Tic tac toe demonstrator

Technical instructions

Table of contents

Upload and launch the program on the Arduino board

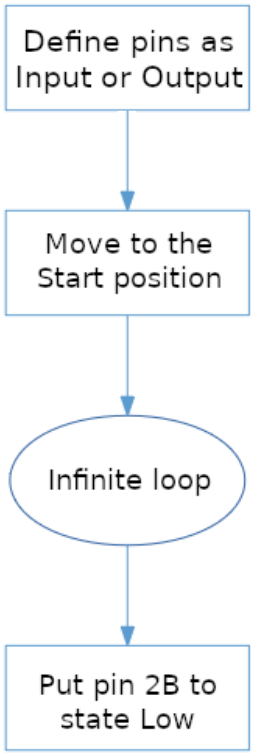
Once the program is written you can upload it to your Arduino board. To do so, you first need to tell to your Arduino environment on which type of board you will upload the program. Click on Tools, then Board and search for the right board “Arduino/Genuino Mega or Mega 2560”. Then you need to specify on which Port your board is connected. Click on Tool, then on Port and click on the one where there is the name of the board – for example, COM3 (Arduino/Genuino Mega or Mega 2560). You can finally click on the upload button (the one with an arrow).

Launch the program on Niryo Studio

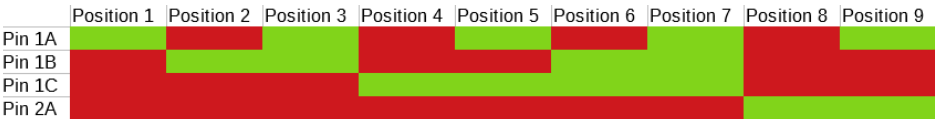
When programs are uploaded on their board, you need to launch and upload programs on Niryo Studio and on the robot.
Launch Niryo Studio, connect to the robot, click on this icon ( ) , then click on the two arrows icon to upload your code, search for the right program and upload it. Then click on the green arrow to start the program on the robot.

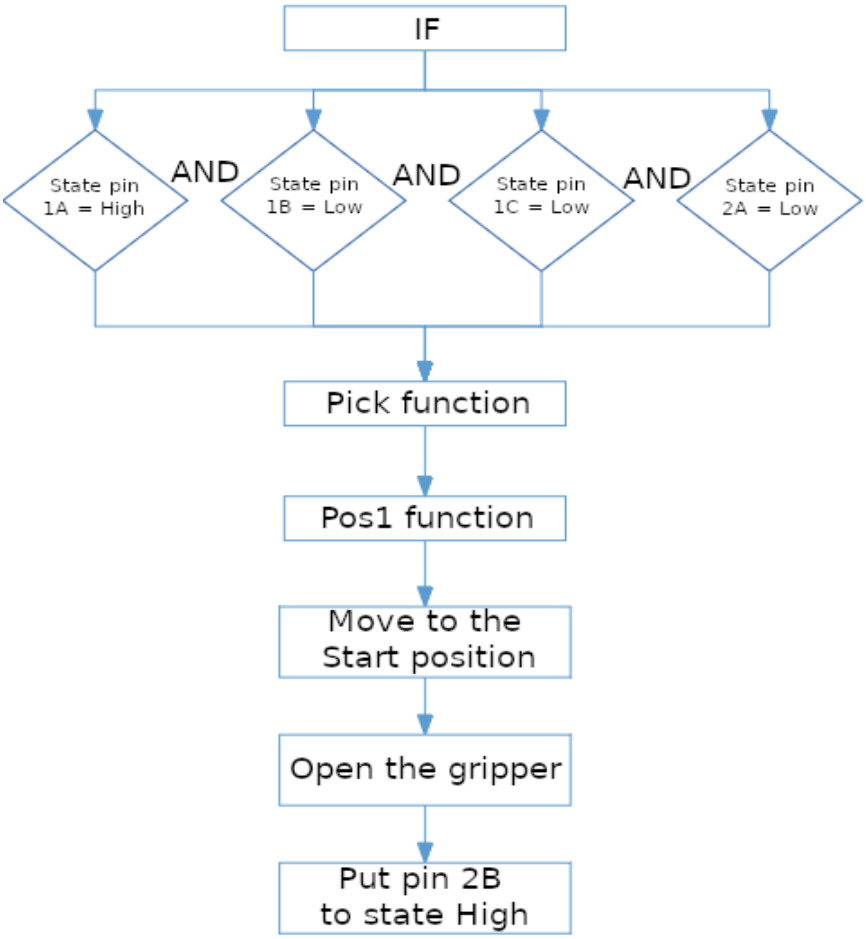
Basic explanation of the program

Robot

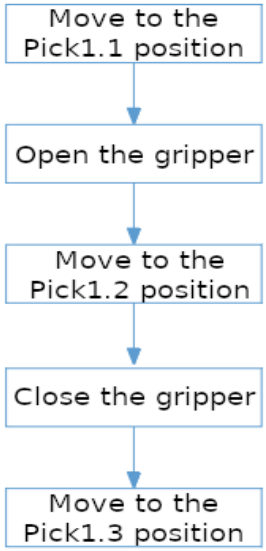


The robot can play on 9 different positions, we use a boolean logic to define which position to play. The 9 positions are defined according to the following table:

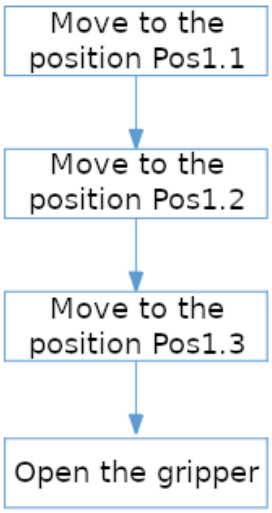




Each position are then defined according to the same scheme, and call the corresponding function (Pos1, Pos2, Pos3, …) according to the pins state.

