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-3.

**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:**

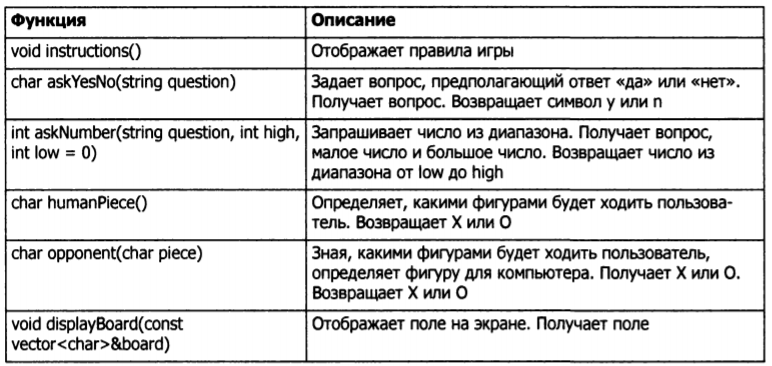
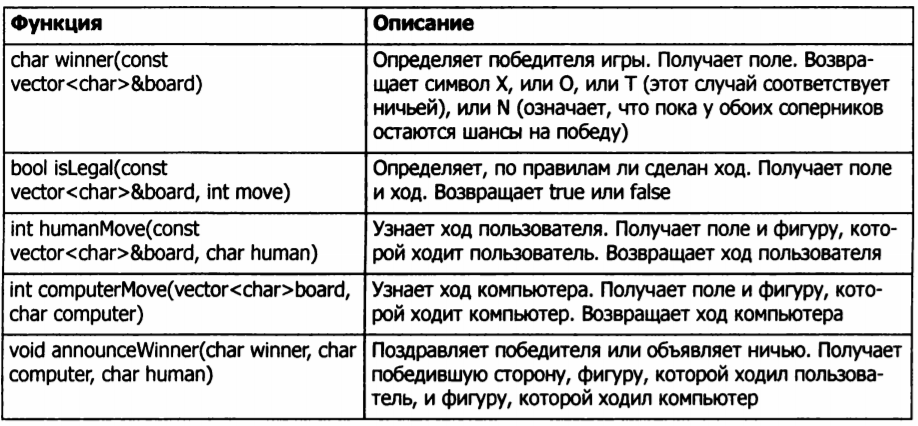
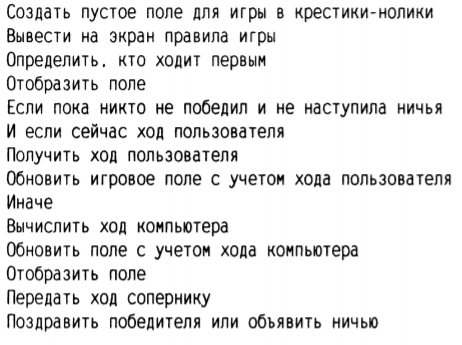
* Работать с константными ссылками на объект-вектор
* Понимать и разбираться в выборе структур данных для проектов

### 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. |

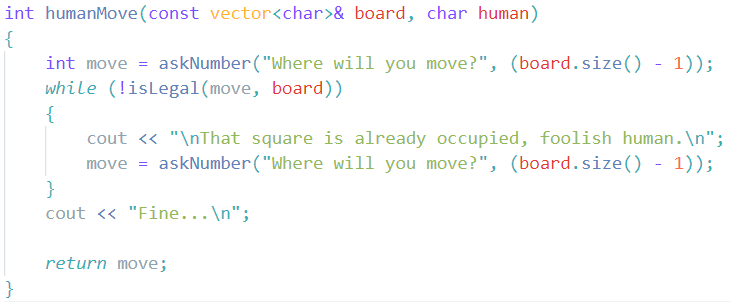
# ОПИСАНИЕ ЗАНЯТИЯ

**Практическая часть Практическая часть**



**Функция humanMove()**

Эта функция получает игровое поле и ту фигуру, которой ходит пользователь. Она возвращает номер той клетки, в которую пользователь хочет поставить свой символ. Функция запрашивает у пользователя номер клетки, в которую он хочет поставить символ, и ждет, пока пользователь не походит по правилам. Затем функция возвращает сделанный ход:



**Функция computerMove()**

Функция получает игровое поле и ту фигуру, которой ходит компьютер. Она возвращает ход компьютера. Первым делом отмечу, что здесь я не передаю поле по ссылке:



Вместо этого я решаю передать поле по значению, пусть этот способ и менее эффективен, чем передача по ссылке. Я передаю поле по значению, так как далее мне придется с ним работать и изменять копию поля, пока я буду пробовать различные варианты хода и определять за компьютер оптимальный вариант. Работая с копией, я оставляю исходный вектор нетронутым - оригинал поля по-прежнему защищен от изменений.

