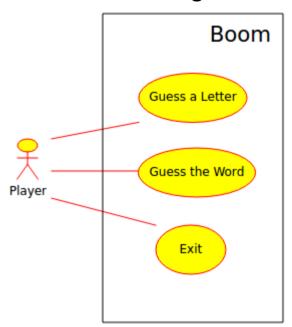
Homework Retrospective Full Credit UML

Use Case Diagram



git log

ricegf@pluto:~/dev/cpp/201701/P3/xb\$ git log
commit 8f5faaa3cd97c01e1ffc6ccacea98b500a0eb648

Author: ricegf <george.rice@uta.edu>
Date: Mon Jan 30 21:58:01 2017 -0600

selects random word

commit 8a42f7de528b098778c06eb25a9be512d7bc2061

Author: ricegf <george.rice@uta.edu>
Date: Mon Jan 30 21:33:11 2017 -0600

only fizz on a miss

commit ab78e428dd66106b852bb4e31b544bc83e2736c6

Author: ricegf <george.rice@uta.edu>
Date: Mon Jan 30 21:17:27 2017 -0600

finalized FC version

commit de40be42f421fc5694e5da49ff49b2f04a3ec908

Author: ricegf <george.rice@uta.edu>
Date: Mon Jan 30 21:05:08 2017 -0600

First draft - basic game ricegf@pluto:~/dev/cpp/201701/P3/xb\$

Homework Retrospective Full Credit – Fuse Class

```
#include <string>
using namespace std;
class Fuse {
                                                                          Puzzle
  public:

    solution : string

     Fuse(int time) : _time{time} { };
                                                           quesses[255] : bool
                                                           + Puzzle(solution : string)
     bool burn(); // true if any fuse remain
                                                           + guess(c : char) : bool
     string to_string();
                                                           + solve(proposed solution : string) : string
  private:
                                                           + to string(): string
                                                           + get_solution() : string
     int _time;
};
```

```
#include "fuse.h"
   bool Fuse::burn() {
     _time = (_time > 0) ? --_time : _time;
nse.cpp
     return _time > 0;
   string Fuse::to string() {
     string result = " ";
     for(int i = 0; i < _time; ++i) result += '_';
     result += "*\n /\n,+,\n| |\n|_|\n";
     return result;
```

Main + main(): int Fuse time : int + Fuse(time : int) + burn(): bool + to string(): string

ouzzle.h

Homework Retrospective Full Credit – Puzzle Class .h

Main

```
+ main(): int
                                                                      Puzzle
                                                                                                   Fuse

    solution : string

                                                                                               time : int
                                                        quesses[255] : bool
                                                                                             + Fuse(time : int)
#include <string>
                                                        + Puzzle(solution : string)
                                                                                             + burn(): bool
                                                        + guess(c : char) : bool
                                                                                             + to string(): string
using namespace std;
                                                        + solve(proposed solution : string) : string
                                                        + to string(): string
class Puzzle {
                                                        + get solution(): string
  public:
     Puzzle(string solution) : _solution{solution} {_guesses[' '] = true;}
     bool quess(char c);
                                                   // true if valid guess
     bool solve(string proposed_solution); // true if correctly guessed
     string to string();
     string get solution();
  private:
     string _solution;
     bool _guesses[255] = {false};
};
```

Homework Retrospective Full Credit – Puzzle Class .cpp

```
Main
#include "puzzle.h"
                                                                                    + main(): int
bool Puzzle::quess(char c) {
  if (c<'a' || c>'z' || _guesses[c]) {
                                                                    Puzzle
                                                                                                 Fuse
    return false;

    solution : string

                                                                                             time : int
  } else {
                                                       quesses[255]: bool
                                                                                            + Fuse(time : int)
                                                       + Puzzle(solution : string)
                                                                                           + burn(): bool
     _quesses[c] = true;
                                                       + quess(c : char) : bool
                                                                                           + to string(): string
    return true;
                                                       + solve(proposed solution : string) : string
                                                       + to string(): string
                                                       + get solution(): string
bool Puzzle::solve(string proposed_solution) {
  return (proposed_solution == _solution);
string Puzzle::to_string() {
  string result = "";
  for (char c: solution)
     result += _guesses[c] ? c : '_';
  return result;
string Puzzle::get_solution() {
  return solution;
```

Homework Retrospective Full Credit – Main (1 of 2)

```
Main
#include <iostream>
                                                                               + main(): int
#include "puzzle.h"
#include "fuse.h"
                                                                 Puzzle
                                                                                            Fuse
using namespace std;

    solution : string

    time : int

                                                    quesses[255]: bool
                                                                                      + Fuse(time : int)
                                                    + Puzzle(solution : string)
                                                                                      + burn(): bool
int main() {
                                                    + guess(c : char) : bool
                                                                                      + to string(): string
  Puzzle puzzle
                                                    + solve(proposed solution : string) : string
                                                    + to string(): string
       {"a stitch in time saves nine"};
                                                    + get solution() : string
  Fuse fuse{8};
  char quess;
  string proposed solution;
  bool winner = false; // true if the player won
  cout << "
                       << "
                           B O O M !"
                                            << endl
       << "
                       ========= << endl << endl;
  cout << "Enter lower case letters to guess, " << endl
        << "! to propose a solution," << endl
        << "0 to exit." << endl << endl;
```

Homework Retrospective Full Credit – Main

```
while(true) {
  cout << endl << fuse.to string() << puzzle.to string() << ": ";</pre>
  cin >> guess;
  if (quess == '0') {
    exit(0);
  } else if (quess == '!') {
    cout << "Disarming the firecracker - what is the solution? ";</pre>
    cin.iqnore();
    getline(cin, proposed_solution);
    winner = puzzle.solve(proposed solution);
    break;
  } else {
    if (puzzle.guess(guess)) {
      if (!fuse.burn()) break;
     } else {
      cerr << "Invalid character - try again" << endl;</pre>
if (winner) {
  cout << "*** W I N N E R ***" << endl;
} else {
  cout << "##### BOOM ######" << endl;
  cout << "The answer was: '" << puzzle.get solution() << "'" << endl;</pre>
```

Homework Retrospective Full Credit – Makefile

```
# Makefile for Boom
CXXFLAGS += --std=c++11
all: main
debug: CXXFLAGS += -q
debug: main
rebuild: clean main
main: main.o puzzle.o fuse.o
        $(CXX) $(CXXFLAGS) main.o puzzle.o fuse.o
main.o: main.cpp puzzle.h fuse.h
        $(CXX) $(CXXFLAGS) -c main.cpp
puzzle.o: puzzle.cpp puzzle.h
        $(CXX) $(CXXFLAGS) -c puzzle.cpp
fuse.o: fuse.cpp fuse.h
        $(CXX) $(CXXFLAGS) -c fuse.cpp
clean:
        -rm -f *.o *~ a.out
```

Homework Retrospective Full Credit – Game in Progress

```
a stit__ i_ ti__ sa__s _i__: n
a stit_ in ti_ sa_s nin_: !
Disarming the firecracker - what is the solution? a stitch in time saves nine
*** W I N N E R ***
ricegf@pluto:~/dev/cpp/201701/P3/fc$
```

Homework Retrospective Bonus – Puzzle Class

```
#include <string>
#include <exception>
using namespace std;
class Puzzle {
  public:
    Puzzle(string solution) : _solution{solution} {_guesses[' '] = true;}
    class Bad char : public exception { };
    bool quess(char c);
                                           // true if char is in solution
                                           // throws Bad char is invalid
    bool solve(string proposed_solution); // true if correctly guessed
    string to_string();
    string get solution();
  private:
    string solution;
    bool _quesses[255] = {false};
```

```
bool Puzzle::guess(char c) {
   if (c<'a' || c>'z' || _guesses[c]) throw Bad_char{ };
   _guesses[c] = true;
   for (char a : _solution) {
     if (a == c) return true;
   }
   return false;
}
```

Homework Retrospective Bonus – Main

```
while(true) {
  cout << endl << fuse.to string() << puzzle.to string() << ": ";</pre>
  cin >> guess;
  if (guess == '0') {
    exit(0);
  } else if (quess == '!') {
    cout << "Disarming the firecracker - what is the solution? ";</pre>
    cin.ignore();
    getline(cin, proposed_solution);
    winner = puzzle.solve(proposed solution);
    break;
  } else {
    try {
      if (!puzzle.quess(quess)) {
        if (!fuse.burn()) break; // fuse burned up - BOOM!
    } catch (Puzzle::Bad_char e) {
      cerr << "Invalid character - try again" << endl;</pre>
```

Homework Retrospective Bonus – Game in Progress

Homework Retrospective Bonus - ddd Source Commands Status

(); | main.cpp:24 GNU DDD 3.3.12 (x86_64-pc-linux-gnu), by Dorothea LReading symbols from a.out...done. (gdb) break main.cpp:24 Breakpoint 1 at 0x4015ff; file main.cpp, line 24. (gdb) run Starting program: /home/ricegf/dev/cpp/201701/P3/bonus/a.out _____ B O O H ! #include <iostream> Enter lower case letters to guess. #include "puzzle.h" ! to propose a solution. #include "fuse.h" 0 to exit. using namespace std; int main() { Puzzle puzzle{"a stitch in time saves nine"}: Fuse fuse {83; char guess: string proposed_solution: bool winner = false; // true if the player won cout << " ======= " << end1 Breakpoint 1, main () at main.cpp:24 << " B O O H !" << endl (gdb) graph display winner at (119, 42)

Display 1: winner (enabled, scope main, address 0x7fffffffdcaf)

(gdb)

```
while(true) {
   cout << endl << fuse.to_string() << puzzle.to_string() << ": ";</pre>
   cin >> guess;
🎟 if (guess == '0') €
     exit(0):
   3 else if (guess == '!') {
     cout << "Disarming the firecracker - what is the solution? ";
     cin.ignore():
     getline(cin, proposed_solution);
     winner = puzzle.solve(proposed_solution);
     break:
   3 else €
     try {
       if (!puzzle.guess(guess)) {
         if (!fuse.burn()) break;
```

