P.PORTO ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO	Tipo de Prova Model test	Ano letivo XXXX/XXXX	Data XX-XX-XXXX
	Curso Informatics Engineering Computer and Networks Security		Hora XX:XX
	Unidade Curricular Programming Fundamentals		Duração XXhXX

Observations
- Fill in the whole header of the test sheet(s): full name and student number, date of the evaluation test, name of the course unit and course.
- If you want to give up you must write on the "I give up" exam sheet and put your signature under it.
- It is not allowed the use of any documentation other than that indicated or provided by the teacher.
- You must deliver everything that was delivered by the teacher: test sheets, draft sheets, and statement.
- Students must not leave the examination room without signing the attendance sheet.

Consider the 3-in-line game. The implementation of this game will consider a matrix (3x3) of integers. A value of -1 will identify that the position is available to play. During the game, the user will be asked for the row and column where he wants to make his move. When the possible is validated, the player's number 0 or 1 must be assigned to the position. A player wins the game when he has his number in 3 consecutive places (row, column, or diagonal).

	0	1	2
0	-1	-1	-1
1	-1	-1	-1
2	-1	-1	-1

Part 1

1. Implement a function (**`getValue`**) that asks the user for an integer value until the value 0, 1 or 2 is inserted. This function will be used to request the row and column on which the player wants to make his move.
2. Implement a function (**`printMatrix`**) that takes an array of integers as an argument and writes that array to the console.
3. Implement a function (**`checkEmptyPosition`**) that receives an array of integers, an integer value to identify the row and an integer value to identify the column. This function should return 1 if the position specified by the arguments has a value of -1 (indicating a position available to play). It should return 0, if the value is different from -1, that is, it has already been used by a player.
4. Implement a function (**`verifyVitory`**) that receives an array of integers and an integer value that identifies the player, and check if the player made a winning move (in a row, column or diagonal).

Part 2

1. Assume (even if not implemented in part 1 of this test) that the previous functions (**`getValue`**, **`printMatrix`**, **`checkEmptyPosition`** and **`verifyVitory`**) are found in the module (**`myModule.h`**). Implement a program that (in the order presented):
   - 1.1. Initialize the matrix by assigning -1 to all positions.
   - 1.2. Print the matrix (use the **`printMatrix`** function).
   - 1.3. As long as the maximum number of moves has not been reached and no one has won.
     - 1.3.1. Ask the current player for a valid position (use the **`getValue`** and **`checkEmptyPosition`** functions).
     - 1.3.2. Print matrix (use the **`printMatrix`** function).
     - 1.3.3. Checks whether the player has won (use the **`verifyVitory`** function).