Lab 3: Makefiles
What is a Makefile?

- From https://www.gnu.org/software/make/::
  - GNU Make is a tool which controls the generation of executables and other non-source files of a program from the program's source files.
- In other words, it’s a way to specify and automate your build process.
How is a C++ program compiled?

- When you type:
  - `g++ library.cpp main.cpp -o main`

  … what exactly is the compiler doing?
The diagram illustrates the process of compiling and linking.

1. **Compilation**:
   - `main.cpp` compiles into `main.o` (object file).
   - `library.cpp` compiles into `library.o` (object file).

2. **Linking**:
   - `main.o` and `library.o` are linked to form `Main (executable)`.

The process involves:
- **Compilation**: Translating source code into object files.
- **Linking**: Combining object files into an executable program.
The “-c” flag

- Turns out we can ask the compiler to only do the “compile” step:
  - g++ main.cpp -c -o main.o
  - g++ library.cpp -c -o library.o
- “-c” stands for compile only.
- The above commands creates “main.o” and “library.o”.
- It does not link them together.
Linking them together

- Pass the object files to g++, as if they were cpp files:
  - `g++ library.cpp main.cpp -o main`
  - `g++ library.o main.o -o main`

Why would you do that?

- Simple: if you changed “library.cpp”, you do not have to recompile “main.cpp”!
Now we can explain Makefiles!

- Think of makefiles as recipes:
  - The ingredients for “main.o” is “main.cpp”.
  - The ingredients for “library.o” is “library.cpp”.
  - The ingredients for “main” is “main.o” and “library.o”.

  - To make “main.o”, do “g++ main.cpp –c –o main.o”
  - To make “library.o”, do “g++ library.cpp –c –o library.o”
  - To make “main”, do “g++ main.o library.o –o main”
In Makefile language:

main.o: main.cpp
        g++ main.cpp -c -o main.o

library.o: library.cpp
        g++ library.cpp -c -o library.o

main: main.o library.o
        g++ main.o library.o -o main
Now with make with more flags

main.o: main.cpp
  g++ -Wall -g main.cpp -c -o main.o

library.o: library.cpp
  g++ -Wall -g library.cpp -c -o library.o

main: main.o library.o
  g++ -Wall -g main.o library.o -o main
cxx = g++ -Wall -g

main.o: main.cpp
  $(cxx) main.cpp -c -o main.o

library.o: library.cpp
  $(cxx) library.cpp -c -o library.o

main: main.o library.o
  $(cxx) main.o library.o -o main