EECS 293 Assignment 6

Keegan Simmons, Noah Houpt, Tyler Anderson

February 28, 2019

1 Objects

1.1 Gone

Gone represents the game of Gone.

1.1.1 Fields

Gone has two hidden fields, *rounds* and *gameBoard*. *rounds* is the number of rounds it takes to play Gone on the board configuration, and *gameBoard* holds information on current BoardConfiguration state.

1.1.2 Routines

Algorithm 1: Gone($boardConfiguration$)

Input : A BoardConfiguration $boardConfiguration$
Output: A Gone object with a valid gameBoard
$gameBoard \leftarrow boardConfiguration$

Algorithm 2: roundsAndBlackPebbleRemains

Input : None
Output: A Tuple of the number of iterations the game runs and whether or not a black pebble remains
while *gameBoard.morePebblesToReplace()* **do**
  $gameBoard.applyReplacementRulesOnce()$
  $rounds \leftarrow rounds + 1$
end
return $rounds$, $gameBoard.blackRemains()$

1.2 BoardConfiguration

BoardConfiguration represents the configuration of pebbles on the board.

1.2.1 Fields

The BoardConfiguration hides *boardMap*, a Map of pebble coordinates to their color.

1.2.2 Routines

Algorithm 3: BoardConfiguration($possibleBoardMap$)

Input : A Map of Coordinate objects mapped to PebbleColor values $possibleBoardMap$
Output: A BoardConfiguration object that is a valid board
Result : If the $possibleBoardMap$ is not a valid board, an $error$ will occur
if *isValidBoardConfiguration(*$possibleBoardMap$*)* **then**
  $boardMap \leftarrow possibleBoardMap$
end
else
  $error$, invalid input
end

Algorithm 4: isValidBoardConfiguration($boardMap$)

Input : A Map of Coordinate objects mapped to PebbleColor values $boardMap$
Output: $true$ if the $boardMap$ is valid, and $false$ otherwise
List $keys \leftarrow$a list of keys in the $boardMap$
noInvalidCoordinates $\leftarrow true$
foreach $coordinate$ *Coordinate object in* $keys$ **do**
  if $coordinate.x() < 0$ *OR* $coordinate.y() < 0$ **then**
    noInvalidCoordinates $\leftarrow false$
    break out of for loop
  end
end
return noInvalidCoordinates

Algorithm 5: applyReplacementRulesOnce

Input : None
Result : This BoardConfiguration object will have an updated boardMap for after the replacement rules have been applied once
List $whiteCoordinates \leftarrow whitePebbleCoordinates(coordinates)$
foreach $coordinate$ *in* $whiteCoordinates$ **do**
  foreach $neighbor$ *in* $coordinate.adjacentCoordinates()$ **do**
    if $neighbor$ *is a black pebble* **then**
      $boardMap$ value at $neighbor$'s key $\leftarrow white$
    end
  end
end
return $boardMap$

Algorithm 6: whitePebbleCoordinates

Input : None
Output: A List of all Coordinate objects mapped to a white pebble color in this $boardMap$
List $coordinates \leftarrow$a list of keys in this $boardMap$
List $whiteCoordinates \leftarrow$empty
foreach $coordinate$ *in* $coordinates$ **do**
  if $boardMap$ *value at* $coordinate$ *is white* **then**
    add $coordinate$ to $whiteCoordinates$
  end
end
return $whiteCoordinates$

Algorithm 7: morePebblesToReplace

Input : None
Output: A boolean value that is $true$ if there are pebbles that need to be replaced, and $false$ otherwise
$replacementNeeded \leftarrow false$
foreach *$coordinate$ key in this $boardMap$* **do**
  if *$boardMap$ value at $coordinate$ is white* **then**
    if *$coordinate$ has a neighbor with a black pebble* **then**
      $replacementNeeded \leftarrow true$
      break from the for each loop
    end
  end
  else
    if *$coordinate$ has a neighbor with a white pebble* **then**
      $replacementNeeded \leftarrow true$
      break from the for each loop
    end
  end
end
return $replacementNeeded$

Algorithm 8: blackRemains

Input : None
Output: A boolean value that is $true$ when there is a black pebble remaining on the board and $false$ otherwise
$blackPebbleFound \leftarrow false$
foreach *$coordinate$ key in this $boardMap$* **do**
  if *$boardMap$ value at $coordinate$ is black* **then**
    $blackPebbleFound \leftarrow true$
    break from the for each loop
  end
end
return $blackPebbleFound$

1.3 PebbleColor

PebbleColor is an enumeration that represents the color of the pebble. The enumerations are *Black* and *White*.

1.4 Coordinate

Coordinate represents the pebble coordinates.

1.4.1 Fields

It hides *x*, *y*, and *adjacent*. *x* is the integer x location on the board, *y* is the integer y location on the board, and *adjacent* is the List of Coordinate objects adjacent to this Coordinate.

1.4.2 Routines

Algorithm 9: Coordinate($inputX$, $inputY$)

Input : An integer representing an x location in a 2D coordinate plane $inputX$, an integer representing a y location in a 2D coordinate plane $inputY$
Output: A Coordinate object
$x \leftarrow inputX$
$y \leftarrow inputY$

Algorithm 10: adjacentCoordinates

Input : None
Output: The List of Coordinate objects adjacent to this Coordinate
if $adjacent$ *is empty* **then**
    $adjacentCoordinateList \leftarrow$empty
    add the neighboring coordinates to $adjacentCoordinateList$
    $adjacent \leftarrow adjacentCoordinateList$
end
return $adjacent$

Algorithm 11: x

Input : None
Output: The integer x location associated with this Coordinate
return x

Algorithm 12: y

Input : None
Output: The integer y location associated with this Coordinate
return y