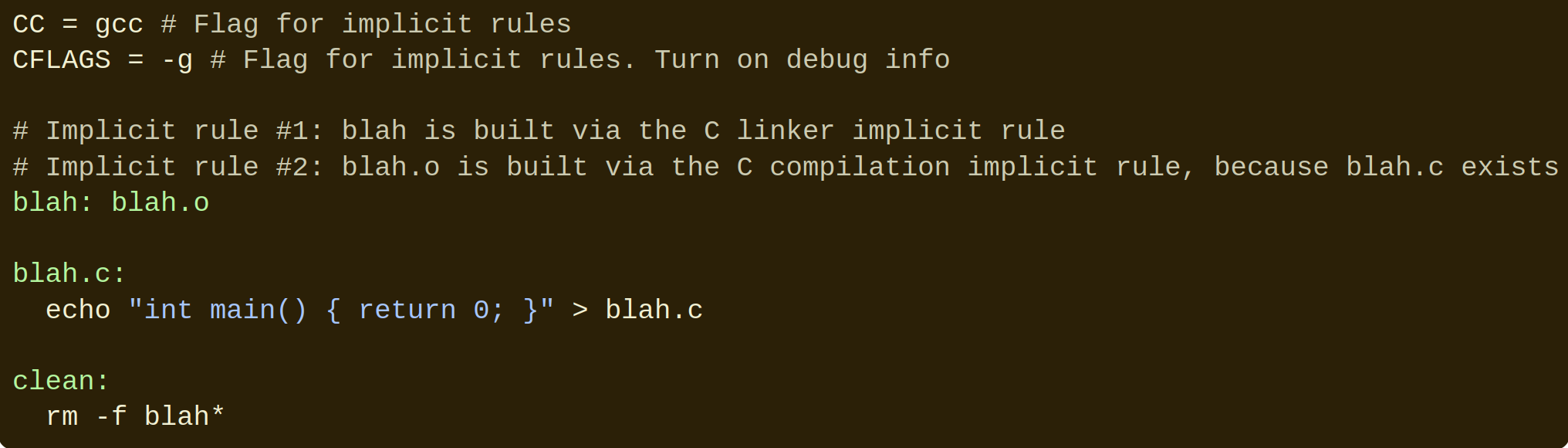
* .cpp complied to object file .o (-c) then linked to make an executable
* -o : naming option of output file.
* Automatic variables:
  + $@ : target
  + $^ : prerequisites
  + $< : first prerequisite
  + $? : prerequisites newer than the target
* Target : dependencies

command

command

* When we run make, it will check the timestamp of target and the dependencies. If timestamp of a dependency is more than of target, it will recompile that dependency and link it to the target.
* Add @ before a command for it to not be visible in terminal (silence it).
* *make -s* to silence all the commands.
* clean wont run unless called explicitly with ***make clean.***
* $() or ${} to reference variables
* = When values might change later and should be re-evaluated at use time.
* := When values should be fixed at assignment to avoid unnecessary re-evaluation.
* ?= can be overridden from CLI.
* all target defined first (if make is called w/o specifying target, it will run all)
* Multiple targets can be defined at once.
* Eg : $(wildcard \*.o)
* Implicit Rules:
* Compiling a C++ program: n.o is made automatically from n.cc or n.cpp with a command of the form $(CXX) -c $(CPPFLAGS) $(CXXFLAGS) $^ -o $@
* Linking a single object file: n is made automatically from n.o by running the command $(CC) $(LDFLAGS) $^ $(LOADLIBES) $(LDLIBS) -o $@



* targets...: target-pattern: prereq-patterns ...

commands

* Add -k when running make to continue running even in the face of errors. Helpful if you want to see all the errors of Make at once.
* Add a - before a command to suppress the error
* Add -i to make to have this happen for every command.
* String Substitution: $(patsubst pattern,replacement,text)
* .PHONY: clean
* % :

# Define a pattern rule that compiles every .c file into a .o file

%.o : %.c

$(CC) -c $(CFLAGS) $(CPPFLAGS) $< -o $@

—-------------------------------------------------------------------------------------------

objects = foo.o bar.o all.o

all: $(objects)

$(CC) $^ -o all

$(objects): %.o: %.c

$(CC) -c $^ -o $@

—-------------------------------------------------------------------------------------------

foo := a.o b.o l.a c.o

one := $(patsubst %.o,%.c,$(foo))

