Inheritance-Based Connect 4

Implement an inheritance-based version of the Connect 4 game. Connect 4 builds on a playing field composed out of 7 columns each having 6 rows. When a player puts a stone into a column, gravity pulls the stone towards the lowest unoccupied row. The player who first has 4 stones in a row (horizontally, vertically, or diagonally) wins.

After each turn display the game field using simple ASCII graphics. Implement the game in such a way that players can be exchanged easily. To facilitate this, implement a playfield class that is based on the following playfield skeleton. You may extend these classes but you are not allowed to modify the class itself.

Implement a human player as well as a computer player, each implementing the player interface. For the computer-player, implement it in a way that you can use the same code for the template-based version and the inheritance-based version with, if necessary, a small façade or adapter.

This game will later-on be extended to allow exchanging players. To facilitate this, implement a playfield class that is based on the playfield interface. Do not change this interface.

```
#ifndef PLAYFIELD_H_
#define PLAYFIELD_H_

class playfield {
public:
  // the size of the field
  const static int width=7;
  const static int height=6;

  // the elements stored at individual field positions
  const static int none=0;
  const static int player1=1;
  const static int player2=2;

  // returns the stone (none/player1/player2) at the position
  // 0 <= x <= width
  // 0 <= y <= height
  // stoneat(0,0) ................ top left
  // stoneat(width-1,height-1) ... bottom right
  // if we insert a stone in a new game in column i,
  // it lands at (i,height-1)
  virtual int stoneat(int x, int y) const = 0;
  virtual ~playfield() {}
};

#endif /* PLAYFIELD_H_ */
```

Implement a computer player. The computer player of this version does not have to be intelligent. At a minimum, however, the computer player should be able to identify whether the opponent can win the game by placing a stone and block this.

The computer player shall inherit from the player interface. It can access the playfield through the playfield interface. Your player should not rely on any extra functionality provided by the playfield class. That is, no down-casts from the playfield interface are permitted. Do not modify the player interface.

```
#ifndef PLAYER_H_
#define PLAYER_H_

#include <iostream>

#include "playfield.h"
#include "my_playfield.h"

class player {
public:
  virtual int play(const playfield &field) = 0;
  virtual ~player() {}
};

#endif /* PLAYER_H_ */
```