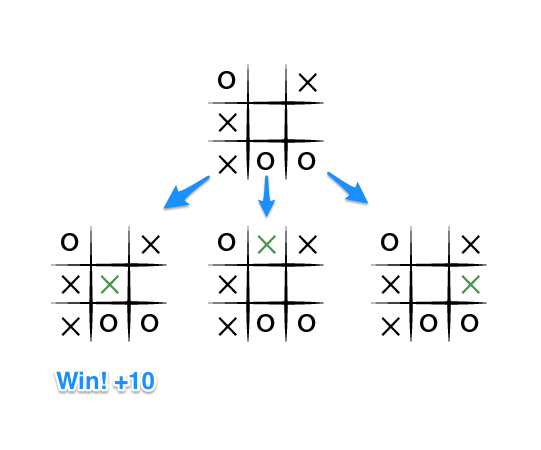
**The laboratory work 9**

**The tic-tac-toe game realization 2**

**Looking at a Brief Example**

To apply this, let's take an example from near the end of a game, where it is my turn. I am X. My goal here, obviously, is to *maximize* my end game score.



If the top of this image represents the state of the game I see when it is my turn, then I have some choices to make, there are three places I can play, one of which clearly results in me wining and earning the 10 points. If I don't make that move, O could very easily win. And I don't want O to win, so my goal here, as the first player, should be to pick the maximum scoring move.

At this step we need to build the template of the game. **We play with a computer, but we still need to add THE MACHINE MOVE using MINIMAX function**.

The code for building the whole board and making steps in the game

|  |
| --- |
| #include "stdafx.h"  #include <iostream>  #include <iterator>  #include <string>  #include "MiniMax.h"  using namespace std;  void print\_board(int board[9])  {  int i, j;  for (i = 0; i < 3; i++)  {  cout << " ";  for (j = 0; j < 3; j++)  {  if (board[i \* 3 + j] == 1)  cout << " X ";  else if (board[i \* 3 + j] == -1)  cout << " O ";  else if (board[i \* 3 + j] == 0)  cout << " ";  if (j != 2)  cout << " | ";  }  if (i != 2)  cout << "\n----------------" << endl;  else  cout << " ";  }  cout << "\n" << endl;  }  void print\_instructions()  {  cout << "\nThe board is laid out as folows:\n";  cout << " 1 | 2 | 3" << endl;  cout << "-----------------" << endl;  cout << " 4 | 5 | 6" << endl;  cout << "-----------------" << endl;  cout << " 7 | 8 | 9" << endl;  }  char select\_starter()  {  cout << "\nWeclome to Tic Tac Toe" << endl;  cout << "Would you like to be the 'X' or the 'O'?" << endl;  bool valid\_user = false;  char user = ' ';  while (!valid\_user)  {  cout << "Type X or O: ";  cin >> user;  if (user == 'X' || user == 'O')  {  valid\_user = true;  cout << "Great. You are " << user << endl;  if (user == 'X')  cout << "You will go first" << endl;  else  {  cout << "The machine will go first" << endl;  cout << "Good luck";  }  }  }  return user;  }  int get\_input(int board[9], char turn)  {  bool valid = false;  int response;  int move;  int arr[9] = { 1,2,3,4,5,6,7,8,9 };  while (!valid)  {  cout << "\nWhere would you like to make your move (You are "<< turn << ")? Pick 1-9" << endl;  cin >> response;  bool exists = find(begin(arr), end(arr), response);  if (exists)  {  move = response - 1;  if (board[move] == 0)  return move;  else  {  cout << "That position has already been taken " << endl;  }  }  else  cout << "That is not a valid move. Try again!" << endl;  }  }  int machine\_move(int board[9], char my\_symbol)  {  bool valid\_move = false;  int move;  int arr[9] = { 0, 1, 2, 3, 4, 5, 6, 7, 8 };  while (!valid\_move)  {  move = mymove(board, my\_symbol);  bool exists = find(begin(arr), end(arr), move);  if (exists)  {  if (board[move] == 0)  return move;  else  {  cout << "That position has already been taken " << endl;  return 0;  }  }  else  cout << "That is not a valid move! Please try again! " << endl;  }  }  string check\_win(int board[9])  {  int i, j;  int threes[8][3] = { {1,2,3},{4,5,6},{7,8,9},{1,4,7},{2,5,8},{3,6,9},{1,5,9},{3,5,7} };  int total = 0;  for (i = 0; i < 8; i++)  {  total = 0;  for (j = 0; j < 3; j++)  {  total += board[threes[i][j] - 1];  }  if (total == -3)  return "O";  else if (total == 3)  return "X";  }  return "No winner";  }  int get\_move(int board[9], char turn, char user)  {  if (turn == user)  return get\_input(board, turn);  else  return machine\_move(board, turn);  }  int main()  {  int board[9] = { 0,0,0,0,0,0,0,0,0 };  print\_instructions();  char user\_symbol = select\_starter();  bool win = false;  int move = 0;  char turn;  int target;  int turn\_value;  string winner;  while (!win)  {  move = move + 1;  cout << "\nThe state of the game is as follows:" << endl;  print\_board(board);  if (move % 2 == 1)  {  turn = 'X';  target = 1;  }  else  {  turn = 'O';  target = -1;  }  turn\_value = get\_move(board, turn, user\_symbol);  board[turn\_value] = target;  cout << "\nChecking for winner ";  winner = check\_win(board);  if (winner != "No winner")  {  cout << "The winner is " << winner << endl;  print\_board(board);  win = true;  }  else if (move >= 9)  {  cout << "The game is a draw" << endl;  print\_board(board);  win = true;  }  else {  cout << "No winner yet" << endl;  }  }  return 0;  } |