# Makefiles & Project 1 Q&A

15-441 Recitation 2

441 Staff

## Outline

- gcc
- make and Makefile
- Useful commands
- Project 1 Q&A

# Simple gcc

If we have files:

- prog.c: The main program file
- lib.c: Library  .c file
- lib.h: Library  header file

```
% gcc -c prog.c -o prog.o
% gcc -c lib.c -o lib.o
% gcc lib.o prog.o -o binary
```

# gcc flags

- Useful flags
  1. -g: debugging hook
  2. -Wall: all warning
  3. -Werror: treat warning as errors
  4. -O2, -O3: optimization
  5. -DDEBUG: macro for DEBUG (#define DEBUG)

## Examples

```
% gcc -g -Wall -Werror -c prog.c -o prog.o
% gcc -g -Wall -Werror -c lib.c -o lib.o
% gcc -g -Wall -Werror lib.o prog.o -o binary
```

But Don’t Repeat Yourself!

# Makefile

```
% gcc -g -Wall -Werror -c prog.c -o prog.o
% gcc -g -Wall -Werror -c lib.c -o lib.o
% gcc -g -Wall -Werror lib.o prog.o -o binary
```

```
CC = gcc
CFLAGS = -g -Wall -Werror
OUTPUT = binary
```

# Makefile

```
target: dependency1 dependency2 ...
	unix command (start line with TAB)
	unix command
	...
```

```
% gcc lib.o prog.o -o binary
```

```
binary: lib.o prog.o
	gcc lib.o prog.o -o binary
```

```
binary: lib.o prog.o
	gcc -g -Wall lib.o prog.o -o binary

lib.o: lib.c
	gcc -g -Wall -c lib.c -o lib.o

prog.o: prog.c
	gcc -g -Wall -c prog.c -o prog.o

clean:
	rm *.o binary
```

```
binary: lib.o prog.o
	gcc -g -Wall lib.o prog.o -o binary

lib.o: lib.c
	gcc -g -Wall -c lib.c -o lib.o

prog.o: prog.c
	gcc -g -Wall -c prog.c -o prog.o

clean:
	rm *.o binary
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary

$(OUTPUT): lib.o prog.o
	$(CC) $(CFLAGS) lib.o prog.o -o binary

lib.o: lib.c
	$(CC) $(CFLAGS) -c lib.c -o lib.o

prog.o: prog.c
	$(CC) $(CFLAGS) -c prog.c -o prog.o

clean:
	rm *.o $(OUTPUT)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary

$(OUTPUT): lib.o prog.o
	$(CC) $(CFLAGS) lib.o prog.o -o binary

lib.o: lib.c
	$(CC) $(CFLAGS) -c lib.c -o lib.o

prog.o: prog.c
	$(CC) $(CFLAGS) -c prog.c -o prog.o

clean:
	rm *.o $(OUTPUT)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary
OBJFILES = lib.o prog.o

$(OUTPUT): $(OBJFILES)
	$(CC) $(CFLAGS) $(OBJFILES) -o binary

lib.o: lib.c
	$(CC) $(CFLAGS) -c lib.c -o lib.o

prog.o: prog.c
	$(CC) $(CFLAGS) -c prog.c -o prog.o

clean:
	rm *.o $(OUTPUT)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary
OBJFILES = lib.o prog.o

$(OUTPUT): $(OBJFILES)
	$(CC) $(CFLAGS) $(OBJFILES) -o binary

lib.o: lib.c
	$(CC) $(CFLAGS) -c lib.c -o lib.o

prog.o: prog.c
	$(CC) $(CFLAGS) -c prog.c -o prog.o

clean:
	rm *.o $(OUTPUT)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary
OBJFILES = lib.o prog.o

$(OUTPUT): $(OBJFILES)
	$(CC) $(CFLAGS) $(OBJFILES) -o binary

%.o: %.c
	# $<: dependency (%.c)
	# $@: target (%.o)
	$(CC) $(CFLAGS) -c $< -o $@

clean:
	rm *.o $(OUTPUT)
```

## Simple Test Script

```
% ./server 6667 &
% cat testfile.01 | ./testscript.py
% cat testfile.02 | ./testscript.py
% killall -9 server
```

## Simple Test Script

```
#/bin/sh

echo “Starting server on port 6667.”
./server 6667 &
SERVERPID = $!

echo “Running test files.”
cat testfile.01 | ./testscript.py
cat testfile.02 | ./testscript.py

echo “Killing server process.”
kill $(SERVERPID)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary
OBJFILES = lib.o prog.o

all: $(OUTPUT)

$(OUTPUT): $(OBJFILES)
	$(CC) $(CFLAGS) $(OBJFILES) -o binary

%.o: %.c
	# $<: dependencies (%.c)
	# $@: target (%.o)
	$(CC) $(CFLAGS) -c $< -o $@




clean:
	rm *.o $(OUTPUT)
```

```
CC = gcc
CFLAGS = -g -Wall
OUTPUT = binary
OBJFILES = lib.o prog.o

all: $(OUTPUT)  test

$(OUTPUT): $(OBJFILES)
	$(CC) $(CFLAGS) $(OBJFILES) -o binary

%.o: %.c
	# $<: dependencies (%.c)
	# $@: target (%.o)
	$(CC) $(CFLAGS) -c $< -o $@

test: $(OUTPUT)
	sh ./testscript.sh

clean:
	rm *.o $(OUTPUT)
```

# Use Makefile

```
% make
% make test
% make clean
```

Google

- “makefile example”
- “makefile template”
- “make tutorial”

# Useful Unix Commands

- find “func_name” in files

```
% grep -r func_name .
```

- replace “bad_func_name” to “good_func_name”

```
% sed -e “s/bad_func_name/good_func_name/g”\
    prog.c > prog.c.new
```

## Useful Unix Commands

- find a file named “prog.c”

```
% find -name prog.c
```

- download files from Internet

```
% wget http://address/to/file.tar.gz
```

- untar and unzip the file

```
% tar xzvf file.tar.gz
```

## Project 1

- Checkpoint 2
  - Echo server
  - Handle multiple clients
  - Handle TCP framing
- Q & A