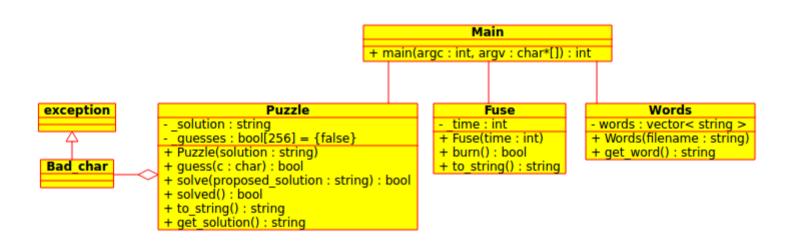
cout << "Enter lower case letters to guess, " << endl << "! to propose a solution," << endl</pre>

<< "0 to exit." << endl << endl;</pre>

View

Program

Homework Retrospective Extreme Bonus – UML



Homework Retrospective Extreme Bonus – Words

```
#include <string>
#include <vector>

using namespace std;

class Words {
  public:
    Words(string filename);
    string get_word();
  private:
    vector<string> words;
};
```

Words loads a phrase list file into the vector "words" during construction, then returns a random word on request. Puzzle and Fuse are unchanged.

```
#include "words.h"
#include <iostream>
#include <fstream>
string Words::get_word() {
    return words[rand() % words.size()];
Words::Words(string filename) {
  srand(time(NULL));
  trv {
    string word;
    ifstream ifs;
    ifs.open(filename, ifstream::in);
    while (getline(ifs, word)) {
      words.push back(word);
    ifs.close();
  } catch(exception e) {
    cerr << "Unable to load wordlist: "
         << filename << endl;
    exit(1);
```

Homework Retrospective Extreme Bonus – Main (1 of 2)

```
int main(int argc, char *argv[]) {
 srand(time(NULL)); // Randomize the random number generator rand()
 // Select a word for the game
 string solution = "a stitch in time saves nine";
 if (argc == 2) {
   Words wordlist{argv[1]};
   solution = wordlist.get word();
 Puzzle puzzle{solution};
 Fuse fuse{8};
 char guess;
 string proposed solution;
 bool winner = false; // true if the player won
 cout << "
                 << "
                    B 0 0 M !" << endl
     cout << "Enter lower case letters to guess, " << endl
      << "! to propose a solution," << endl
      << "0 to exit." << endl << endl:
```

This is all that is needed to meet the requirements.

But the program has a serious usability problem –

it doesn't know when the puzzle is solved until you tell it!

Homework Retrospective Extreme Bonus – Main (2 of 2)

```
while(!winner) {
  cout << endl << fuse.to_string() << puzzle.to_string() << ": ";</pre>
  cin >> quess;
  if (guess == '0') {
    exit(0);
  } else if (quess == '!') {
    cout << "Disarming the firecracker - what is the solution? ";</pre>
    cin.iqnore();
    getline(cin, proposed solution);
                                                     bool Puzzle::solved() {
    winner = puzzle.solve(proposed_solution);
                                                       for (char a : _solution) {
    break:
                                                         if (! quesses[a]) return false;
  } else {
    trv {
                                                       return true;
      if (!puzzle.quess(quess)) {
        if (!fuse.burn()) break;
    } catch (Puzzle::Bad_char e) {
      cerr << "Invalid character - try again" << endl;</pre>
  winner = puzzle.solved();
```

These three changes also detect when the puzzle is solved by guessing all of the letters.

Homework Retrospective Extreme Bonus – Makefile

```
# Makefile for Roving Robots
CXXFLAGS += --std=c++11
all: main
debug: CXXFLAGS += -g
debug: main
rebuild: clean main
main: main.o puzzle.o fuse.o words.o
        $(CXX) $(CXXFLAGS) main.o puzzle.o fuse.o words.o
main.o: main.cpp puzzle.h fuse.h words.h
        $(CXX) $(CXXFLAGS) -c main.cpp
puzzle.o: puzzle.cpp puzzle.h
        $(CXX) $(CXXFLAGS) -c puzzle.cpp
fuse.o: fuse.cpp fuse.h
        $(CXX) $(CXXFLAGS) -c fuse.cpp
words.o: words.cpp words.h
        $(CXX) $(CXXFLAGS) -c words.cpp
clean:
        -rm -f *.o *~ a.out
```

Homework Retrospective Extreme Bonus–Games in Progress

```
The answer was: 'swim'
riceaf@pluto:~/dev/cpp/201701/P3/xb
```

```
or ar : d
or ard: f
The answer was: 'forward'
  ceaf@pluto:~/dev/cpp/201701
```

```
e otion: r
e otion: m
The answer was: 'demotion'
 iceaf@pluto:~/dev/cpp/201701
```