**Assignment 2 (CSC341)**

(Some of you had a similar assignment in CSC226 with a bit different problem specification), but obviously you are expected to redo it with a different design)

You’d like to simulate a racing game among N players. The players move in sequence starting with player 1 and ending with player N, and the same cycle repeats as necessary. On each move, a player goes forward or backward depending on the player’s position change for the move. A positive position change implies a forward move, whereas a negative position change results in a backward move. As players move in sequence, the first player whose total move exceeds certain number of units (say M units) wins, and the game is over. The position change a player receives on each move is determined by:

* The outcome of throwing a fair die (you can, if needed, find references over the Web about how it is done in code)
* The relative position of the player who makes a move with respect to the positions of other players.
* The type of move a player uses.

There are three distinct types of move:

*Move Type 1*

The change in position is computed as:

*die throw + ( position of leading player – player’s position ) / 2*

*If the die throw is 3, 4, 5, or 6, then the change in position is positive; otherwise it is negative.*

*Move Type 2*

The change in position is computed as:

*3 times the die throw if the die throw is an even number; otherwise the die throw if the die throw is an odd number.*

(Thus, the change in position is always positive.)

Move Type 3

The change in position is computed as:

*die throw + (player’s position – position of the trailing player) / 2*

*If the die throw is 1 or 2, then the change in position is positive; otherwise it is negative.*

A player randomly picks a move type, and uses it for a random number (between 2 and 5) of times (this random number is called the *lifetime* of a move type). When a lifetime becomes 0, the player picks randomly a move type again and another lifetime). As the contest progresses and after each player’s move, the console output should display the player and its current position. When the game is over, the console output should display the winner of the game, and a table (of any format) that identifies the players and their final positions.

*Here is an example with 4 players:*

At some point, the four players are at positions shown in the following table:

Player 1 2 3 4

Position 10 5 12 7

Suppose player 3 is making a move. She selects move type 2 (by luck). So she throws a die (assuming it’s a 2). Thus her position change would be 3\*2 = 6 (according to the type 2 move logic). So her new position would be 18. Now, game continues and it’s player 4’s turn….

**Requirements:**

1. A class diagram (hand-drawing is fine) including all the relations you identified. For each class, show data fields and the methods.
2. Write about your thought process that led to the design.
3. A working program

*Basic principles we have covered:*

* *Design responsibility-driven elements*
* *Keep concerns separate (design concern-segregated interfaces)*
* *A program should be modular*
* *Prefer aggregation over inheritance (unless polymorphism is desired)*