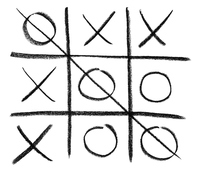
**NAUGHTS AND CROSSES**



**PROGRAMMING FOR BUSINESS**

**BACHELOR’S DEGREE IN MANAGEMENT AND TECHNOLOGY**

**PROFESSOR: FUENSANTA MEDINA DOMINGUEZ**

**Group 13**

**Sin Myeong Eun(NIA: 100433843)**

**Nicole Vásquez Muñoz (NIA: 100381657)**

**TABLE OF CONTENTS**

1.-Introduction………………………………………………………………………….page 2

2.-Description of the data structures used…………...……………………………........page 3

3.-Functions performed………………………………………………………………....page 4   
 3.1.- menu()   
 3.2.- coordinates()  
 3.3.-user1()/user2()  
 3.4.- new\_board(user1\_coordinates,user2\_coordinates)

4.-Example of the operation of the program…………………………………………....page 5

5.- Commented program code……………………………………………………..........page 10

**1.- Introduction**

“Noughts and crosses” is a game for two players, each player with a symbol (X or O), each player has to mark a space, the player who succeeds in placing three of their marks in a horizontal, vertical or diagonal row is the winner.

The first part of this project is the menu, the program will welcome both players to the game and will show a menu with the four options users have at the beginning of the game, the user will have the chance to choose between access to configuration, where users will have to set up the system, to play directly, to see the record where the system will store the player names and their scores in a file or to exit the game.   
To achieve this, we will have to use output and input functions that we’ve learned in class, output functions in order to display to the users the list of choices they have, and input function so users can let us know their choice.

The second part of this project consist on developing the option number 2 which let the users start with the game. First, the system will ask each user for the coordinates where they want to put their symbol, to guarantee that each player uses just one spot of the board the system will have to verify if the coordinate that the user entered is empty and in the board (between 1 and 10 for row and column). If it is, the system will automatically put the symbols in the board, if not, the system will ask to the user to enter a valid coordinate.   
Once each player has entered a valid coordinate to put their symbols the system will display the board with the X’s and O’s entered.   
In this part we will be using a matrix for the board and some other functions to guarantee the efficiency of the game.

**2.-Description of the data structures used**

Data structures in R programming are explained as the elements used for storing multiple types of data.   
Some data structures we’ve seen by far are vector and matrix.

**List of data structures:**

1. **board: matrix**A matrix can hold values of only one data type and it can store and display that data in tabular format, columns and rows.

In order to create the board where users will be placing their symbols, we will define a 10 by 10 matrix, filled with b’s, where “b” means an empty place.   
To create the board we execute the following code:

board = matrix("b",10,10)

1. **user1\_coordinates/ user2\_coordinates: vector**Vectors are used to group together multiple values of the same data type.

These two variables are vector variables which store the coordinates of each user.

**3.-Functions performed**

1. **menu()**

The menu() function displays all 4 options that users can choose from. When this function is called, it prompts the user for the option that he/she would like to choose and checks if it is a valid input. Once it validates the input, it stores the input as a variable “choice” so that the function can return the variable.

1. **coordinates()**

This function prompts the user for row and column number and checks if the user keyed in valid integers. If the user keyed in wrong integers (not within the range of 1~10), the “while” loop will run in order to let the user know that the inputs were wrong and also to prompt the user again for another row/column number. Once the user keys in correct inputs, the row and column numbers are stored into a vector variable called coordinates and the function returns this variable.

1. **user1()/user2()**

These two functions work the same, but for different users. User1() is to store the coordinates returned from coordinates() as user1\_coordinates variable, whereas user2() stores the coordinates returned from coordinates() as user2\_coordinates variable. The purpose is just to store the coordinates returned from coordinates() into a variable.

1. **new\_board(user1\_coordinates, user2\_coordinates)**

This function takes in 2 parameters (user1\_coordinates and user2\_coordinates) and checks if the two users’ coordinates can be put onto the board.

Firstly, it checks user1\_coordinates against the board if the particular square on the board is empty or not by using the if statement as follows. If the square has “b”, it is empty so therefore put “O” on the square. Below is the pseudo code for this part.

# If user1's coordinates on the board is empty, replace it with “O”

If the square is NOT empty, user1 will be prompted to enter other coordinates until he enters the coordinates where the square is empty. Below is the pseudo code:

# Else, keep prompting user1

# Check the new coordinates if the square is empty

# If empty, replace with “O” and break

# Else, repeat again

Secondly, we do the exact same thing for user2\_coordinates(), but replace the square with “X” instead.

