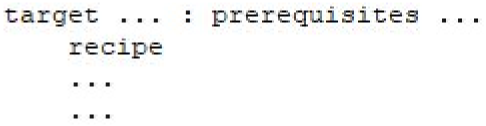
Reading from GNU make documentation

- Makefiles are a set of directions made up of **rules** to create a **target** (program)
- Makefiles are strictly operated on when **cleanup** (make clean) of files was called or when a .cpp or .h file was changed

Makefile "language" rules

- Comments are with #
- If a line of a code is getting too long, use a backslash, then keep typing the command on the new line

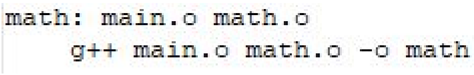
A Rule



```
target ... : prerequisites ...
        recipe
        ...
        ...
```

General syntax of a rule

target: This is going to be the program, and what it's going to be named
.o & .exe files
prerequisites: A file used to generate the target
.cpp, .h, .o
recipe: An action that `make` performs
g++ main.cpp matrix.cpp (console commands to compile)

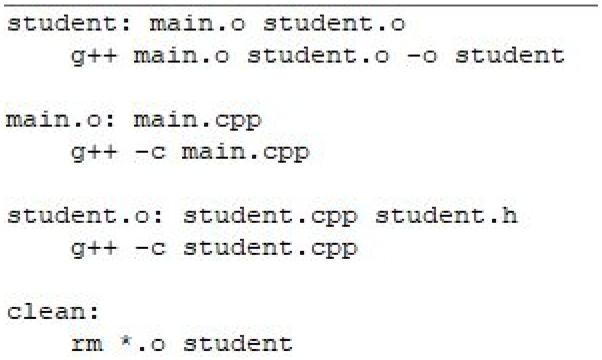


```
math: main.o math.o
        g++ main.o math.o -o math
```

Example of a rule

How do Makefiles know the main rule for compiling your WHOLE program?

- The first rule in your Makefile will be your program rule, and the other rules (main.o in graphic below) are prerequisites



```
student: main.o student.o
    g++ main.o student.o -o student

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

student.o: student.cpp student.h
    g++ -c student.cpp

clean:
    rm *.o student
```

Very simple makefile

- `make`
    - When we type `make` into our command line, it'll execute the **recipes** to make our program
- `make clean`
    - When we type `make clean` it will execute `clean:` which has no prerequisite files and only actions
    - **Phony target:** Targets with no prerequisite files