_64-1510-win32-seh-msvcrt-rt_v12-rev0/mingw64 --enable-host-shared --disable-multilib --enable-lang-uages=c,c++,fortran,lto --enable-libstdcxx-time=yes --enable-threads=win32 --enable-libstdcxx-threads=yes --enable-libgomp --enable-libatomic --enable-lto --enable-graphite --enable-checking=release --enable-mingw-wildcard --enable-fully-dynamic-string --enable-version-specific-runtime-libs --enable-libstdcxx-filesystem-ts=yes --disable-libssp --disable-libstdcxx-pch --disable-libstdcxx-debug --enable-bootstrap --disable-rpath --disable-win32-registry --disable-nls --disable-werror --disable-symvers --with-gnu-as --with-gnu-ld --with-arch=nocona --with-tune=core2 --with-libiconv --with-system-zlib --with-gmp=C:/buildroot/prerequisites/x86_64-w64-mingw32-static --with-mpfr=C:/buildroot/prerequisites/x86_64-w64-mingw32-static --with-mpc=C:/buildroot/prerequisites/x86_64-w64-mingw32-static --with-isl=C:/buildroot/prerequisites/x86_64-w64-mingw32-static --with-pkgversion='x86_64-win32-seh-rev0, Built by MinGW-Builds project' --with-bugurl=https://github.com/nixman/mingw-builds LD FOR TARGET=C:/buildroot/x86_64-1510-win32-seh-msvcrt-rt_v12-rev0/mingw64/bin/ld.exe --with-boot-ldflags='-pipe -fno-ident -L/C:/buildroot/x86_64-1510-win32-seh-msvcrt-rt_v12-rev0/mingw64/opt/lib -L/C:/buildroot/prerequisites/x86_64-zlib-static/lib -L/C:/buildroot/prerequisites/x86_64-w64-mingw32-static/lib -Wl,--disable-dynamicbase -static-libstdc++ -static-libgcc'
Thread model: win32
Supported LTO compression algorithms: zlib
gcc version 15.1.0 (x86_64-win32-seh-rev0, Built by MinGW-Builds project)

C:\Users\dohi>
```