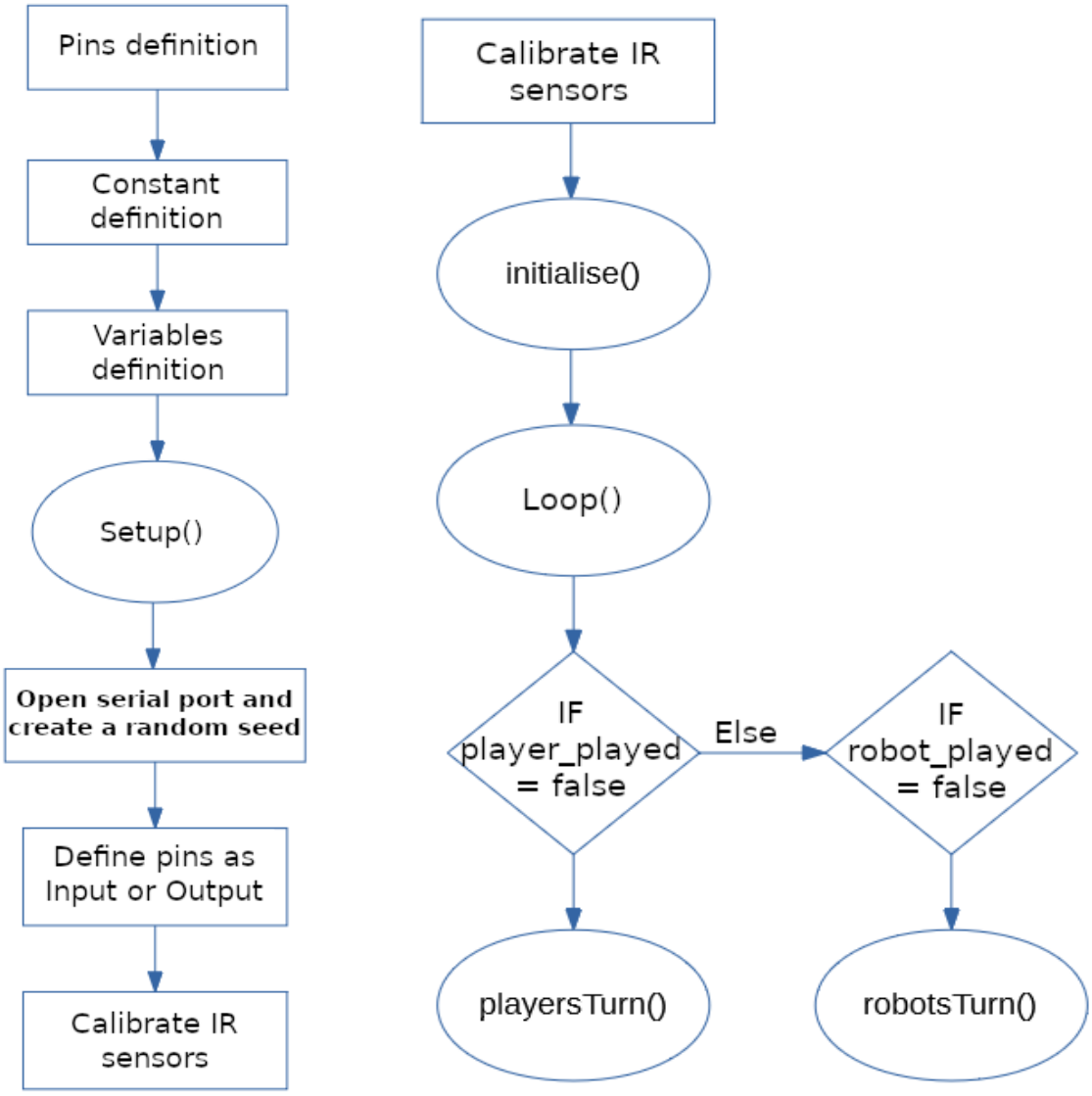


Description of the Pick function

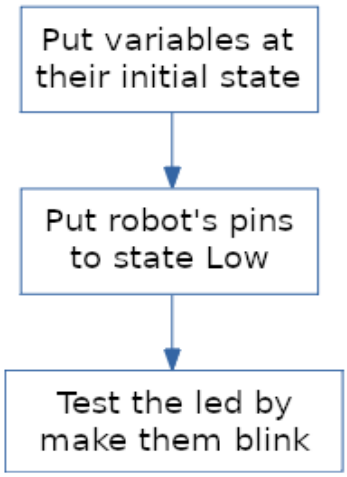


Description of the Pos1 function

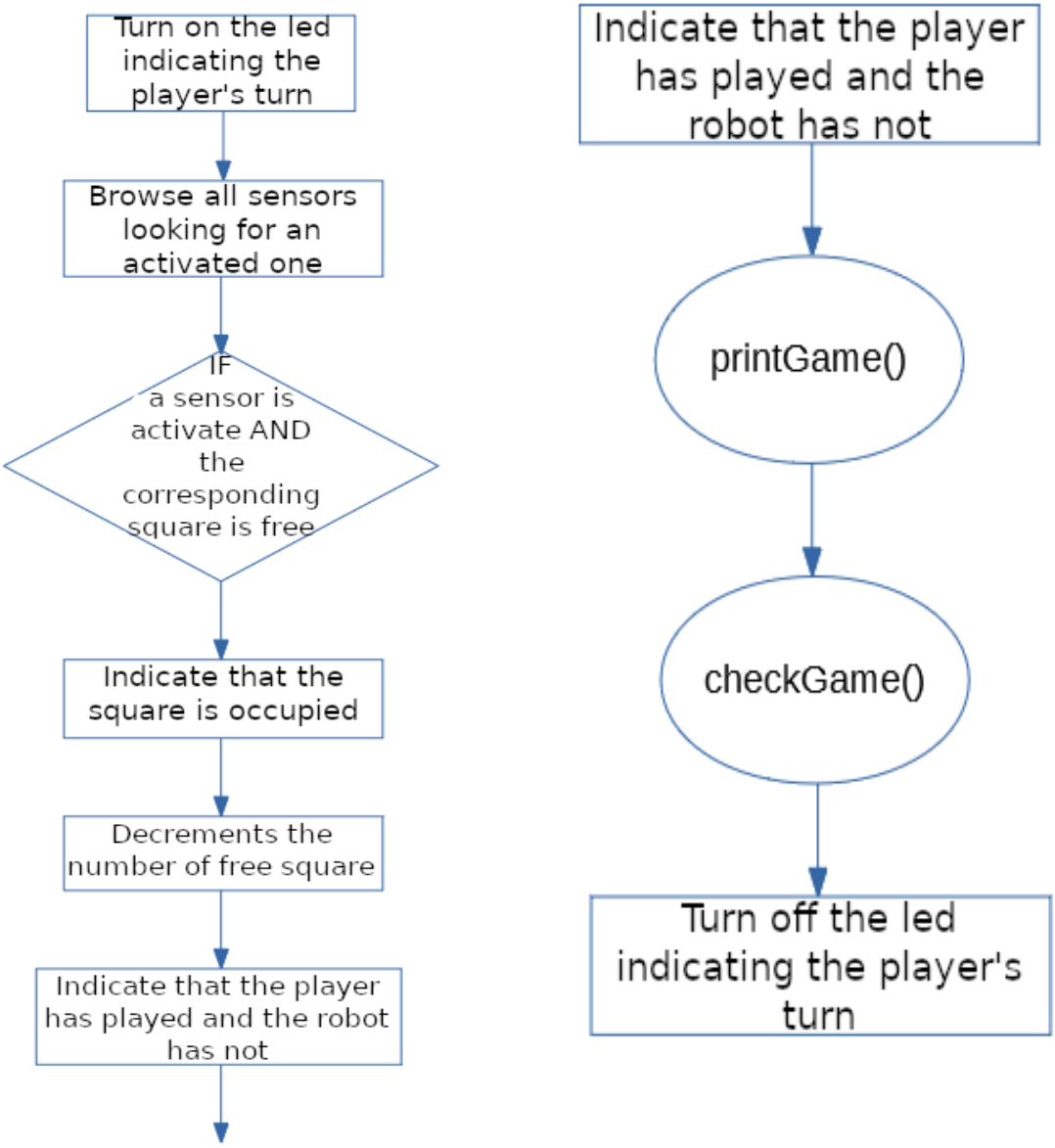
Arduino



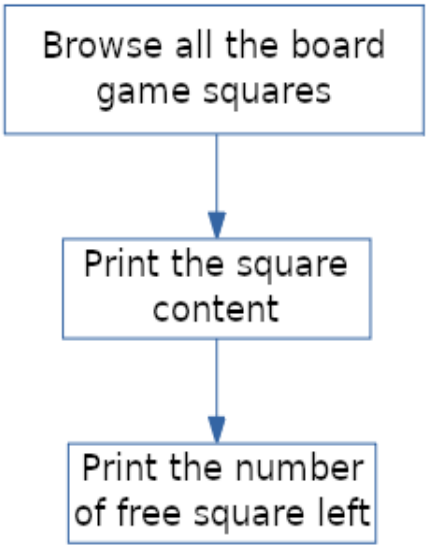
initialise()



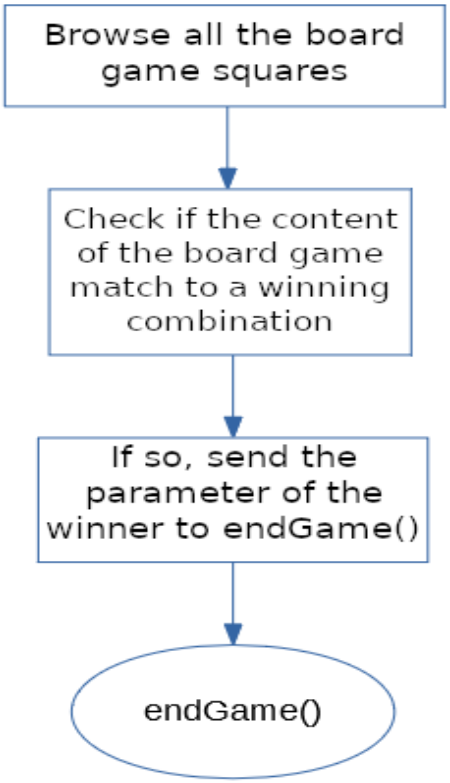
playersTurn()



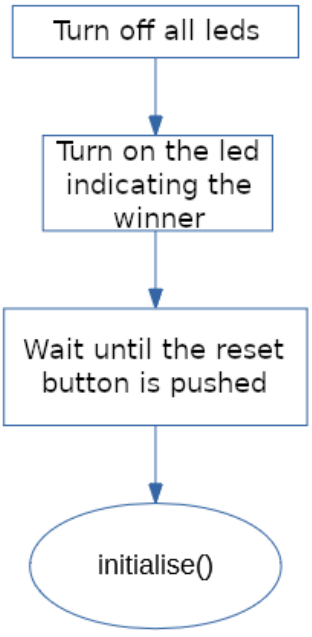
printGame()



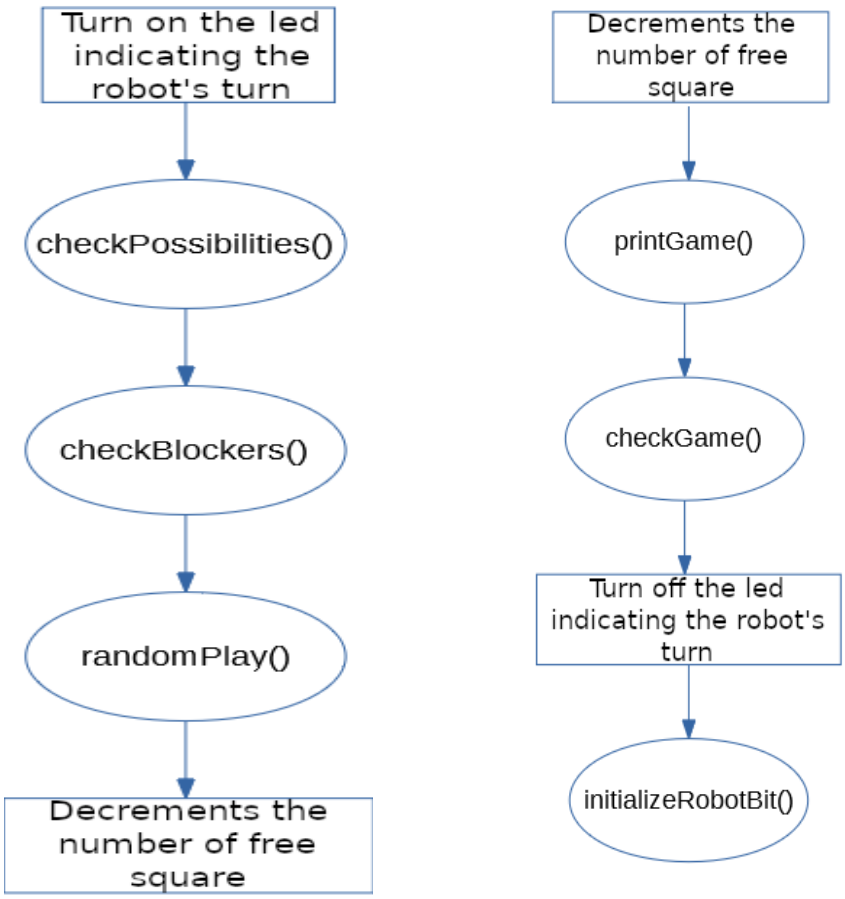
checkGame()



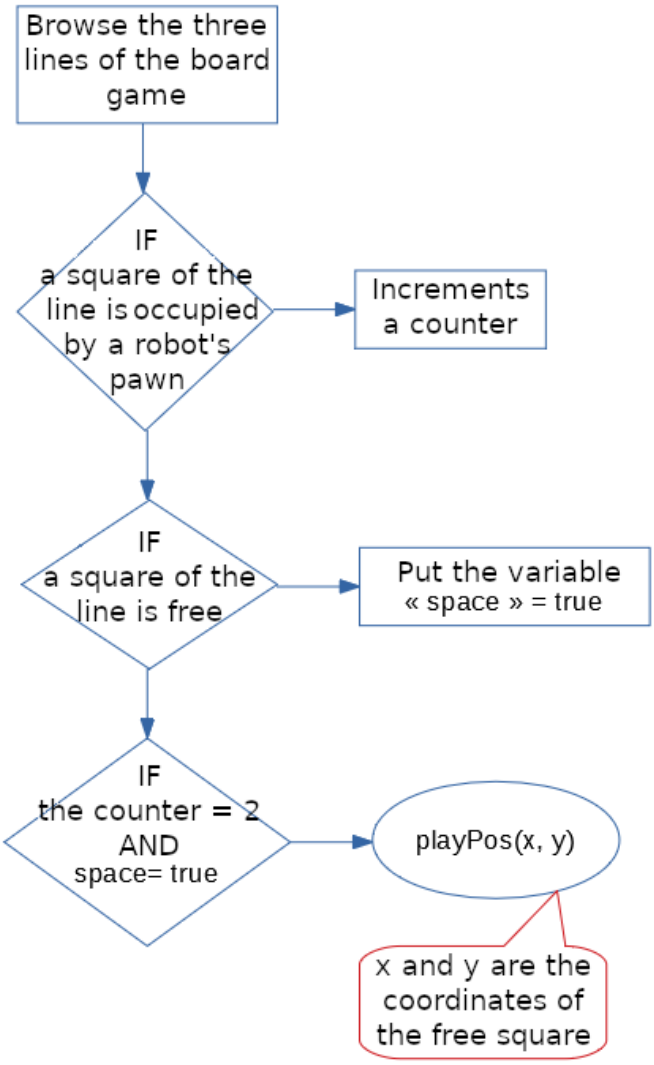
endGame()

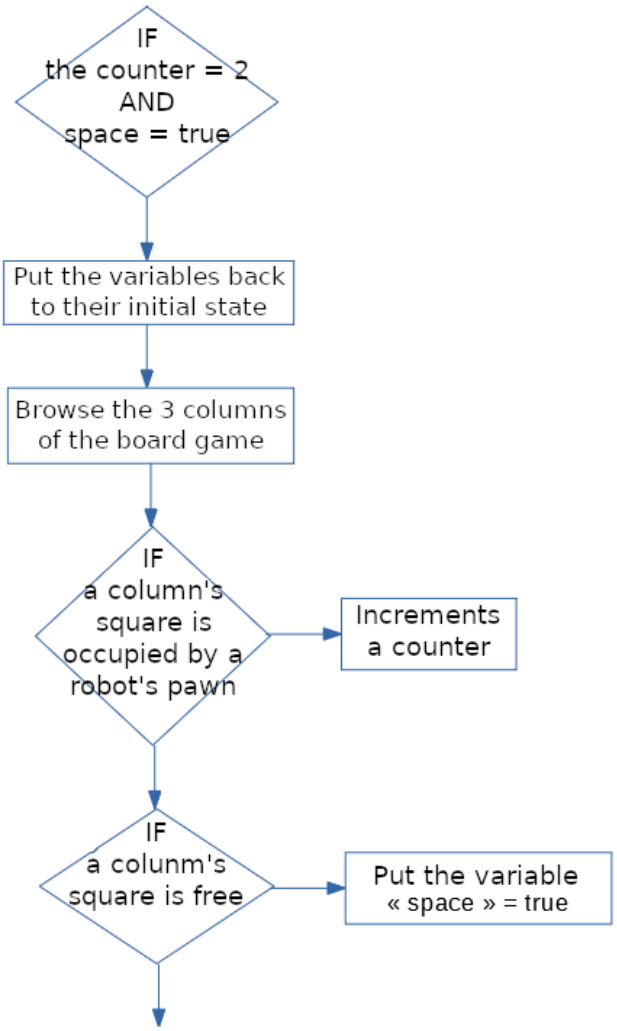


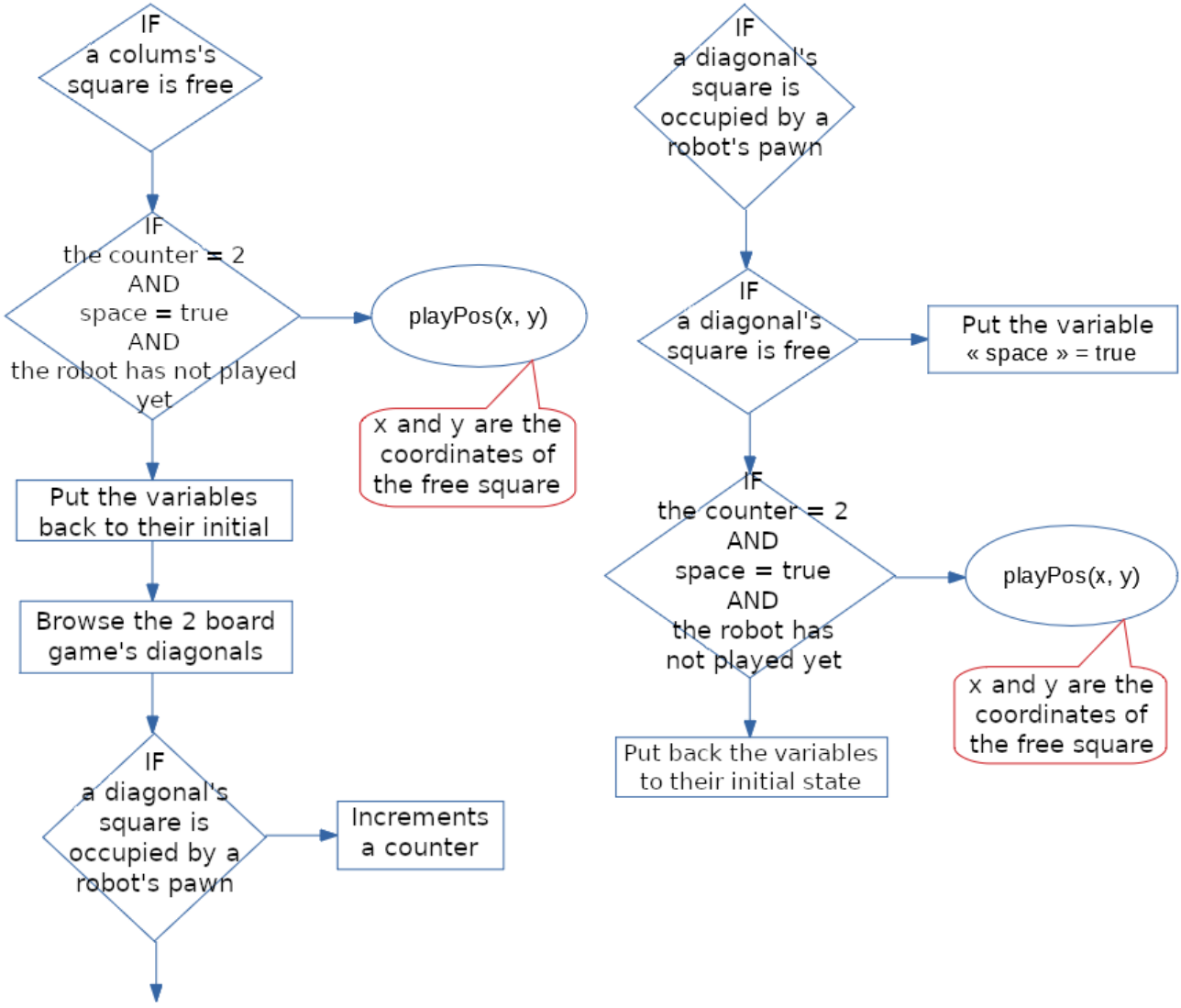
robotsTurn()


Turn on the led indicating the robot's turn
checkPossibilities()
checkBlockers()
randomPlay()
Decrements the number of free square
Decrements the number of free square
printGame()
checkGame()
Turn off the led indicating the robot's turn
initializeRobotBit()

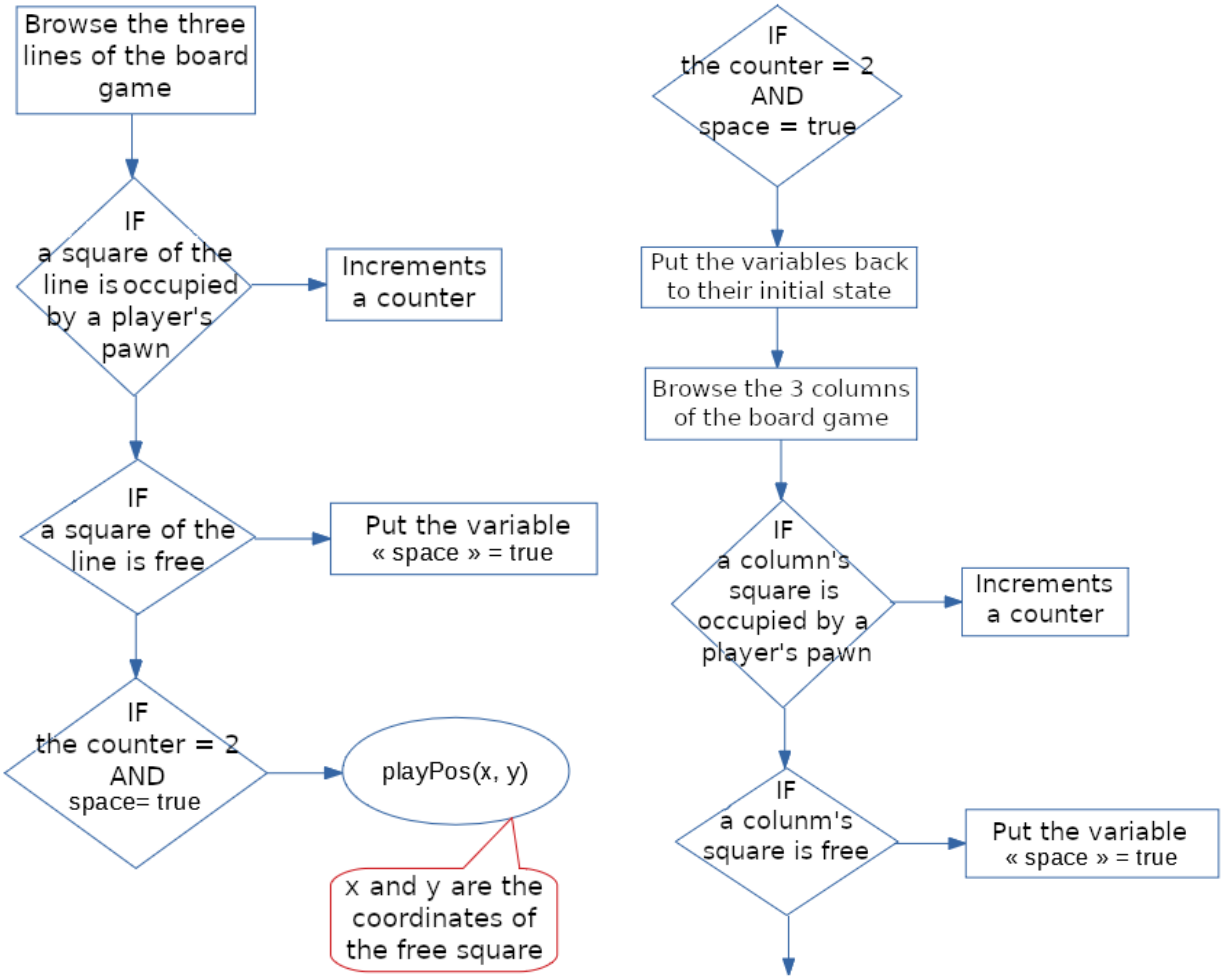
checkPossibilities()

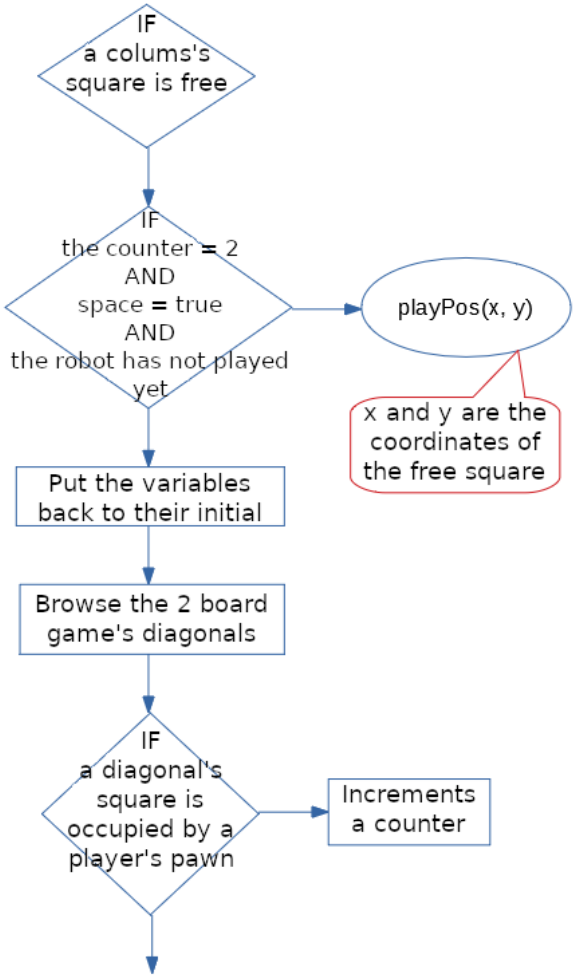


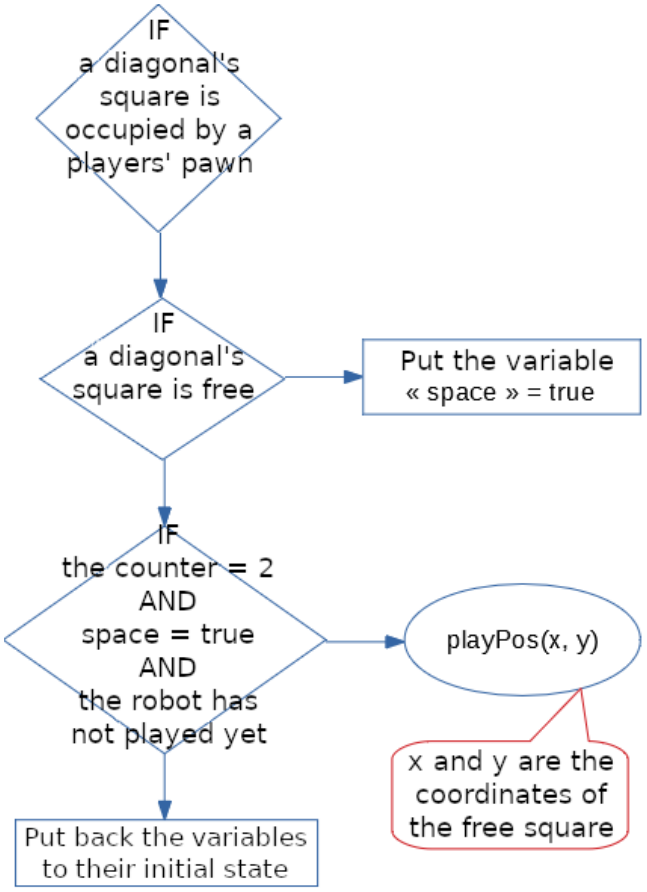



IF
a colums's
square is free
IF
the counter = 2
AND
space = true
AND
the robot has not played
yet
playPos(x, y)
x and y are the
coordinates of
the free square
Put the variables
back to their initial
Browse the 2 board
game's diagonals
IF
a diagonal's
square is
occupied by a
robot's pawn
Increments
a counter
IF
a diagonal's
square is
occupied by a
robot's pawn
IF
a diagonal's
square is free
Put the variable
« space » = true
IF
the counter = 2
AND
space = true
AND
the robot has
not played yet
playPos(x, y)
x and y are the
coordinates of
the free square
Put back the variables
to their initial state

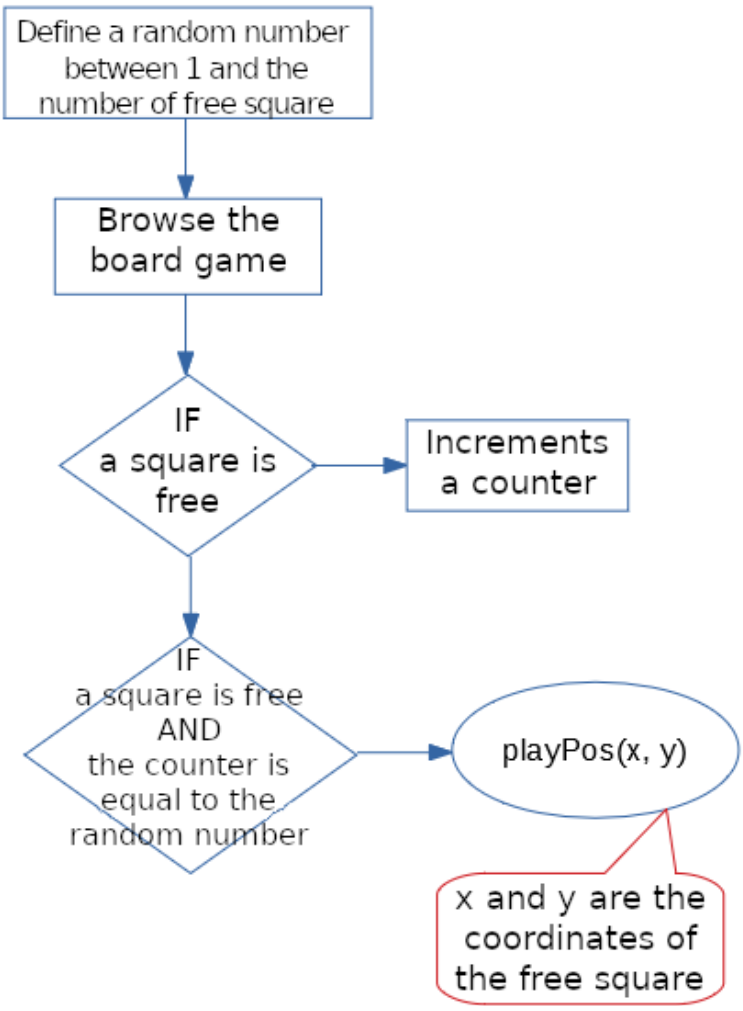
checkBlockers()



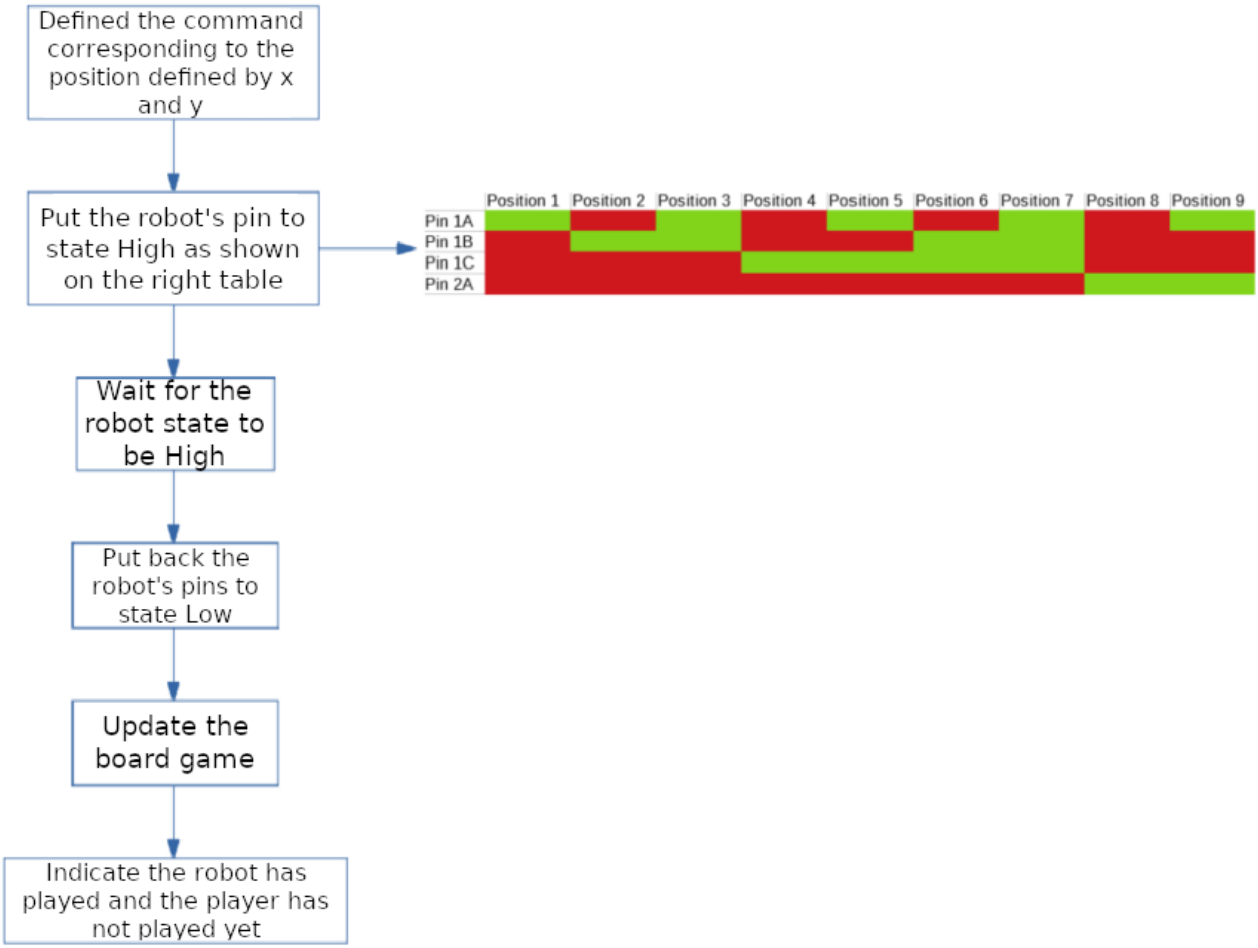

IF
a colums's
square is free
IF
the counter = 2
AND
space = true
AND
the robot has not played
yet
playPos(x, y)
x and y are the
coordinates of
the free square
Put the variables
back to their initial
Browse the 2 board
game's diagonals
IF
a diagonal's
square is
occupied by a
player's pawn
Increments
a counter


IF
a diagonal's
square is
occupied by a
players' pawn
IF
a diagonal's
square is free
Put the variable
« space » = true
IF
the counter = 2
AND
space = true
AND
the robot has
not played yet
playPos(x, y)
x and y are the
coordinates of
the free square
Put back the variables
to their initial state

randomPlay()


Define a random number between 1 and the number of free square
Browse the board game
IF a square is free
Increments a counter
IF a square is free AND the counter is equal to the random number
playPos(x, y)
x and y are the coordinates of the free square

playPos(x,y)



initializeRobotBits()

Put all robot’s pins to state Low.