# If user2's coordinates on the board is empty, replace it with “X”

# Else, keep prompting user1

# Check the new coordinates if the square is empty

# If empty, replace with “X” and break

# Else, repeat again

**4.-Example of the operation of the program**

> menu <- function(){

+ cat("Welcome to noughts and crosses\n

+ 1: Configuration\n

+ 2: Play\n

+ 3: Record\n

+ 4: Game over\n")

+ choice = as.integer(readline(prompt = "Which option would you like to choose?: "))

+ while(choice>4 || choice<1){

+ print("Please, enter a valid option")

+ choice = as.integer(readline(prompt = "Which option would you like to choose?: "))

+ }

+ return(choice)

+ }

>

> board = matrix("b",10,10)

>

> coordinates <- function(){

+ row = as.integer(readline(prompt="Please, insert row: "))

+ while(row<1 || row>10){

+ row = as.integer(readline(prompt="The row is not correct. Please enter another row: "))

+ }

+ col = as.integer(readline(prompt="Please, insert column: "))

+ while(col<1 || col>10){

+ col = as.integer(readline(prompt="The column is not correct. Please enter another column: "))

+ }

+ coordinates = c(row,col)

+ return(coordinates)

+ }

>

> user1 <- function(){

+ user1\_coordinates = coordinates()

+ return(user1\_coordinates)

+ }

> user2 <- function(){

+ user2\_coordinates = coordinates()

+ return(user2\_coordinates)

+ }

>

>

> new\_board <- function(user1\_coordinates,user2\_coordinates){

+

+ if(board[user1\_coordinates[1],user1\_coordinates[2]] == "b"){

+ board[user1\_coordinates[1],user1\_coordinates[2]] <- "O"

+ }

+

+ else{

+ repeat{

+ cat("User 1, the square is not empty, enter other coordinates.")

+ user1\_coordinates <- user1()

+ if(board[user1\_coordinates[1],user1\_coordinates[2]] == "b"){

+ board[user1\_coordinates[1],user1\_coordinates[2]] <- "O"

+ break

+ }

+ }

+ }

+

+ if(board[user2\_coordinates[1],user2\_coordinates[2]] == "b"){

+ board[user2\_coordinates[1],user2\_coordinates[2]] <- "X"

+ }

+

+ else{

+ repeat{

+ cat("User 2, the square is not empty, enter other coordinates.")

+ user2\_coordinates <- user2()

+ if(board[user2\_coordinates[1],user2\_coordinates[2]] == "b"){

+ board[user2\_coordinates[1],user2\_coordinates[2]] <- "X"

+ break

+ }

+ }

+ }

+

+ return(board)

+ }

>

> if(menu()==2){

+ user1\_coordinates = user1()

+ user2\_coordinates = user2()

+

+ board = new\_board(user1\_coordinates,user2\_coordinates)

+ board

+ }

Welcome to noughts and crosses

1: Configuration

2: Play

3: Record

4: Game over

Which option would you like to choose?: 5

[1] "Please, enter a valid option"

Which option would you like to choose?: 2

Please, insert row: 3

Please, insert column: 4

Please, insert row: 3

Please, insert column: 4

User 2, the square is not empty, enter other coordinates.

Please, insert row: 3

Please, insert column: 6

[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]

[1,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[2,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[3,] "b" "b" "b" "O" "b" "X" "b" "b" "b" "b"

[4,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

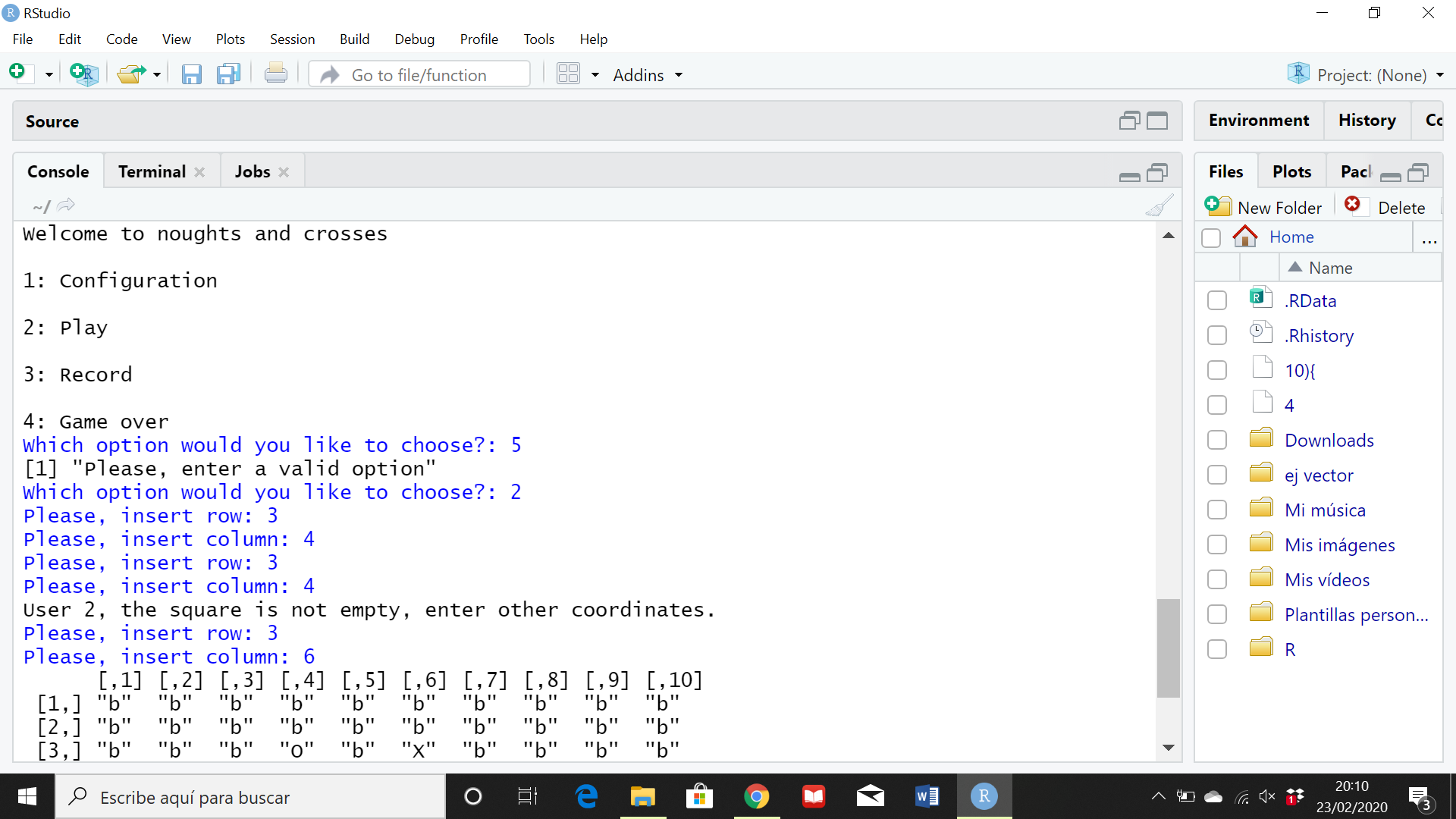
[5,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

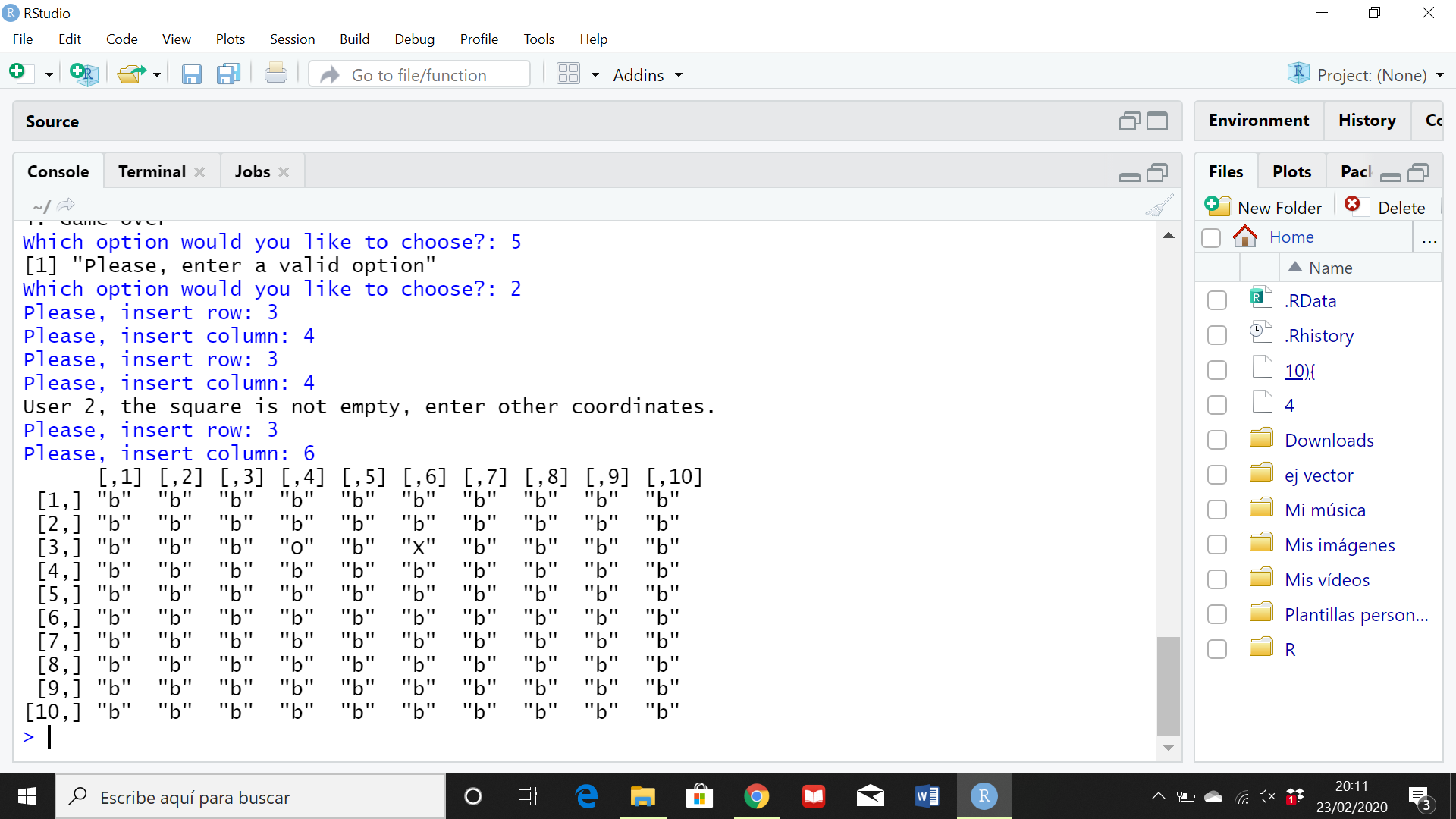
[6,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[7,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[8,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[9,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

[10,] "b" "b" "b" "b" "b" "b" "b" "b" "b" "b"

****

**5.- Commented program code**

# Program shows menu and allows user to choose an option

menu <- function(){

cat("Welcome to noughts and crosses\n

1: Configuration\n

2: Play\n

3: Record\n

4: Game over\n")

choice = as.integer(readline(prompt = "Which option would you like to choose? "))

while(choice>4 || choice<1){

print("Please, enter a valid option")

choice = as.integer(readline(prompt = "Which option would you like to choose? "))

}

return(choice)

}

# Defining of the board

board = matrix("b",10,10)

# Option 2

# This function validates the user's coordinates and returns them in a vector format

coordinates <- function(){

row = as.integer(readline(prompt="Please, insert row: "))

while(row<1 || row>10){

row = as.integer(readline(prompt="The row is not correct. Please enter another row: "))

}

col = as.integer(readline(prompt="Please, insert column: "))

while(col<1 || col>10){

col = as.integer(readline(prompt="The column is not correct. Please enter another column: "))

}

coordinates = c(row,col)

return(coordinates)

}

# For storing each user's coordinates in a variable

user1 <- function(){

user1\_coordinates = coordinates()

return(user1\_coordinates)

}

user2 <- function(){

user2\_coordinates = coordinates()

return(user2\_coordinates)

}

# Function to check user1's coordinates on the board

new\_board <- function(user1\_coordinates,user2\_coordinates){

# If user1's coordinates on the board is empty, replace it with appropriate symbol

if(board[user1\_coordinates[1],user1\_coordinates[2]] == "b"){

board[user1\_coordinates[1],user1\_coordinates[2]] <- "O"

}

# If the coordinates entered are not empty, keep prompting user1

else{

repeat{

cat("User 1, the square is not empty, enter other coordinates.")

user1\_coordinates <- user1()

if(board[user1\_coordinates[1],user1\_coordinates[2]] == "b"){

board[user1\_coordinates[1],user1\_coordinates[2]] <- "O"

break

}

}

}

# If user2's coordinates on the board is empty, replace it with appropriate symbol

if(board[user2\_coordinates[1],user2\_coordinates[2]] == "b"){

board[user2\_coordinates[1],user2\_coordinates[2]] <- "X"

}

# If not empty, keep prompting user2

else{

repeat{

cat("User 2, the square is not empty, enter other coordinates.")

user2\_coordinates <- user2()

if(board[user2\_coordinates[1],user2\_coordinates[2]] == "b"){

board[user2\_coordinates[1],user2\_coordinates[2]] <- "X"

break

}

}

}

return(board)

}

if(menu()==2){

user1\_coordinates = user1()

user2\_coordinates = user2()

# Re-assign board to the new returned board from the function

board = new\_board(user1\_coordinates,user2\_coordinates)

board

}