```
#ifndef GOOSE_ESCAPE_ACTORS
#define GOOSE_ESCAPE_ACTORS
#include <cmath>
#include <BearLibTerminal.h>
#include "gooseEscapeUtil.hpp"

/*
    Modify this class to contain more characteristics of the "actor".  Add
    functions that will be useful for playing the game that are specific to
    the Actor.

    Feel free to add additional Classes to your program.
*/

/*
    Going further:  Learn the other syntax for implementing a class that is
    more appropriate for working with multiple files, and improve the class code.
*/

class Actor
{
  private:
    bool power;
    int actorChar;
    int location_x, location_y;

  public:

    Actor()
    {
        actorChar = int('A');
        power=false;
        location_x = MIN_SCREEN_X;
        location_y = MIN_SCREEN_Y;
        put_actor();
    }

    Actor(char initPlayerChar, int x0, int y0)
    {
        change_char(initPlayerChar);
        location_x = MIN_SCREEN_X;
        location_y = MIN_SCREEN_Y;
        update_location(x0,y0);
    }

    void set_power(bool jump)
    {
        power=jump;
    }

    bool power_up() const
    {
        return power;
    }

```

```
56:
57:     int get_x() const
58:     {
59:         return location_x;
60:     }
61:
62:     int get_y() const
63:     {
64:         return location_y;
65:     }
66:
67:     string get_location_string() const
68:     {
69:         char buffer[80];
70:         itoa(location_x,buffer,10);
71:         string formatted_location = "(" + string(buffer) + ",";
72:         itoa(location_y,buffer,10);
73:         formatted_location += string(buffer) + ")";
74:         return formatted_location;
75:     }
76:
77:     void change_char(char new_actor_char)
78:     {
79:         actorChar = min(int('~'),max(int(new_actor_char),int(' ')));
80:     }
81:
82:     bool can_move(int delta_x, int delta_y) const
83:     {
84:         int new_x = location_x + delta_x;
85:         int new_y = location_y + delta_y;
86:
87:         return new_x >= MIN_BOARD_X && new_x <= MAX_BOARD_X
88:           && new_y >= MIN_BOARD_Y && new_y <= MAX_BOARD_Y;
89:     }
90:
91:     void update_location(int delta_x, int delta_y)
92:     {
93:         if (can_move(delta_x, delta_y))
94:         {
95:             terminal_clear_area(location_x, location_y, 1, 1);
96:             location_x += delta_x;
97:             location_y += delta_y;
98:             put_actor();
99:         }
100:    }
101:
102:    void put_actor() const
103:    {
104:        terminal_put(location_x, location_y, actorChar);
105:        terminal_refresh();
106:    }
107:
108:    int distance(Actor const & otherActor) const
109:    {
110:        return round(sqrt(pow(((*this).location_x-otherActor.location_x),2)
```

```
111:                             *pow(((*this).location_y-otherActor.location_y),2)));
112:     }
113:
114: };
115: #endif
```