Лабораторная работа №4 Иксики-нолики

[c++](https://neroid.ru/category/coding/cpp/) / От [admin](https://neroid.ru/author/admin/)

**Цель:**

1. Закрепить навыки работы с перечислениями;
2. Закрепить навыки работы с структурами;
3. Освоить методы составления многофайловых программ.

Постановка задачи

Создайте три файла:

* Файл реализующий игру «Иксики-нолики» используя функции описанные в вспомогательном файле;
* Вспомогательный файл. Содержит реализацию основных функций игры;
* Заголовочный файл. Нужен для связи главного и вспомогательного файлов. Не забудьте добавить защиту от повторного включения файла.

Вспомогательный файл должен содержать следующие функции:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | /\*\*  \* Выполняет следующие действия:  \*  1. Очищает игровое поле заполняя его пробелами  \*  2. Случайным образом определяет чей ход  \*  3. Устанавливает символ для Игрока (Задаётся параметром userChar)  \*  4. Устанавливает символ для бота  \*  5. Возвращает заполненную структуру  \*/  Game initGame(char userChar);  /\*\*  \* Выполняет следующие действия:  \*  1. Очищает экран  \*  2. Отображает содержимое игрового поля. Например так (можно по своему):  \*         a   b   c  \*       -------------  \*     1 | O | O | X |  \*       -------------  \*     2 |   |   |   |  \*       -------------  \*     3 |   |   | X |  \*       -------------  \*/  void updateDisplay(const Game game);  /\*\*  \* Выполняет ход бота. В выбранную ячейку устанавливается символ которым играет бот.  \* Бот должен определять строку, столбец или диагональ в которой у игрока больше всего иксиков/ноликов и ставить туда свой символ. Если на поле ещё нет меток, бот должен ставить свой знак в центр. В остальных случаях бот ходит рандомно.  \*/  void botTurn(Game\* game);  /\*\*  \* Функция отвечает за ход игрока. Если игрок вводит не допустимые  \* значения, ругнуться и попросить ввести заново  \*/  void userTurn(Game\* game);  /\*\*  \* Функция определяет как изменилось состояние игры после последнего хода.  \* Функция сохраняет новое состояние игры в структуре game и передаёт ход другому  \* игроку.  \* Функция возвращает true, если есть победитель или ничья, иначе false.  \*/  bool updateGame(Game\* game); |

Структура Game выглядит следующим образом:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | /\* Состояния игры \*/  enum Status {     PLAY,            // Игра продолжается     USER\_WIN,        // Игрок победил     BOT\_WIN,         // Бот победил     NOT\_WIN          // Ничья. Победителя нет, но и на поле нет свободной ячейки  };  struct Game{     char bord[3][3];  // Игровое поле     bool isUserTurn;  // Чей ход. Если пользователя, то isUserTurn = true     char userChar;    // Символ которым играет пользователь     char botChar;     // Символ которым играет бот     Status status;  }; |

В отчёт включить

1. Код всех 3x файлов. Перед каждым файлом должно быть указано его название, например:  
   **main.cpp**



|  |  |
| --- | --- |
| 1 | Ваш код здесь |

1. Сам проект разместите в репозитории с отчётом.