```
// utils.h: header file ......................................

#ifndef __UTILS_H__
#define __UTILS_H__

void support();

#endif

// utils.cpp: utility functions ..............................

#include <iostream>
#include "utils.h"
using namespace std;

void support()
{
  cout << "support\n";
}

// program.cpp: simple driver program ........................

#include <iostream>
#include "utils.h"
using namespace std;

int main(int argc, char *argv[])
{
  cout << "program\n";
  support();
}

---------------------------------------------------------------

# makefile: version 1
#
# if target is older than dependencies, execute action
#
# target: dependencies
# [tab] action

program: utils.h program.cpp utils.cpp
    g++ -g -Wall -o program program.cpp utils.cpp

clean:
    rm -f program
```

```
# makefile: version 2
#
# macros allow for easy global substitution

CC = g++
CFLAGS = -g -Wall

program: program.o utils.o
    $(CC) $(CFLAGS) -o program program.o utils.o

program.o: utils.h program.cpp
    $(CC) $(CFLAGS) -c program.cpp

utils.o: utils.h utils.cpp
    $(CC) $(CFLAGS) -c utils.cpp

clean:
    rm -f program *.o

---------------------------------------------------------------

# makefile: version 3
#
# macro variable substitution: from=to
# suffix rule: .from.to
# automatic variable $@: target
# automatic variable $^: all dependencies
# automatic variable $<: first dependency

CC = g++
CFLAGS = -g -Wall

HDR = utils.h
BIN = program
SRC = program.cpp utils.cpp
OBJ = $(SRC:%.cpp=%.o)

all: $(BIN)

$(BIN): $(OBJ)
    $(CC) $(CFLAGS) -o $@ $^

%.o: %.cpp utils.h
    $(CC) $(CFLAGS) -c $<

clean:
    rm -f $(BIN) $(OBJ)
```