Далее рассмотрим внутренний механизм этой функции. Как реализовать в программе немного искусственного интеллекта, чтобы она была интересным соперником? При выборе хода я придерживаюсь базовой трехэтапной стратегии, которая строится так.

1. Если у компьютера есть возможность сделать ход, который принесет ему победу, - сделать этот ход.

2. Иначе, если человек сможет победить следующим ходом, блокировать этот ход.

3. Иначе занять лучшую из оставшихся клеток. Самая лучшая клетка расположена в центре поля, менее ценны угловые клетки, еще ниже ценятся все оставшиеся клетки

1. Если у компьютера есть возможность сделать ход, который принесет ему победу, - сделать этот ход.



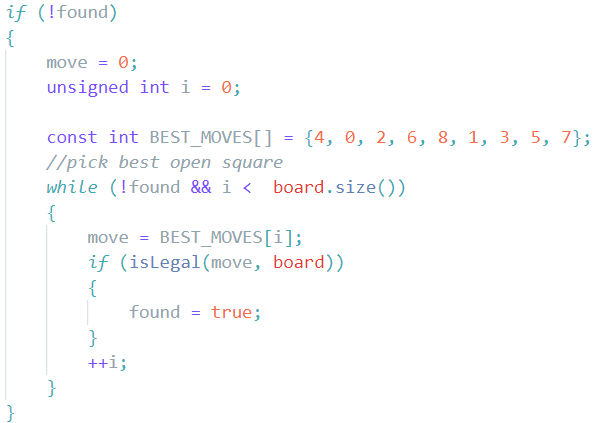
Сначала я перебираю в цикле все возможные варианты хода от 0 до 8. Проверяю, не противоречит ли каждый из ходов правилам. Если ход допустим правилами, то я ставлю фигуру компьютера в выбранную клетку и проверяю, приводит ли такой ход компьютер к победе. Если этот ход не дает победы, пробую поставить фигуру в следующую свободную клетку. Однако если ход привел компьютер к победе, то цикл завершается. Я нашел ход (found равно true), который должен сделать компьютер (занять клетку номер move), чтобы победить.

Далее я проверяю, должен ли искусственный интеллект перейти к следующему этапу описанной стратегии. Если выигрышный ход пока не найден (found равно false ), проверяю, может ли пользователь выиграть следующим ходом:

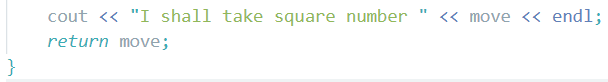


Сначала я перебираю в цикле все возможные варианты хода от 0 до 8. Проверяю, не противоречит ли каждый из ходов правилам. Если ход допустим правилами, то я ставлю фигуру пользователя в выбранную клетку и проверяю, приводит ли такой ход пользователя к победе. Затем отменяю найденный выигрышный ход пользователя так, чтобы эта клетка вновь опустела. Если проверенный ход не приводит пользователя к победе, проверяю следующую пустую клетку. Правда, если ход приводит пользователя к победе, то я нашел тот ход, который сейчас должен обязательно сделать компьютер (found равно true). Компьютер поставит свой символ в клетку номер move, чтобы не дать пользователю выиграть следующим ходом.

Далее я проверяю, должен ли компьютер перейти к третьему этапу стратегии, заложенной в искусственном интеллекте. Если я пока не нашел выигрышного хода ( found равно false ), то просматриваю ряд желательных ходов, начиная с самого выгодного, и выбираю первый, который не противоречит правилам:



На данном этапе выполнения функции я нашел ход, который должен сделать компьютер. В зависимости от ситуации этот ход может принести компьютеру победу, не допустить победы пользователя либо просто позволяет занять оптимальную из оставшихся свободных клеток. Итак, компьютер должен объявить ход и вернуть номер соответствующей клетки:



**announceWinner()**

Эта функция получает победителя игры, фигуру, которой играл компьютер, и фигуру, которой играл пользователь. Далее функция объявляет победителя партии или констатирует ничью:

