Содержание

[ОБЩАЯ ИНФОРМАЦИЯ 2](#_heading=h.gjdgxs)

[Тема занятия](#_heading=h.30j0zll) 2

[Цели и задачи 2](#_heading=h.1fob9te)

[Ожидаемый результат 2](#_heading=h.3znysh7)

[Структура занятия 3](#_heading=h.tyjcwt)

[ОПИСАНИЕ ЗАНЯТИЯ 4](#_heading=h.3dy6vkm)

# ОБЩАЯ ИНФОРМАЦИЯ

### Тема занятия

### Подготовка к проекту. Tic-Tac-Toe-1.

**Цели и задачи:**

* Научиться использовать полученные знания синтаксиса и основ программирования для создания проекта-игры

**По результатам занятия слушатель будет знать:**

* Как лучше подступить к реализации собственных проектов
* Что такое константная ссылка на объект-вектор

**По результатам занятия слушатель будет уметь:**

* Работать с константными ссылками на объект-вектор
* Понимать и разбираться в выборе структур данных для проектов

### Структура занятия

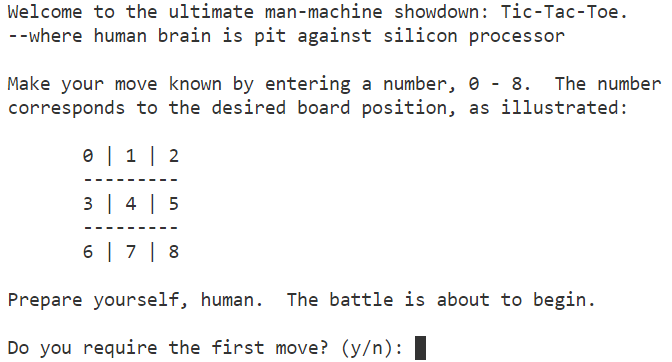
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тайминг занятия** | |  |  |  | Таблица 1 |
| № | Этапы | Что делает преподаватель | Что делает Слушатель | Время | Общее время |
| 1 | Приветственное слово преподавателя |  |  | 5 мин. | 5 мин. |
| 2 | Повторение пройденного |  |  | 5 мин. | 40 мин. |
| 3 | Теоретическая часть |  |  | 30 мин. |
| 4 | Вопросы по теоретической части |  |  | 5 мин. |
|  | *Перерыв* |  |  | 15 мин. | 15 мин. |
| 5 | Практическая часть |  |  | 40 мин. | 45 мин. |
| 6 | Рефлексия и вопросы |  |  | 5 мин. |

# ОПИСАНИЕ ЗАНЯТИЯ

**Теоретическая часть**

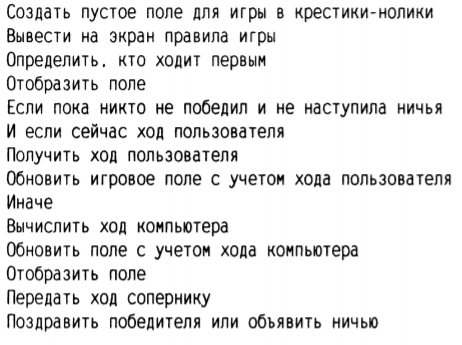
**Знакомство с игрой «Крестики-нолики»**

В проекте к этой главе вы узнаете, как создавать компьютерного соперника. Для этого мы используем толику искусственного интеллекта. В игре пользователь и компьютер, человек и машина, померяются силами в бескомпромиссном поединке в ~Крестики-нолики~. Компьютер играет превосходно (хотя и неидеально) и может действовать достаточно изобретательно для того, чтобы любая партия получилась интересной.

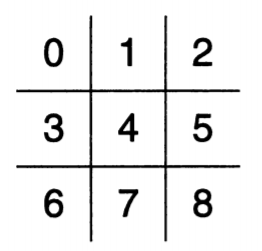
****

**Практическая часть**

Написание псевдокода Возвращаемся к нашему любимому языку, который, строго говоря, языком называется условно. Это псевдокод. Поскольку большинство задач в программе будет решаться с помощью функций, я могу без опаски подготовить довольно абстрактную модель кода. Каждая строка псевдокода условно соответствует одному вызову функции. Затем мне останется всего лишь написать функции, предусмотренные в плане.

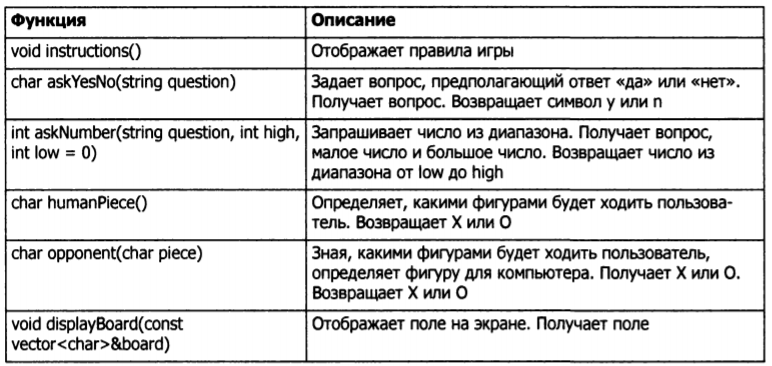
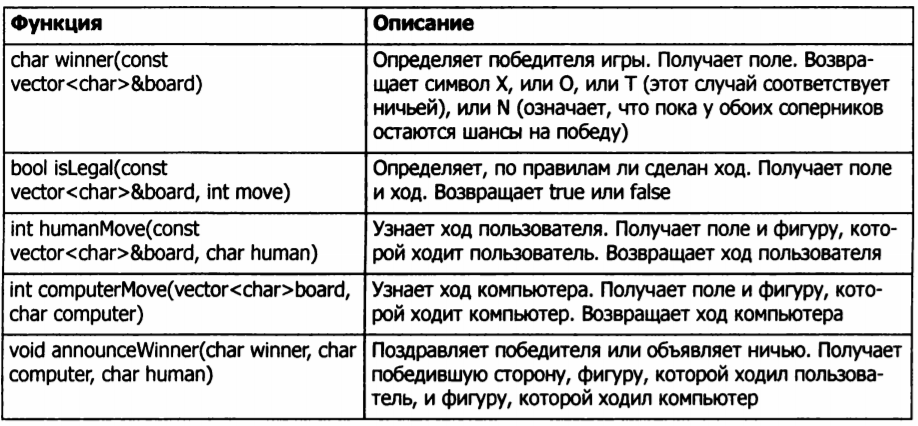


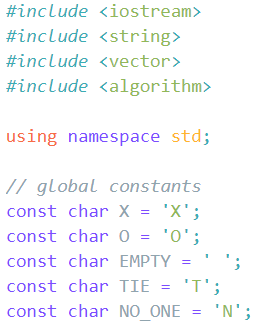
Поскольку я собираюсь отображать игровое поле на экране, почему бы не представить крестик как символ “Х”, а нолик - как символ “О”? В качестве пустой клетки можно просто оставить пробел. Следовательно, само игровое поле может представлять собой вектор, содержащий экземпляры типа char. На игровом поле “Крестиков-ноликов” девять клеток, соответственно, наш вектор должен содержать девять элементов.

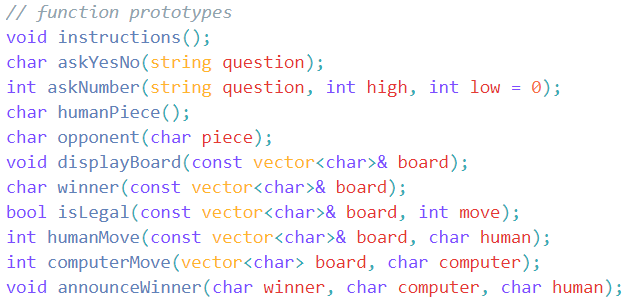


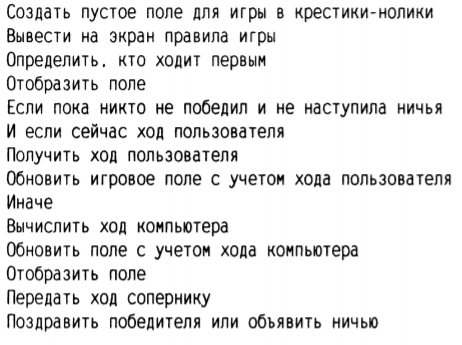
Каждая клетка (позиция) на поле представлена числом от О до 8. Это означает, что в векторе будет девять элементов (позиций), пронумерованных от О до 8. Поскольку при каждом ходе указывается клетка, в которой должен быть поставлен символ, ходы также будут пронумерованы от О до 8. Таким образом, ход должен быть представлен как целое число ( int ).

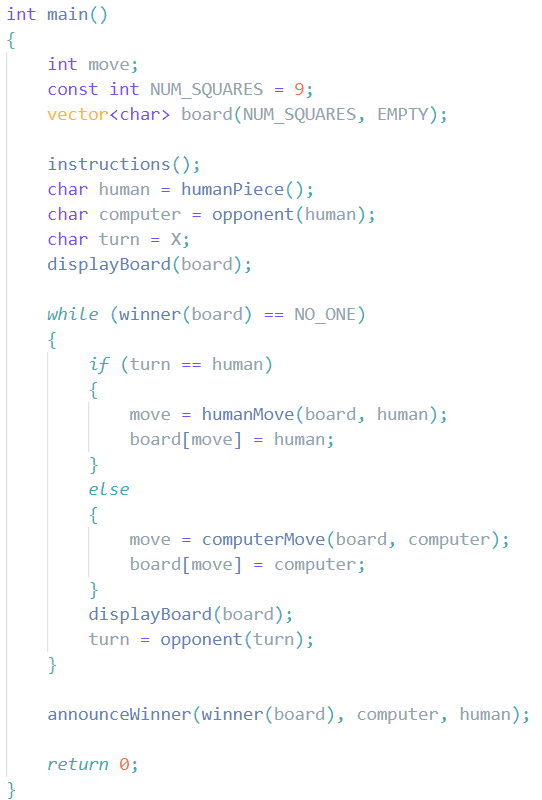
Та сторона, которую занимает пользователь, и та, которую занимает компьютер, должны быть представлены символом char Х или О, точно так же, как фигуры, которыми ходит тот или иной соперник. Переменная, определяющая, чья очередь сейчас делать ход, также будет представлять собой символ char Х или О.

