```
 1: // $Id: hello.h,v 1.3 2014-06-10 20:00:20-07 - - $
 2:
 3: #ifndef __HELLO_H__
 4: #define __HELLO_H__
 5:
 6: #include <iostream>
 7: #include <string>
 8:
 9: class hello {
10:    private:
11:       std::string message {"Hello, World!"};
12:    public:
13:       hello();                            // default constructor
14:       hello (const hello&);               // copy constructor
15:       hello& operator= (const hello&); // copy operator=
16:       hello (hello&&);                    // move constructor
17:       hello& operator= (hello&&);         // move operator=
18:       ~hello();                           // destructor
19:       hello (const string&);              // alternate constructor
20:       void say (ostream&);                // member function
21: };
22:
23: #endif
24:
```

```
 1: // $Id: hello.cpp,v 1.6 2014-06-10 20:10:40-07 - - $
 2:
 3: #include <iostream>
 4: #include <string>
 5: using namespace std;
 6:
 7: #include "hello.h"
 8:
 9: hello::hello() {
10:    cout << this << "->hello::hello()" << endl;
11: }
12:
13: hello::hello (const hello& that): message (that.message) {
14:    cout << this << "->hello::hello (const hello&)" << endl;
15: }
16:
17: hello& hello::operator= (const hello&) {
18:    cout << this << "->hello& hello::operator= (const hello&)" << endl;
19:    return *this;
20: }
21:
22: hello::hello (hello&&) {
23:    cout << this << "->hello::hello (hello&&)" << endl;
24: }
25:
26: hello& hello::operator= (hello&&) {
27:    cout << this << "->hello& hello::operator= (hello&&)" << endl;
28:    return *this;
29: }
30:
31: hello::~hello() {
32:    cout << this << "->hello::~hello()" << endl;
33: }
34:
35: hello::hello (const string& message): message(message) {
36:    cout << this << "->hello::hello (" << message << ")" << endl;
37: }
38:
39: void hello::say (ostream& out) {
40:    out << message << endl;
41: }
42:
```

```
 1: // $Id: main.cpp,v 1.4 2014-06-10 20:08:02-07 - - $
 2:
 3: #include <algorithm>
 4: #include <cstdlib>
 5: #include <iostream>
 6: #include <memory>
 7: #include <string>
 8: #include <vector>
 9: using namespace std;
10:
11: #include <libgen.h>
12:
13: #include "hello.h"
14:
15: int main (int argc, char** argv) {
16:    string execname {basename (argv[0])};
17:    vector<string> args (&argv[1], &argv[argc]);
18:    cout << execname << endl;
19:    auto hello_ptr = make_shared<hello>();
20:    hello_ptr->say (cout);
21:    hello goodbye {"Goodbye, world!"};
22:    goodbye.say (cout);
23:    hello second {*hello_ptr};
24:    second.say (cout);
25:    for (const auto& arg: args) cout << "for: " << arg << endl;
26:    for_each (&argv[0], &argv[argc],
27:              [=] (const char* arg) {
28:                 cout << "for_each: " << arg << endl;
29:              });
30:    return EXIT_SUCCESS;
31: }
32:
```

```
# $Id: Makefile,v 1.7 2014-06-19 20:25:02-07 - - $

MKFILE      = Makefile
DEPFILE     = ${MKFILE}.dep
NOINCL      = ci clean spotless
NEEDINCL    = ${filter ${NOINCL}, ${MAKECMDGOALS}}
GMAKE       = ${MAKE} --no-print-directory
COMPILECPP  = g++ -g -O0 -Wall -Wextra -rdynamic -std=gnu++0x
MAKEDEPCPP  = g++ -MM

CPPHEADER   = hello.h
CPPSOURCE   = hello.cpp main.cpp
ALLCPPSRC   = ${CPPHEADER} ${CPPSOURCE}
