Lab 3 report

Ethan Coe-Renner

September 9, 2021

Contents

1.
   ```
   all:
   gcc main.c output.c factorial.c -o factorial_program
   ```

2.
   ```
   all: factorial_program

   factorial_program: main.o factorial.o output.o
   gcc main.o factorial.o output.o -o factorial_program

   main.o: main.c
   gcc -c main.c

   factorial.o: factorial.c
   gcc -c factorial.c

   output.o: output.c
   gcc -c output.c

   clean:
   rm -rf *.o factorial_program
   ```

3. when `make -f Makefile-2` is entered, make runs the all recipe in the Makefile-2. this calls the `factorial_program` recipe, which, in turn, runs the main.o, factorial.o, and output.o recipes. These recipes construct their respective files by compiling there associated source code. finally, the factorial program recipe runs and links all the object files together.

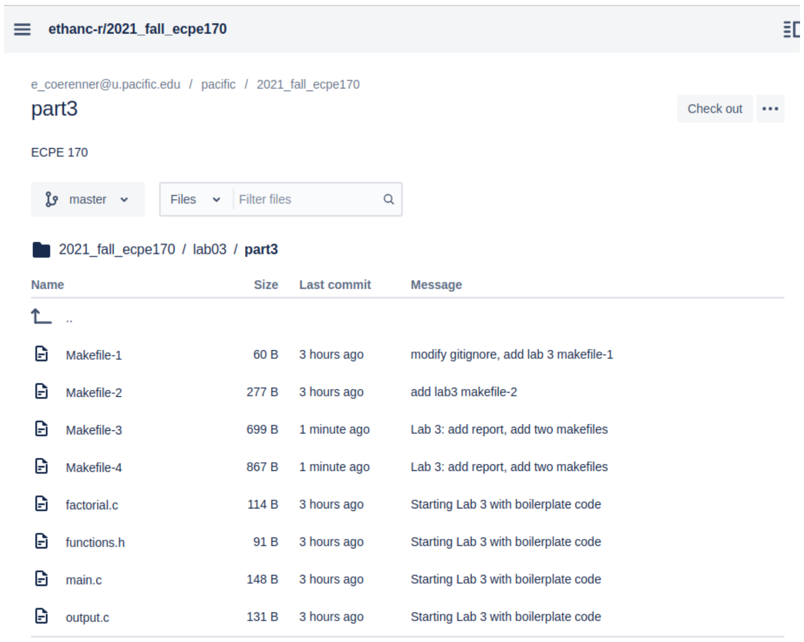
4.
```
# The variable CC specifies which compiler will be used.
# (because different unix systems may use different compilers)
CC=gcc

# The variable CFLAGS specifies compiler options
#   -c :    Only compile (don't link)
#   -Wall:  Enable all warnings about lazy / dangerous C programming
CFLAGS=-c -Wall

# The final program to build
EXECUTABLE=factorial_program

# --------------------------------------------

all: $(EXECUTABLE)

$(EXECUTABLE): main.o factorial.o output.o
$(CC) main.o factorial.o output.o -o $(EXECUTABLE)

main.o: main.c
$(CC) $(CFLAGS) main.c

factorial.o: factorial.c
$(CC) $(CFLAGS) factorial.c

output.o: output.c
$(CC) $(CFLAGS) output.c

clean:
rm -rf *.o $(EXECUTABLE)
```

5.
```
# The variable CC specifies which compiler will be used.
# (because different unix systems may use different compilers)
CC=gcc

# The variable CFLAGS specifies compiler options
#   -c :    Only compile (don't link)
#   -Wall:  Enable all warnings about lazy / dangerous C programming
#  You can add additional options on this same line..
#  WARNING: NEVER REMOVE THE -c FLAG, it is essential to proper operation
```

```
CFLAGS=-c -Wall

# All of the .h header files to use as dependencies
HEADERS=functions.h

# All of the object files to produce as intermediary work
OBJECTS=main.o factorial.o output.o

# The final program to build
EXECUTABLE=factorial_program

# --------------------------------------------

all: $(EXECUTABLE)

$(EXECUTABLE): $(OBJECTS)
$(CC) $(OBJECTS) -o $(EXECUTABLE)

%.o: %.c $(HEADERS)
$(CC) $(CFLAGS) -o $@ $<

clean:
rm -rf *.o $(EXECUTABLE)
```

6. When `make -f Makefile-4` is run, the all recipe is called. this calls the `factorial_program` recipe which calls all the object recipes. This triggers the %.o recipe which compiles all the src to object files. The `factorial_program` recipe then links all the object files into an executable called `factorial_program`.

7. I would need to change the variables HEADERS, OBJECTS, EXECUTABLE, and perhaps CFLAGS.



ethanc-r/2021_fall_ecpe170

e_coerenner@u.pacific.edu / pacific / 2021_fall_ecpe170

part3

Check out

ECPE 170

master | Files | Filter files

2021_fall_ecpe170 / lab03 / **part3**

Name	Size	Last commit	Message
..			
Makefile-1	60 B	3 hours ago	modify gitignore, add lab 3 makefile-1
Makefile-2	277 B	3 hours ago	add lab3 makefile-2
Makefile-3	699 B	1 minute ago	Lab 3: add report, add two makefiles
Makefile-4	867 B	1 minute ago	Lab 3: add report, add two makefiles
factorial.c	114 B	3 hours ago	Starting Lab 3 with boilerplate code
functions.h	91 B	3 hours ago	Starting Lab 3 with boilerplate code
main.c	148 B	3 hours ago	Starting Lab 3 with boilerplate code
output.c	131 B	3 hours ago	Starting Lab 3 with boilerplate code

8.