Assignment 2: Hi-Lo Game

Due: 19:00, Thu 15 Oct 2015 Full marks: 100

Introduction

In this assignment, you will develop a program that plays a game called *Hi-Lo* (開口中). A secret number between 1 and 100 (inclusive) is generated randomly. Multiple players take turns to make guesses and will be responded with a **HI** or **LO** message. **HI** means the guess is larger than the secret number and **LO** means the guess is smaller than the secret number. With more and more guesses, the possible range of the secret number becomes narrower and narrower. The goal of each player is to avoid revealing the secret number when making guesses. Multiple games will be played, each with a different secret number. The player(s) who revealed the secret number the fewest number of times is/are the overall winner(s).

Program Specification

1. At the beginning, your program should prompt the user to enter the number of players. (The players will be called Player 1, Player 2, … hereafter.)
2. Only 2–4 players are accepted. In case the input number of players is not within 2–4, display a warning message and go back to step 1.
3. Then, the program should prompt to input the number of games to be played. (The games will be called Game #1, Game #2, … hereafter.)
4. The number of games to be played must not be smaller than the number of players. In case this input is invalid, display a warning message and go back to step 3.
5. After setting up the number of players and the number of games, insert this statement in your program.

   ```
   srand(n * g);
   ```

   where `n` is the number of players and `g` is the number of games to be played. You also have to insert `#include <cstdlib>` at the beginning of your program.
6. Then repeatedly play a Hi-Lo game `g` times as follows.
   6.1. Randomly generate a secret number within 1–100 (inclusive) with this statement:

   ```
   secret = rand() % 100 + 1;
   ```

   6.2. Then players take turns (Player 1, Player 2, …, Player `n`, Player 1, …) to make guesses within an acceptable range. In Game #1, Player 1 is the first player to guess, while from Game #2 onward, the first player to guess is the loser in the previous game. (E.g., suppose Player 3 lost in Game #1. Then Player 3 will be the first player to guess in Game #2.)
   6.3. If the guess is outside the acceptable range (e.g., the current range is 6–89 but a player enters 3 or 91), you need to display a warning message and ask the same player to enter again. Repeat this range check until a valid guess is made.
   6.4. In case the guess does not reveal the secret number, respond with a **HI**/**LO** message accordingly and narrow the acceptable range for the next player.
   6.5. In case a player reveals the secret number, this game is finished and he/she loses this game.
7. After `g` games are played, display the statistics of the number of games that each player loses.

8. Finally, display the overall winner(s), which is/are the player(s) who lose(s) the fewest number of games. Note that there can be more than one overall winner; you have to display all of them.

Notes:

- You can assume that *all user input are always integers*.
- You *cannot* use arrays in this assignment. (Arrays will be taught in latter lectures.)
- You will notice that the generated secret numbers are not truly random in your program. The same user input in different program runs will always lead to the same sequence of secret numbers in the games. This program behavior is normal; we purposely design the assignment in this way so as to make our marking process easier.

Program Output

The following shows some sample output of the program. The **bold blue** text is user input and the other text is the program output. You can try the provided sample program for other input. *Your program output should be exactly the same as the sample program* (i.e., same text, same symbols, same letter case, same number of spaces, etc.). Otherwise, it will be considered as *wrong*, even if you have computed the correct result.

```
How many players (2-4)? 1↲
Number of players must be 2-4! Try again.
How many players (2-4)? 6↲
Number of players must be 2-4! Try again.
How many players (2-4)? 2↲
How many games? -1↲
At least 2 games! Try again.
How many games? 1↲
At least 2 games! Try again.
How many games? 2↲
Game #1:                                    Suppose secret number is 52.
Player 1, make a guess (1-100): 68↲
Your guess is HI.
Player 2, make a guess (1-67): 35↲
Your guess is LO.
Player 1, make a guess (36-67): 20↲
Guess must be 36-67! Try again.
Player 1, make a guess (36-67): 689↲
Guess must be 36-67! Try again.
Player 1, make a guess (36-67): 43↲
Your guess is LO.
Player 2, make a guess (44-67): 52↲
Boom! Player 2 loses.
Game #2:                                    Suppose secret number is 46.
Player 2, make a guess (1-100): 46↲
Boom! Player 2 loses.
Player 1 loses 0 time(s).
Player 2 loses 2 time(s).
Overall winner(s):
Player 1
```

```
How many players (2-4)? 3↲
How many games? 2↲
At least 3 games! Try again.
How many games? 3↲
Game #1:
Player 1, make a guess (1-100): 69↲
Boom! Player 1 loses.
Game #2:
Player 1, make a guess (1-100): 52↲
Boom! Player 1 loses.
Game #3:
Player 1, make a guess (1-100): 76↲
Boom! Player 1 loses.
Player 1 loses 3 time(s).
Player 2 loses 0 time(s).
Player 3 loses 0 time(s).
Overall winner(s):
Player 2
Player 3
```

Suppose secret number is 69.

Suppose secret number is 52.

Suppose secret number is 76.

Submission and Marking

- Your program file name should be `hilo.cpp`. Submit the file in Blackboard (https://elearn.cuhk.edu.hk/).
- Insert your name, student ID, and e-mail address as comments at the beginning of your source file.
- You can submit your assignment multiple times. Only the latest submission counts.
- Your program should be *free of compilation errors and warnings*.
- Your program should *include suitable comments as documentation*.
- *Plagiarism is strictly monitored and heavily punished if proven.* Lending your work to others is subjected to the same penalty as the copier.