**Semester Project**

Problem Solving and Programming in C++

**Department of Computer Science**

**Old Dominion University**

**Game Description :**

Note: Video with Description is Available on Blackboard

Patolli is an ancient board game originating in pre-Columbian Meso-America. It is played by 2-4 players on a 2D board shaped like a cross, illustrated in Figure 1. The goal is to traverse the entire board with all your pieces before your opponent does the same. The pieces move based on a die-roll and always in a clock-wise movement pattern.

Each player has his own start and end points indicated by circles in the figure (red-yellow for player 1, blue-gray for player 2). There are a few special squares on which different mechanics apply. The squares at the ends of the cross (marked red squares) grant an additional turn to the player whose piece ended its move there. The four squares in the middle are the only squares on which capturing an opponent’s piece is possible. The squares marked with black triangles are incorporated in a version of Patolli that includes gambling and will thus not be used in this variation of the game.

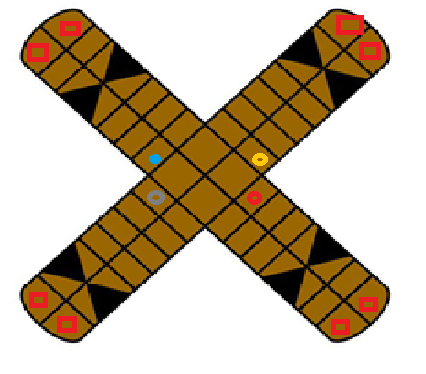


Figure 1

**Initial Position**

Initially, the board is empty. On the first turn, the player can place his piece on the starting square and end his turn. Then second player will then do the same. In this stage, the die roll doesn’t matter, regardless of the roll, the piece will start in its initial position and stay there until the next turn.

**Piece Movement**

Each player tosses a die in the beginning of the turn, the values on the dice are 0,1,2,3,4,5, and will be able to move a single piece by that many squares in the appropriate direction (red arrow in Figure 2). A piece can only move forward in the clockwise direction and not back. Alternatively, the player may place one of his pieces on the board (usually done on low dice rolls), or pass his turn.

A piece cannot occupy a square if another piece (enemy or not) is already there, except for the four middle squares. If an opponent’s piece occupies any of the middle squares, then, upon the appropriate die-roll, the player can remove that piece and replace it with his own. The removed piece goes back to the pool of available pieces for the opponent. Pieces can jump over both enemy and friendly pieces, as long as they don’t land in the same square.

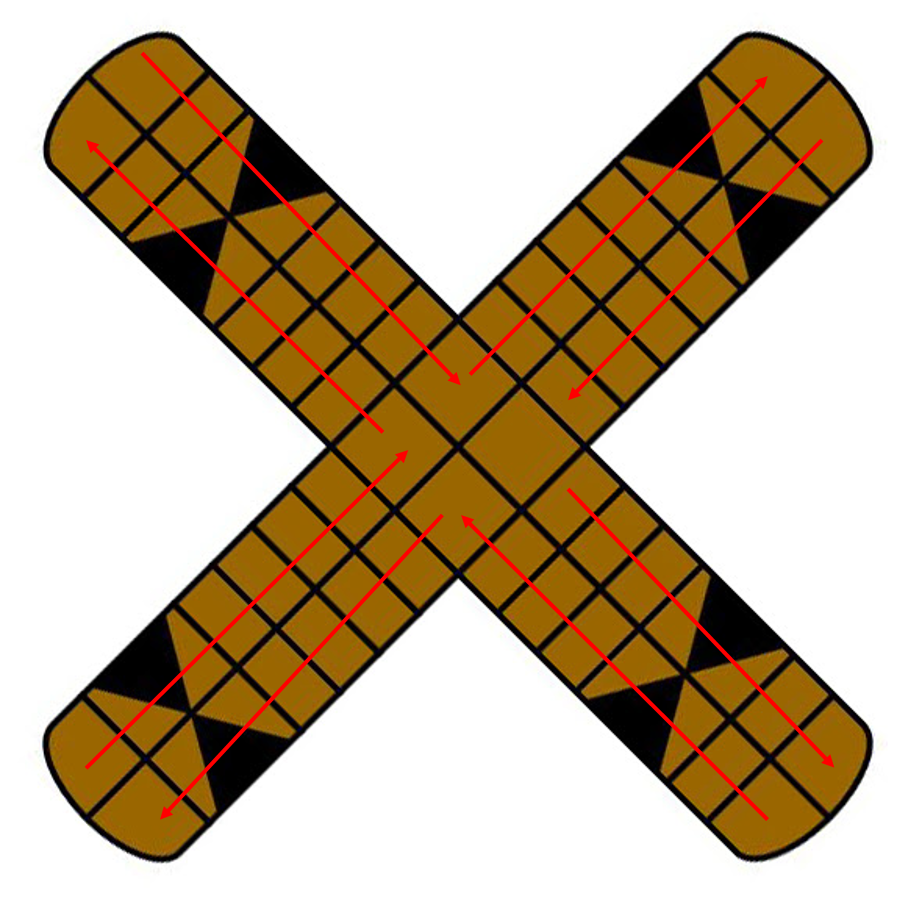


Figure 2

If a piece lands on any of the end squares (red squares on Figure 1.), the player may roll the die again and move again any of his pieces. The player can also choose to forfeit his turn, either for strategic purposes or in the case where none of his pieces can legally move. Rolling a 0 is equivalent to losing your turn.

When a piece reaches the end point (see Figure 1.), it is removed from the board and the player’s score is incremented by one (the scoring piece may not be played again). The piece has to land exactly on that end square. If based on the die roll, the piece overshoots and passes the end square, it must complete another turn in order to increment the score and the turn it has just completed will not count.

The first player who has moved all six of his pieces to the end-point, wins.

**Problem Description:**

You are to implement the variant of Patolli specified above, for two human players. You must implement three ADTs, one for a player, one for the board, and one for the referee.

At each turn the program should show the die roll and prompt the player to a select a piece to move. In case of incorrect input, the program should continue to prompt the user until a correct command is issued.

When a player wins, the game should stop and a message indicating which player is victorious must be printed on the screen.

Since all six pieces are identical but must be distinguished during the piece selection phase of the turn, you can represent them with A, B, C, D , E ,F for player 1, and 1, 2, 3, 4, 5, 6 for player 2. Based on this piece scheme, you can name player 1 as player L (Letter) and player 2 as player N. (Number).

**Grading:**

* Compilation 15
* Correct Output
  + Correct piece movement 10
  + Correct capturing mechanics 10
  + Correct game termination 5
  + Correct turn taking 5
  + Correct turn gain (land on edge squares) and turn loss (roll 0) 5
* Appropriate ADTs
  + Three ADTs 15
  + Use of private data members, accessor and mutators 15
  + ADT Cohesion 10
* Style: Comments, Indentation, Simplicity of Main 10

**Total** 100

Tip: When designing the structure of your ADTs, it is useful to think in terms of designing a player’s A.I., in order to see how the ADTs should interact. Your Referee should act as an interface between player and board, and not allow the player to cheat (player setting his own score, making illegal moves, etc.)

Example Output:



