Board Array= {

{ ‘ ‘ , ‘ ‘ , ‘ ‘ },

{ ‘ ‘ , ‘ ‘ , ‘ ‘ },

{ ‘ ‘ , ‘ ‘ , ‘ ‘ },

};

Position= {

a { ‘1‘ , ‘2‘ , ‘3‘ },

b { ‘4‘ , ‘5‘ , ‘6‘ },

c { ‘7‘ , ‘8‘ , ‘9‘ },

      a    b     c

};

Let players = [ ‘X’ , ’O’ ];

Board Cell = [ “empty”, “X”, “O” ]

Player1 : “X” [pos 5]

Player2 : “O” [pos = random.available]

Checkwinner () {

{

String a, string b, string c in row or column;

return (a == b && b == c && a!= “empty”)

winner = player on string

}

{

If (winner == null && available.cell () == 0)

return“tie”

} else {

 return winner;

}

}

nextTurn () {

random available cell ();

#before all next turns,

Checkalways ();

pos currentPlayer2;

if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)

pos currentPlayer1;

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

}

# Startgame () ; {

“X” = [pos 5]

nextTurn (); {

for (“O” playing in side positions) {

{ if ( “O” == (pos [2] )

then nextTurn Checkalways # “X” play in cell next to pos [2]

(“X” == (pos [1]); or (“X” == pos [3])

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete diagonal line

(“O” == (pos [9]); or (“O” == pos [7]) respectively

then nextTurn Checkalways # “X” plays on last cell in the line of “O”’s last play (forming an empty triangle with previous “X” plays)

(empty triangle mean “O” must not be present within the drawn triangle when a line that connects all present “X”s are drawn)

(“X” == (pos [7]); or (“X” == pos [9]) respectively

#Recall from above

[if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)]

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete line

(“O” == (pos [3]) or (pos [4])); or (“O” == (pos [1]) or (pos [5])) respectively

#Recall from above

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

then nextTurn Checkalways # “X” completes line to win

}

{ if ( “O” == (pos [4] )

then nextTurn Checkalways # “X” play in cell next to pos [4]

(“X” == (pos [1]); or (“X” == pos [7])

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete diagonal line

(“O” == (pos [9]); or (“O” == pos [3]) respectively

then nextTurn Checkalways # “X” plays on last cell in the line of “O”’s last play (forming an empty triangle with previous “X” plays)

(empty triangle mean “O” must not be present within the drawn triangle when a line that connects all present “X”s are drawn)

(“X” == (pos [3]); or (“X” == pos [9]) respectively

#Recall from above

[if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)]

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete line

(“O” == (pos [2]) or (pos [7])); or (“O” == (pos [1]) or (pos [8])) respectively

#Recall from above

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

then nextTurn Checkalways # “X” completes line to win

}

{ if ( “O” == (pos [6] )

then nextTurn Checkalways # “X” play in cell next to pos [6]

(“X” == (pos [3]); or (“X” == pos [9])

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete diagonal line

(“O” == (pos [7]); or (“O” == pos [1]) respectively

then nextTurn Checkalways # “X” plays on last cell in the line of “O”’s last play (forming an empty triangle with previous “X” plays)

(empty triangle mean “O” must not be present within the drawn triangle when a line that connects all present “X”s are drawn)

(“X” == (pos [1]); or (“X” == pos [7]) respectively

#Recall from above

[if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)]

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete line

(“O” == (pos [2]) or (pos [9])); or (“O” == (pos [3]) or (pos [8])) respectively

#Recall from above

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

then nextTurn Checkalways # “X” completes line to win

}

{ if ( “O” == (pos [8] )

then nextTurn Checkalways # “X” play in cell next to pos [8]

(“X” == (pos [7]); or (“X” == pos [9])

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete diagonal line

(“O” == (pos [3]); or (“O” == pos [1]) respectively

then nextTurn Checkalways # “X” plays on last cell in the line of “O”’s last play (forming an empty triangle with previous “X” plays)

(empty triangle mean “O” must not be present within the drawn triangle when a line that connects all present “X”s are drawn)

(“X” == (pos [1]); or (“X” == pos [3]) respectively

#Recall from above

[if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)]

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete line

(“O” == (pos [4]) or (pos [9])); or (“O” == (pos [7]) or (pos [6])) respectively

#Recall from above

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

then nextTurn Checkalways # “X” completes line to win

}

}

for (“O” playing in corner positions) {

{ if ( “O” == (pos [1]) or ( “O” == (pos [3]) or ( “O” == (pos [7]) or ( “O” == (pos [9])

then nextTurn Checkalways # “X” play in cell that completes the diagonal

(“X” == pos [9]) or (“X” == pos [7]) or (“X” == pos [3]) or (“X” == pos [1]) respectively

==> then nextTurn Checkalways #if “O” plays in one of the side cells

if (“O” == (pos [2]) or (“O” == (pos [4]) or (“O” == (pos [6]) or (“O” == (pos [8])

then nextTurn Checkalways # “X” plays to form empty triangle with previous “X” plays

(empty triangle mean “O” must not be present within the drawn triangle when a line that connects all present “X”s are drawn

(“X” == available.add)

#Recall from above

[if row or column or diagonal = “X”, “X”, “empty”

for true (available.add = “O” to replace “empty” in that row or column)

for false (available.add = “O” to replace “empty” in any row or column)]

then nextTurn Checkalways # “O” to prevent “X” from winning plays to complete line

(“O” == available.add)

#Recall from above

if row or column or diagonal = “O”, “O”, “empty”

for true (available.add = “X” to replace “empty” in that row or column)

or false (available.add = “X” to replace “empty” in any row or column)

then nextTurn Checkalways # “X” completes line to win

==> then nextTurn Checkalways #if “O” plays in one of the corner cells

if (“O” == (pos [1]) or (“O” == (pos [3]) or (“O” == (pos [7]) or (“O” == (pos [9])

then nextTurn Checkalways

Both players continue to take turns until all cells are filled.

“X” and “O” tie.

}

// horizontal {

If ((pos [1] != “empty”)

&&

(pos [1] = = pos [2])

&&

(pos [2] = = pos [3]

Print win

If ((pos [4] != “empty”)

&&

(pos [4] = = pos [5])

&&

(pos [5] = = pos [6]

Print win

If ((pos [7] != “empty”)

&&

(pos [7] = = pos [8])

&&

(pos [8] = = pos [9]

Print win

}

// vertical {

If ((pos [1] != “empty”)

&&

(pos [1] = = pos [4])

&&

(pos [4] = = pos [7]

Print win

If ((pos [2] != “empty”)

&&

(pos [2] = = pos [5])

&&

(pos [5] = = pos [8]

Print win

If ((pos [3] != “empty”)

&&

(pos [3] = = pos [6])

&&

(pos [6] = = pos [9]

Print win

}

// diagonal {

If ((pos [1] != “empty”)

&&

(pos [1] = = pos [5])

&&

(pos [5] = = pos [9]

Print win

If ((pos [3] != “empty”)

&&

(pos [3] = = pos [5])

&&

(pos [5] = = pos [7]

Print win

}

checkWinner ();

 If (result != null) {

noLoop ();

If (result == “tie”) {

print (“Tie!”);

} else  {

print (result + “wins!”) ;

} else {

nextTurn ();

}

}