# API

Voor ons programma hebben we de volgende API geschreven:

Deel 1, implementeren van een onopgeloste binairo

|  |
| --- |
| Binairo |
| -elements : int[][]  //a table consists of n rows, and n columns |
| +Binairo(int n)  pre: n > 0  post: has created a Binairo object with n rows and n columns, the value of all elements = -1.  +set(int row, int col, int val)  pre: 0 <= row < n, 0 <= col < n, val = -1 || val = 0 || val = 1  post: if pre and val = 0 , elements[row][col] = 0, if val = 1 elements[row][col] = 1, else elements[row][col] = -1.  +get(int row, int col) : int  pre: 0 <= row < n, 0 <= col < n  post: if pre returns elements[row][col], returns 7 otherwise  +numRows() : int  post: has returned the number of rows  //since the number of columns equals number of rows, there is no need for a ‘numColumns()’ method.  +toString() : String  post: has returned a String representation of the Binairo object |

Deel 2, checkMethods.

Gegeven de specificatie van klasse checkMethods.

|  |
| --- |
| **checkMethods** |
| **-check : boolean** |
| **+checkMethods()**  **post: check = true**  **+noMoreThanTwoRow(Binairo b)**  post: if 3 zeros or ones next to eachother, check = false  **+noMoreThanTwoColumn(Binairo b)**  post: if 3 zeros of ones above eachother, check = false  **+numberOfRows(Binairo b)**  post: if the number of ones, or zeros in the row is more than half of the row length, check = false  **+numberOfColumns(Binairo b)**  post: if the number of ones, or zeros in the column is more than half of the row length, check = false  **+identicalRows(Binairo b)**  Pre: the first two rows are filled with ones and zeros  post: if all the elements of 2 rows match, check = false  **+identicalColumns(Binairo b)**  Pre: all rows are filled with ones and zeros  Post: if all the elements of 2 columns match, check = false  **+getCheck() : boolean**  post: return check |

Deel 3, SolveBinairo

|  |
| --- |
| **SolveBinairo** |
| **+copy(Binairo b) : Binairo**  post: returns a copy of Binairo b  **+check(Binairo b) : Boolean**  post: if one of the checkmethods returns false, return false, otherwise return true    **Main method:**  //First we create the binairo:  We gebruiken een scanner om de grootte en de vaste getallen van de binairo in te voeren  //Solving the binairo:  We use backtracking to solve the binairo. We fill a copy of the given binairo. First we fill in 0, if that doesn’t satisfy all of the checkmethods, we fill in 1, and if that also doesn’t work go back until you find a spot that you can correctly fill in(backtracking). |