OBJECTS     = ${CPPSOURCE:.cpp=.o}
EXECBIN     = say_hello
OTHERS      = ${MKFILE} README.html
ALLSOURCES  = ${ALLCPPSRC} ${OTHERS}
LISTING     = Listing.ps

all : ${EXECBIN}

${EXECBIN} : ${OBJECTS}
        ${COMPILECPP} -o $@ ${OBJECTS}

%.o : %.cpp
        ${COMPILECPP} -c $<

ci : ${ALLSOURCES}
        - checksource ${ALLSOURCES}
        cid + ${ALLSOURCES}

lis : ${ALLSOURCES} test
        mkpspdf ${LISTING} ${ALLSOURCES} ${DEPFILE} test.out

clean :
        - rm ${OBJECTS} ${DEPFILE} core test.out

spotless : clean
        - rm ${EXECBIN} ${LISTING} ${LISTING:.ps=.pdf}

test : ${EXECBIN}
        ./${EXECBIN} foo bar baz qux >test.out 2>&1


dep : ${ALLCPPSRC}
        @ echo "# ${DEPFILE} created `LC_TIME=C date`" >${DEPFILE}
        ${MAKEDEPCPP} ${CPPSOURCE} >>${DEPFILE}

${DEPFILE} : ${MAKEFILE}
        @ touch ${DEPFILE}
        ${GMAKE} dep

again :
        ${GMAKE} spotless dep ci all lis

ifeq (${NEEDINCL}, )
include ${DEPFILE}
endif
```

```
 1: <HEAD>
 2: <STYLE>
 3: p { max-width: 35em; }
 4: </STYLE>
 5: </HEAD>
 6: <BODY>
 7: <TT>
 8: <P>
 9: Make sure you know how to submit files. 
10: Submit these files to asg0. 
11: Verify that the submit actually worked by looking
12: in the submit directory for your username. 
13: The find(1) command is very useful for things like this. 
14: <P>
15: This asg0-intro-unix will not be graded and there
16: is no credit for it. 
17: However, it is important you understand the Unix command
18: line and submit procedure before you actually use it in the
19: first assignment.
20: <P>
21: The submit command copies your files into a directory
22: <BR>
23: <TT>/afs/cats.ucsc.edu/class/cmps109-wm.u14/asg0/$USER</TT>
24: <BR>
25: and prefixes each file with a sequence number. 
26: You can see the names of the files,
27: but not their contents. 
28: <P>
29: Prior experience with Unix is assumed. 
30: Attend a lab section to ask questions if you don't
31: understand how submit works.
32: The TA can explain such things. 
33: <P>
34: Also read the submit checklist: 
35: <BR>
36: <A HREF=
37: http://www2.ucsc.edu/courses/cmps109-wm/:/Syllabus/submit-checklist/>
38: http://www2.ucsc.edu/courses/cmps109-wm/:/Syllabus/submit-checklist/
39: </A>
40: <P>
41: This directory shows a main program and a hello class and
42: illustrates various features of C++11 that will be covered
43: later in the course. 
44: <P>
45: Whenever you need to look up information about C++,
46: use Google,
47: or look at
48: The C++ Resources Network: 
49: <BR>
50: <A HREF=http://www.cplusplus.com/>
51: http://www.cplusplus.com/</A>
52: <P>
53: $Id: README.html,v 1.14 2014-06-17 15:45:48-07 - - $
54: </BODY>
```

```
1: # Makefile.dep created Thu Jun 19 20:25:02 PDT 2014
2: hello.o: hello.cpp hello.h
3: main.o: main.cpp hello.h
```

```
 1: say_hello
 2: 0x1e58148->hello::hello()
 3: Hello, World!
 4: 0x7fff8369c090->hello::hello (Goodbye, world!)
 5: Goodbye, world!
 6: 0x7fff8369c080->hello::hello (const hello&)
 7: Hello, World!
 8: for: foo
 9: for: bar
10: for: baz
11: for: qux
12: for_each: ./say_hello
13: for_each: foo
14: for_each: bar
15: for_each: baz
16: for_each: qux
17: 0x7fff8369c080->hello::~hello()
18: 0x7fff8369c090->hello::~hello()
19: 0x1e58148->hello::~hello()
```