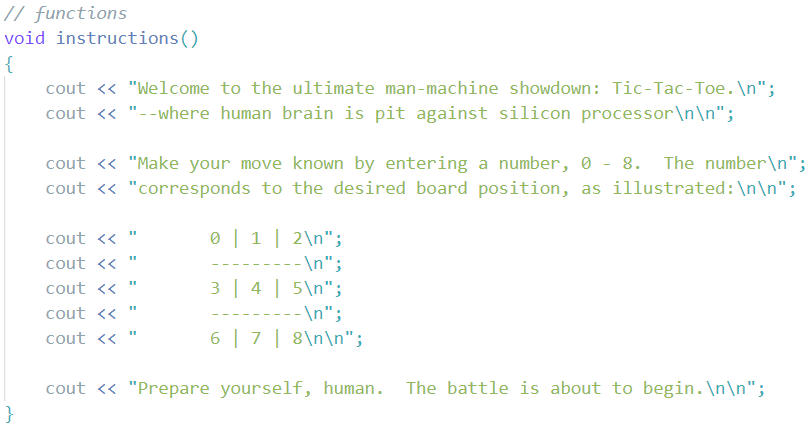






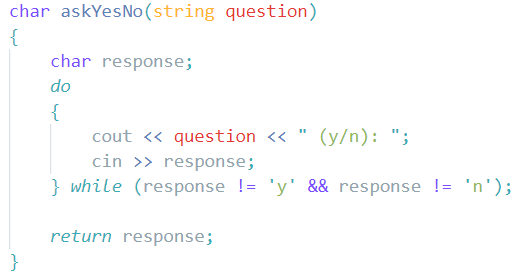






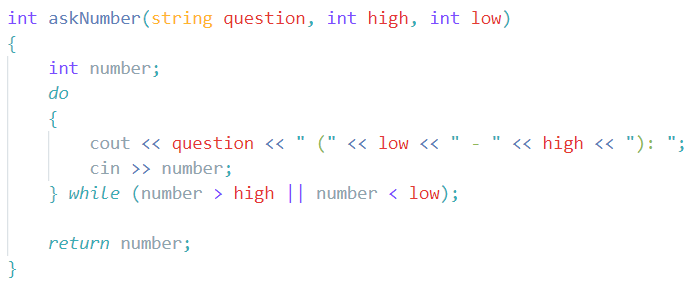
**Функция askYesNo()**

Эта функция задает пользователю вопрос, на который можно ответить “да” или “нет”. Программа продолжает задавать этот вопрос, пока пользователь не введет символ у или n:



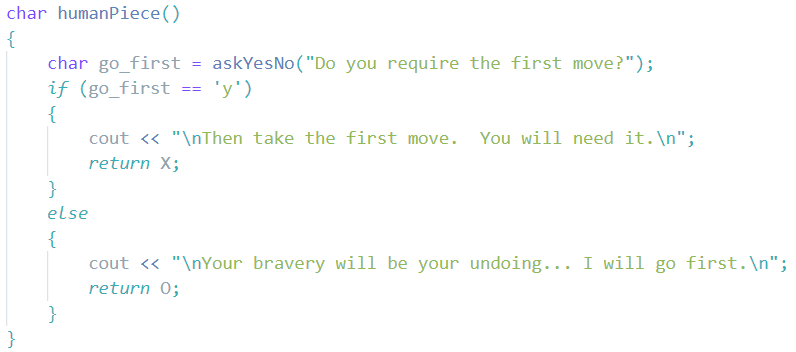
**Функция askNumber()**

Эта функция запрашивает у пользователя число из определенного диапазона и продолжает задавать этот вопрос, пока пользователь не введет удовлетворяющее запросу число. Она принимает вопрос, максимальное число, минимальное число, а возвращает число из указанного диапазона:



**Функция humanPiece()**

Эта функция спрашивает пользователя, хочет ли он пойти первым, и в зависимости от сделанного выбора возвращает ту фигуру, которой будет ходить пользователь. По традиции крестики ходят первыми:



**Функция opponent()**

Эта функция узнает фигуру пользователя (Х или О) и на основании этой информации возвращает фигуру, которой будет ходить соперник-компьютер (Х или О):

