```
 1: #include <BearLibTerminal.h>
 2: #include <cmath>
 3: #include <iostream>
 4: #include <cstdlib>
 5: #include <ctime>
 6: #include <fstream>
 7: using namespace std;
 8: #include <BearLibTerminal.h>
 9: #include "gooseEscapeUtil.hpp"
10: #include "gooseEscapeActors.hpp"
11: #include "gooseEscapeConsole.hpp"
12: #include "gooseEscapeGamePlay.hpp"
13:
14: extern Console out;
15: /*
16: With graphics, screens are given an x,y coordinate system with the origin
17: in the upper left corner.  So it means the coordinate axes are:
18: ---------------->   x direction
19: |
20: |
21: |
22: |
23: |
24: V
25:
26: y direction
27: */
28:
29: const int MAN_STEP=1;
30: const float GOOSE_STEP=1;
31: /*
32:     Print the game world
33:
34:     The functions should draw characters to present features of the game
35:     board, e.g. win location, obstacles, power ups
36: */
37:
38: // print the game board function
39: void print_board(int x, int y, int feature, char featureChar,
40:                     int game_world[NUM_BOARD_X][NUM_BOARD_Y])
41: {
42:     game_world[x][y]=feature;
43:     terminal_put(x,y,featureChar);
44: }
45:
46:
47: void setup (ifstream & fin_wall,ifstream & fin_power,
48:                         int game_world[NUM_BOARD_X][NUM_BOARD_Y])
49: {
50:     int straight=0,num=0,first_line=0,last_line=0,power_x=0,power_y=0;
51:     while(fin_wall>>straight>>num>>first_line>>last_line)
52:     {
53:         for(int rowcol=first_line;rowcol<=last_line;rowcol++)
54:         {
```

```
55:             if(straight!=0)
56:             {
57:                 print_board(rowcol,num,SHALL_NOT_PASS,
58:                     WALL_CHAR,game_world);
59:             }
60:             else
61:             {
62:                 print_board(num,rowcol,SHALL_NOT_PASS,
63:                     WALL_CHAR,game_world);
64:             }
65:         }
66:     }
67:     print_board(WIN,WIN,WINNER,WIN_CHAR,game_world);
68:     while(fin_power>>power_x>>power_y)
69:     {
70:         print_board(power_x,power_y,POWER,POWER_CHAR,game_world);
71:     }
72:
73:
74:     terminal_refresh();
75: }
76:
77: /*
78: Do something when the goose captures the player
79:
80: If you want to attack or something else, this is the function you
81: need to change.  For example, maybe the two touch each other and
82: then fight.  You could add a health to the Actor class that is
83: updated.  Run, use weapons, it's up to you!
84: */
85:
86: bool captured(Actor const & player, Actor const & monster)
87: {
88:     return (player.get_x() == monster.get_x()
89:          && player.get_y() == monster.get_y());
90: }
91:
92: /*
93: Move the player to a new location based on the user input.You may want
94: to modify this if there are extra controls you want to add.
95:
96: All key presses start with "TK_" then the character.So "TK_A" is the a
97: key being pressed.
98:
99: A look-up table might be useful.
100: You could decide to learn about switch statements and use them here.