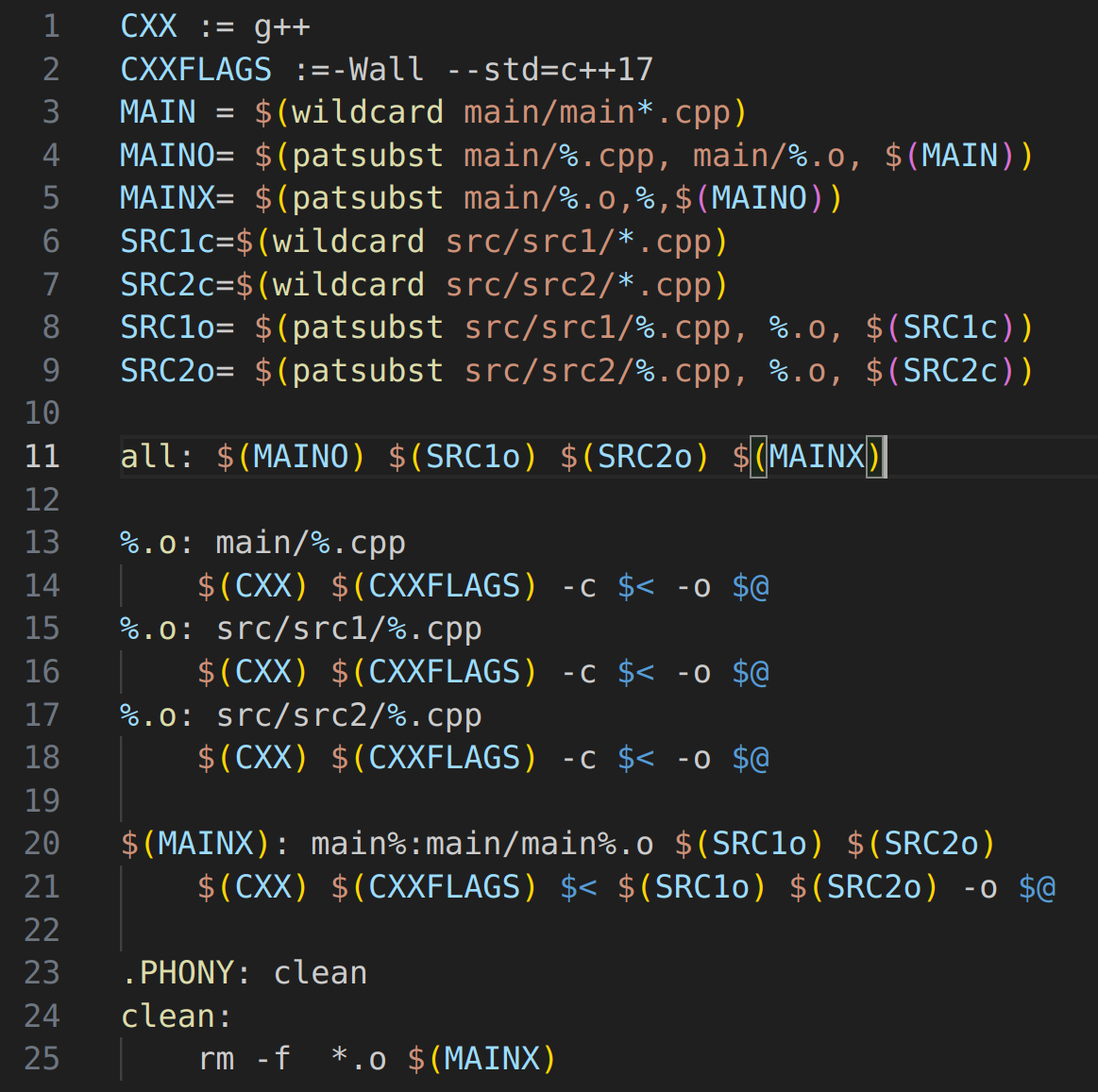
# This is a shorthand for the above

two := $(foo:%.o=%.c)

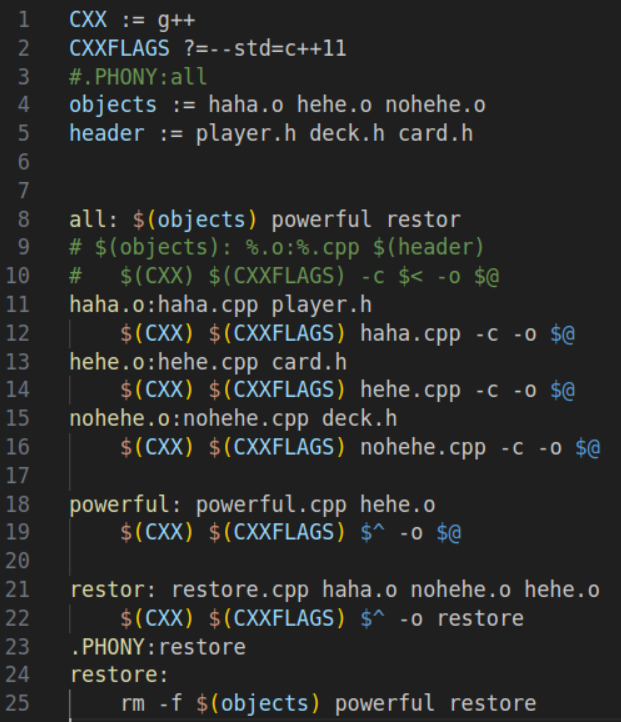
# This is the suffix-only shorthand, and is also equivalent to the above.

three := $(foo:.o=.c)

—-------------------------------------------------------------------------------------------



* CC: Program for compiling C programs; default cc
* CXX: Program for compiling C++ programs; default g++
* CFLAGS: Extra flags to give to the C compiler
* CXXFLAGS: Extra flags to give to the C++ compiler
* CPPFLAGS: Extra flags to give to the C preprocessor
* LDFLAGS: Extra flags to give to compilers when they are supposed to invoke the linker



grep "Name" analysis.txt > michaelsleep.txt

grep "^[A-Z][a-z]\*,[A-Z][A-Z]@[0-9][0-9][0-9],[0-9]\*,[0-9]\*,[0-9]\*days$" analysis.txt | grep -v "000" >> michaelsleep.txt

cut michaelsleep.txt -d ',' -f 1-2 |sort -d >> michaelhero.txt