Лабораторная работа № 1

«Алгоритм поиска оптимального решения в игре крестики-нолики».

1. Цель

Целью данной работы является изучение одного из алгоритмов поиска оптимального решения в теории игр — алгоритма поиска по минимаксу. Результатом выполнения работы должна стать программа, реализующая игру крестики — нолики с компьютером.

2. Теоретический материал

2.1 Правила игры

Рассмотрим правила данной игры. Действие игры происходит между двумя оппонентами на поле размером 3x3 клетки. Один оппонент использует «крестик» - «X», для хода в клетку игрового поля, а другой «нолик» - «0». Ходы выполняются поочередно в любую свободную клетку игрового поля. Пример игрового поля с тремя совершенными ходами представлен на рисунке 1.

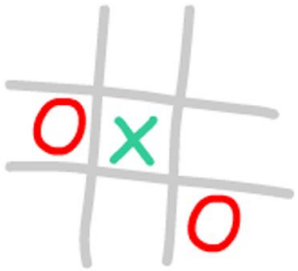

Рисунок 1 — Игровое поле.

Противник, собравший в горизонтальный, вертикальный или диагональный ряд 3 своих фигуры (крестик или нолик) заканчивает игру победой. На рисунке 2 представлена победа игрока, играющим фигурой «крестик».

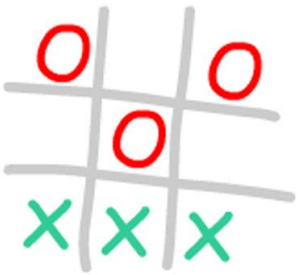

Рисунок 2 — Победа игрока «X»

Реализация этой игры на любом языке программирования для двух реальных игроков задача не сложная. Что если вместо одного из игроков ходы будет совершать компьютер? Как тогда описать его логику игры?

2.2 Алгоритм «минимакс»

Профессиональный шахматист продумывает свою игру на множество ходов вперед, анализируя положения фигур на доске. В нашем случае подход будет аналогичным: алгоритм будет просчитывать ходы вперед до тех пор пока не будет достигнута или победа или поражение или ничья. Назовем эти события — терминальным состоянием. Попав в терминальное состояние компьютер начисляет очки: за победу начисляется 10 очков, за поражение -10 и за ничью 0. Вместе с этим, аналогичные расчеты производятся для ходов игрока: компьютер будет выбирать ход с наибольшим счетом, если ходит он сам или ход с наименьшим счетом если ход выполняет человек. Такое правило называется минимаксом.

Давайте определимся с некоторыми терминами:

- Терминальное состояние — победа одного из игроков или ничья
- Игровое поле — область 3x3 в которую игроки совершают ходы
- Индекс клетки — используется для адресации конкретной клетки. Адресация начинается с 0 и заканчивается 8. Индексация показана на рисунке 1.

0	1	2
3	4	5
6	7	8

Рисунок 3 — Индексация клеток

Кратко алгоритм «минимакса» можно описать как рекурсивную функцию следующего содержания:

1. Функция возвращает счет, если найдено терминальное состояние (победа — 10, проигрыш -10, ничья — 0 очков).

2. Если терминального состояния не найдено, функция проходит по всем свободным клеткам поля, делает в них ходы и рекурсивно вызывает функцию «минимакс» от имени оппонента для каждого своего хода.

3. Функция оценивает наилучший счет для текущего игрока на данном этапе и возвращает этот счет. Наилучшим счетом для игрока-человека является счёт -10, для игрока компьютера — 10.

Давайте подробно рассмотрим алгоритм «минимакс» в действии на примере, изображённом на рисунке 4.

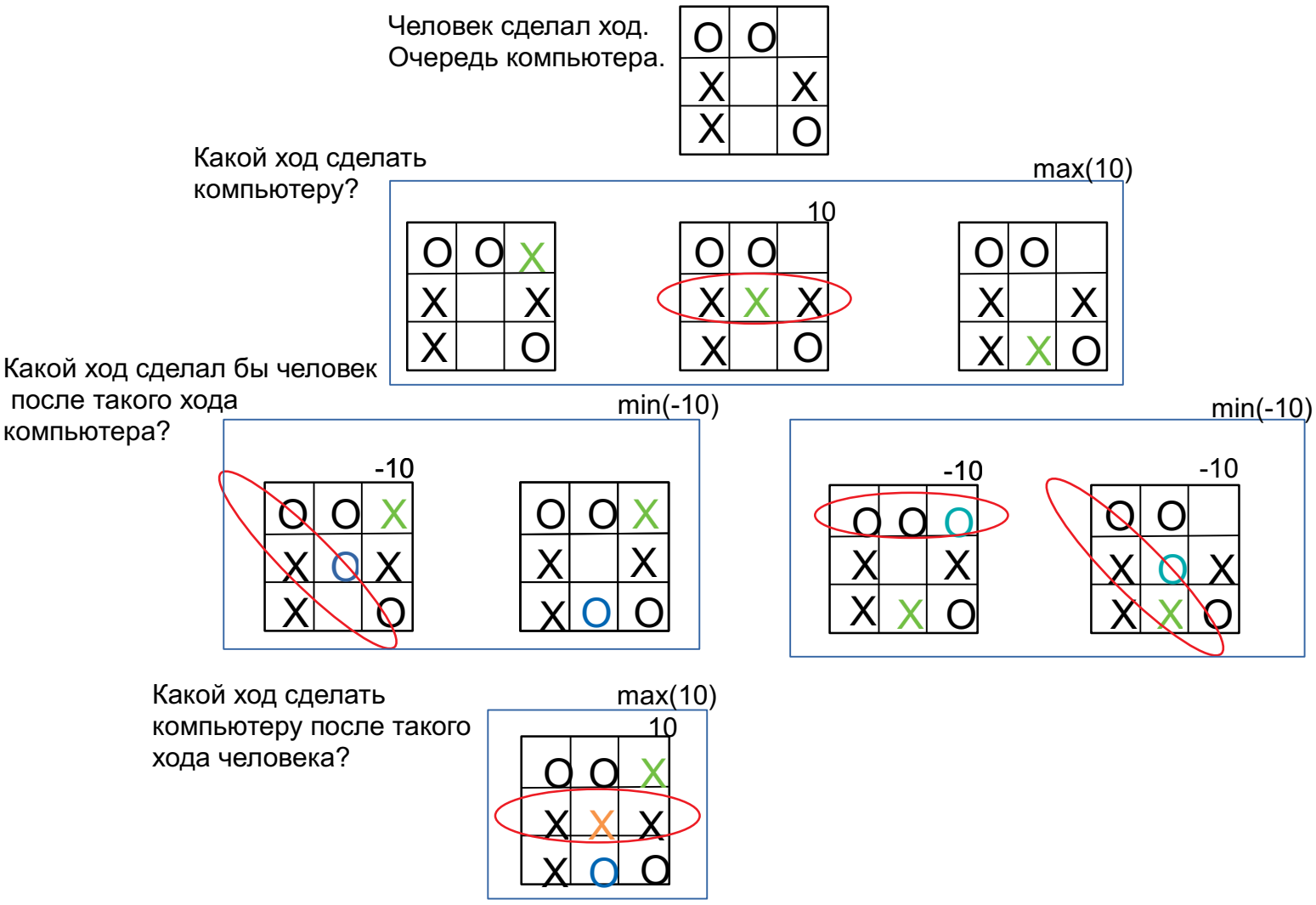

Рисунок 4 Определение оптимального хода компьютера.

Человек сделал свой ход, теперь очередь компьютера. Компьютер поочерёдно перебирает все пустые клетки, и делает в них ходы. Это можно видеть на втором слое дерева. Нужно определить самый оптимальный. Для компьютера самым оптимальным будет ход, со счётом 10. Но не все ходы заканчиваются терминальным состоянием, в этом случае, компьютер начинает перебирать варианты, как бы мог сходить человек. Это можно видеть на третьем слое дерева. Из всех возможных вариантов ходов человека нужно выбрать ход со счётом -10, так как мы предполагаем, что человек думает, куда делает ходы. Если и на этом слое не достигнуто терминального состояния, компьютер перебирает варианты, куда бы он мог сходить после такого хода человека и т.д. пока не будет достигнуто терминальное состояние. Программно это можно представить следующим образом:

Главный цикл(бесконечный){
Спрашиваем у человека ход.
Проверяем, есть ли ничья или победа человека.
Если да, выходим из цикла.
Если такого нет, определяем ход компьютера.
Цикл(перебираем все клетки){
Если клетка пуста{
Делаем в неё ход X (ими играет компьютер)
Запоминаем индекс этой клетки
Вызываем функцию МИНИМАКС(игровое поле, следующий игрок)//возвращает счёт.
Убираем X из клетки.
Если счёт, который вернул МИНИМАКС больше, чем был до этого, запоминаем его, а также индекс этой клетки.
}
}
Делаем ход X в клетку, с найденным индексом.
Проверяем на победу компьютера или ничью.
Если да, выходим из цикла.
Если такого нет, возвращаемся в начало главного цикла.
}

Функция минимакс выглядит следующим образом:

МИНИМАКС(игровое поле, игрок){
Проверка на достижение терминального состояния.
Если такое состояние достигнуто, возвращаем 10 если победил компьютер, -10, если победил человек, 0 — если ничья.
Если терминального состояния не достигнуто, делаем следующее:
Если игрок компьютер{
Цикл(перебираем все клетки){
Если клетка пуста{
Делаем в неё ход X
Вызываем функцию МИНИМАКС(игровое поле, следующий игрок человек)//возвращает счёт.
Убираем X из клетки.
Счёт, который вернул минимакс, запоминаем.
}
}
Возвращаем наибольший счёт.

```
    }

    Если игрок человек{
        Цикл(перебираем все клетки){
         Если клетка пуста{
            Делаем в неё ход О.
            Вызываем функцию МИНИМАКС(игровое поле, следующий игрок
компьютер)//возвращает счёт.
            Убираем О из клетки.
            Счёт, который вернул минимакс, запоминаем.
         }
     }
    Возвращаем наименьший счёт.
    }
}
```

Задание: Реализовать игру крестики нолики с компьютером, не используя никаких готовых реализаций на языке С.