101: */
102:
103: /*we have created a function that takes in the parameters where
104: the goose and the student pass through and if the goose goes through
105: any wall or passes the safe spot while catching the student then the
106: below function will replace that wall or the safe spot icon.*/
107: void wall_win_rebuilder(int x,int y,
108:                     int game_world[NUM_BOARD_X][NUM_BOARD_Y])
```

```
{
    if(game_world[x][y]==SHALL_NOT_PASS)
    {
        terminal_put(x,y,WALL_CHAR);
    }
    else if(game_world[x][y]==WINNER)
    {
        terminal_put(x,y,WIN_CHAR);
    }

    terminal_refresh();
}

void power_point_checker(Actor & player, Actor & goose,
                             int game_world[NUM_BOARD_X][NUM_BOARD_Y])
{
    if(game_world[player.get_x()][player.get_y()]==POWER)
    {
        goose.set_power(UP);
        player.set_power(UP);
        out.writeLine("Extra Energy for you and the goose!");
        out.writeLine("The Goose's speed has increased!!");
        out.writeLine("You get to make a jump of 20 steps just once"
                      " by pressing P");
        out.writeLine("Use it wisely buddy!!");

        game_world[player.get_x()][player.get_y()]==FREE;
    }
}


void movePlayer(int key, Actor & player,
                int game_world[NUM_BOARD_X][NUM_BOARD_Y])
{
    int yMove = 0, xMove = 0;
    if (key == TK_UP)
        yMove = -MAN_STEP;
    else if (key == TK_DOWN)
        yMove = MAN_STEP;
    else if (key == TK_LEFT)
        xMove = -MAN_STEP;
    else if (key == TK_RIGHT)
        xMove = MAN_STEP;

    int prev_X=player.get_x();
    int prev_Y=player.get_y();


    if (game_world[player.get_x()+xMove]
                [player.get_y()+yMove]!=SHALL_NOT_PASS)
    {
        player.update_location(xMove, yMove);
    }

```

```
163:    wall_win_rebuilder(prev_X,prev_Y,game_world);
164: }
165:
166:
167: void move_the_Goose(Actor &player, Actor &goose,
168:                     int game_world[NUM_BOARD_X][NUM_BOARD_Y])
169: {
170:     int x_move=0;
171:     int y_move=0;
172:     int big_step=1;
173:     if(!goose.power_up())
174:     {
175:         if(player.get_x()<goose.get_x())
176:         {
177:             x_move=-GOOSE_STEP;
178:         }
179:         else if(player.get_x()>goose.get_x())
180:         {
181:             x_move=GOOSE_STEP;
182:         }
183:         if(player.get_y()<goose.get_y())
184:         {
185:             y_move=-GOOSE_STEP;
186:         }
187:         else if(player.get_y()>goose.get_y())
188:         {
189:             y_move=GOOSE_STEP;
190:         }
191:         if (goose.can_move(x_move, y_move))
192:         {
193:             int prev_x=goose.get_x();
194:             int prev_y=goose.get_y();
195:
196:             goose.update_location(x_move, y_move);
197:             wall_win_rebuilder(prev_x,prev_y,game_world);
198:         }
199:     }
200:
201:     else
202:     {
203:         if(goose.power_up())
204:         {
205:             big_step=3;
206:
207:             if(player.get_x()<goose.get_x())
208:             {
209:                 x_move=max(-big_step,player.get_x()-goose.get_x());
210:             }
211:             else if(player.get_x()>goose.get_x())
212:             {
213:                 x_move=min(big_step,player.get_x()-goose.get_x());
214:             }
215:
216:             if(player.get_y()<goose.get_y())
```

```
217:             {
218:                 y_move=max(-big_step,player.get_y()-goose.get_y());
219:             }
220:             else if(player.get_y()>goose.get_y())
221:             {
222:                 y_move=min(big_step,player.get_y()-goose.get_y());
223:             }
224:         }
225:     }
226:     int prev_x=goose.get_x();
227:     int prev_y=goose.get_y();
228:     goose.update_location(x_move, y_move);
229:     wall_win_rebuilder(prev_x,prev_y,game_world);
230:
231: }
232:
233:
234:
235: void s_power (int key, Actor & player,
236:                     int game_board[NUM_BOARD_X][NUM_BOARD_Y])
237: {
238:     player.set_power(!POWER);
239:     int yMove=0, xMove=0;
240:
241:     if(key==TK_RIGHT)
242:     {
243:         xMove=SUPE;
244:
245:         if((xMove + player.get_x())>MAX_BOARD_X)
246:         {
247:             xMove=MAX_BOARD_X-player.get_x();
248:         }
249:     }
250:
251:     else if (key==TK_LEFT)
252:     {
253:         xMove=-SUPE;
254:
255:         if((xMove + player.get_x())<MIN_BOARD_X)
256:         {
257:             xMove=MIN_BOARD_X-player.get_x();
258:         }
259:     }
260:     else if (key==TK_UP)
261:     {
262:         yMove=-SUPE;
263:         if((yMove + player.get_y())<MIN_BOARD_Y)
264:         {
265:             yMove=MIN_BOARD_Y-player.get_y();
266:         }
267:     }
268:
269:     else if (key==TK_DOWN)
270:     {
```

```
271:          yMove=SUPE;
272:          if((yMove + player.get_y())>MAX_BOARD_Y)
273:          {
274:              yMove=MAX_BOARD_Y-player.get_y();
275:          }
276:      }
277:
278:      player.update_location(xMove,yMove);
279: }
280:
281: /*
282: What other functions do you need to make the game work?  What can you
283: add to the basic functionality to make it more fun to play?
284: */
```