

In this homework, you will need to work on the eniac-l.seas.upenn.edu server. If you need help logging in to this machine see the following webpage: <http://www.seas.upenn.edu/cets/answers/remote.html> (note: you need to swap eniac-l.seas.upenn.edu for eniac.seas.upenn.edu).

You should do all work for this assignment using emacs. In some questions below, you will be asked to enter some command with some set of arguments. Depending on the command, there may or may not be output generated. For each command, you should use your mouse to highlight the command (including the prompt) and the output (if there is any) and use emacs to copy it to a separate text file. Obviously, when you make mistakes, you should not copy the text. You should also copy the question to the text file as well.

When you are finished with the assignment, attach the text document to an email and send to *mcorliss@cis.upenn.edu*. The subject of the email should be "cse399 - hw5" (without the quotes). The homework is due by the beginning of Monday's class. Total points: 50.

1. Preliminary.

- (a) Change your location to the 'cse399' directory you created in homework 1 (or create it again if you deleted it). Copy the directory /home1/m/mcorliss/teaching/cse399/hw5 and all its subcontents to your current location. Change your location to this new directory (hw5).

Answer:

```
bash$ cd cse399
bash$ cp -r /home1/m/mcorliss/teaching/cse399/hw5 .
bash$ cd hw5
```

2. [12 Points] Environment.

- (a) On the command-line, define an environment variable SEAS and set it to "eniac-l.seas.upenn.edu".

Answer:

```
bash$ export SEAS=eniac-l.seas.upenn.edu
```

- (b) On the command-line, define an alias 'e' for emacs. Afterwards, when you type 'e &', bash should open emacs.

Answer:

```
bash$ alias e=emacs
```

- (c) Make the alias from the previous exercise permanent. That is, everytime you login, the alias 'e' should work. Describe your solution.

Answer:

Open '.bashrc' in emacs and add the line: alias e=emacs

- (d) Add an environment variable SEAS_S to your .profile. Set SEAS_S to "eniac.seas.upenn.edu". Reload the changed .profile without logging out and logging back in.

Answer:

Open '.profile' in emacs and add the line: export SEAS_S=eniac.seas.upenn.edu. To reload .profile without logging out, can use the command: source ~/.profile.

(e) Print out the environment.

Answer:
bash\$ env

(f) Print the environment, but only select the OSTYPE variable.

Answer:
bash\$ env|grep OSTYPE

3. [20 Points] Compiling and makefiles.

(a) Compile the program helloworld.c using *gcc* and the debugging flag. Separate compilation into two steps: 1) compiling the source file to an object file and 2) linking the object file to create an executable.

Answer:
bash\$ gcc -g -c helloworld.c
bash\$ gcc -g -o helloworld helloworld.o

(b) Build a makefile for compiling helloworld. Your Makefile should use the following variables described in the lectures: CC, LD, CFLAGS, LDFLAGS, OBJS, and PROG. Your makefile should also provide a rule to clean up any files generated by the makefile.

Answer:

```
CC = gcc
LD = gcc
CFLAGS = -g
LDFLAGS = -g
OBJS = helloworld.o
PROG = helloworld

all: $(PROG)

$(PROG): $(OBJS)
    $(LD) $(LDFLAGS) -o $(PROG) $(OBJS)

%.o: %.c
    $(CC) $(CFLAGS) -c $*.c

clean:
    /bin/rm -f $(PROG) $(OBJS)
```

(c) Change your location to the directory 'prog'. In this directory there are three C files: main.c, foo.c, goo.c. main.c depends on main.c, foo.h, goo.h. foo.c depends on foo.h. goo.c depends on goo.h. Compile these .c files and build an executable called 'prog'.

Answer:
bash\$ gcc -c main.c
bash\$ gcc -c foo.c
bash\$ gcc -c goo.c
bash\$ gcc -o prog *.o

(d) Build a makefile, like the one in the exercise 3(b) (using the same variables and a clean rule), for this program.

Answer:

```
CC = gcc
LD = gcc
CFLAGS =
LDFLAGS =
OBJS = foo.o goo.o main.o
PROG = prog

all: $(PROG)

$(PROG): $(OBJS)
        $(LD) $(LDFLAGS) -o $(PROG) $(OBJS)

main.o: main.c foo.h goo.h
        $(CC) $(CFLAGS) -c main.c

%.o: %.c %.h
        $(CC) $(CFLAGS) -c $*.c

clean:
        /bin/rm -f $(PROG) $(OBJS)
```

- (e) Change your location to the directory ‘../bad-make’. In this directory there is a makefile for compiling a program. It contains five mistakes, which you should find and describe.

Answer:

- i. On line 5, the LD variable declaration spans two lines.
 - ii. On line 6, comments should start with “#” not “;”
 - iii. On line 15, the top-level all rule should be dependent on \$(PROG). Otherwise, by default make will compile nothing.
 - iv. On line 21, the command is preceded by white spaces not a tab.
 - v. On line 23, the clean rule should not be dependent on any files.
4. [18 Points] Installing new software.

- (a) In this exercise, you will install a new version of bash in the directory ‘~/cse399/hw5/bash’. The source files are stored within ‘bash-3.0.tar.gz’. Extract these files, and then using the file ‘bash-3.0/INSTALL’ as a reference, figure out how to install the program. You also might find the following webpage helpful: <http://www.ee.surrey.ac.uk/Teaching/Unix/unix7.html>.

Answer:

```
bash$ ./configure --prefix=/home1/m/mcorliss/bash
bash$ make
bash$ make install
```

5. About this assignment.

- (a) Approximately, how long did it take you to complete this homework?
- (b) Would you classify this assignment as easy, straight-forward, or difficult?