Санкт-Петербургский институт точной механики и оптики
(технический университет)

Кафедра "Компьютерные технологии"

И.О.Добрицкий, А.А.Куликов, А.А.Шалыто

Игра "Lines"

Программирование с явным выделением состояний

Проектная документация

Проект создан в рамках
"Движения за открытую проектную документацию"
http://is.ifmo.ru

Санкт-Петербург
2003

Содержание

Введение

Для алгоритмизации и программирования задач логического управления была предложена SWITCH-технология, которая в дальнейшем была развита применительно к разработке программного обеспечения событийных систем [1].

Настоящая работа представляет собой одно из применений SWITCH-технологии. При этом используется подход, названный в работе [1] "программированием с явным выделением состояний".

Апробация в области компьютерных игр этой технологии выполнена на примере известной игры "Lines". На каждом шаге этой игры требуется переместить шары по прямоугольной доске, размеры которой могут быть заданы, так, чтобы выстроить шары одинакового цвета в линию (горизонтальную, вертикальную или диагональную) предварительно установленного размера. После этого шары удаляются. Игра заканчивается при заполнении всего поля шарами. Для учета количества набранных очков используется "шкала ценностей" удаленных шаров. Количество набранных очков сохраняется в соответствующей таблице.

Игра представляет интерес даже для неискушенных в компьютерных играх людей (особенно секретарей различных начальников), поскольку она очень увлекательна и широко распространена в сети *Internet* с конца 80-х годов.

Игра, которая лежит в основе настоящей работы, создана Дмитрием Кивилевым в 1998 году. Сайт игры находится по адресу http://sorcerersoft.com/lines98/rus. Особенности и отличия предлагаемой реализации будут описаны ниже.

1. Постановка задачи

Цель настоящей работы состоит в создании программы, реализующей с помощью автоматного подхода [1] игру "Lines", правила которой описаны ниже. Внешний вид программы приведен на рис. 1.

Рис. 1. Фрагмент игры

Опишем правила "классической" игры "Lines".

1. Игровое поле представляет собой квадратную сетку ячеек, в которых располагаются шары, но не более одного в каждой из них.
2. На каждом ходу в выбранных случайным образом ячейках появляются новые шары.
3. Игрок с помощью "мыши" должен переместить один из шаров в свободную ячейку, стараясь выстраивать из одинаковых шаров непрерывные горизонтальные, вертикальные или диагональные линии.
4. Если после перемещения или появления шара линия достигнет заданной длины, то все шары, входящие в нее, удаляются. При этом игрок набирает определенное количество очков.
5. Перемещение шара из исходной ячейки в "ячейку назначения" допускается, если существует путь, проходящий по свободным ячейкам, расположенным рядом друг с другом по вертикали или по горизонтали.
6. Если после перемещения шара была удалена линия, то на следующем ходу новые шары, которые должны были бы появиться в соответствии с п.2, не появляются.
7. Игра заканчивается, когда на игровом поле не остается свободных ячеек – они все заполнены шарами.
8. Цель игры - набрать как можно больше очков, которые запоминаются в соответствующей таблице.
9. В игре используется “шкала ценностей”. При этом количество очков, которые получает игрок при составлении линии шаров, вычисляется (при $n \geq k$) по формуле $q = n \cdot (n - k + 1)$ (n – количество выстроенных в линию шаров, k –минимальное количество удаляемых шаров, q – получаемые очки).
10. В любой момент игры на экран выводится подсказка о шарах, которые появятся на следующем шаге. Эти подсказки отображаются на поле в виде уменьшенных изображений шаров.

2. Отличия по сравнению с “классической” реализацией

В результате модификации программы появилась возможность изменять:

- размеры игрового поля;
- минимальное количество шаров, которые удаляются при выстраивании в линию;
- количество появляющихся шаров на каждом шаге.

В программе, видимо, впервые для написания подобных игр был использован автоматный подход. Вся логика управления игрой сосредоточена в двух автоматах.

3. Графика

При разработке графической части программы было принято решение использовать интерфейс игры, взятой за основу. Его графические элементы выделены при помощи программного продукта *HyperSnap-DX 4* компании *Hyperionics* (http://www.hyperionics.com).

4. Структура программы

Программа написана на языке *C++* в среде *Visual C++ 6.0.*

Состояние, цвет и номер картинки, выводимой в соответствующую клетку игрового поля, хранятся в двумерном массиве ***map***, как элементы структуры. Разработан класс ***cell,*** в котором агрегируется автомат “Ячейка”. Существует два экземпляра этого класса *ball* и *click_ball*. Объект *ball* – игровой шар (шар, с которым в данный момент производятся действия). Объект *click_ball* необходим только для того, чтобы сообщать программе о том, в какой ячейке поля было произведено нажатие курсором мыши.

Программа построена по стандартной схеме использования системных функций *win32 API.*

По событиям таймера (*WM_TIMER*) и нажатию левой кнопки мыши (*WM_LBUTTONDOWN*) происходит передача управления в автомат “Управление игрой” (в последнем случае инициализируется объект *click_ball*).

В процессе работы программы объект *ball* изменяет свои поля *posx* и *posy*, с помощью которых производится поиск состояний автомата в описанном выше двумерном массиве ***map***. Поэтому, только изменяя поля *posx* и *posy,* можно переходить от одной ячейки к другой и при переходах вызывать соответствующие автоматы. Это позволяет применять только два экземпляра автомата, а не создавать экземпляр автомата для каждой ячейки. Кроме того, массив ***map*** удобен для поиска кратчайшего пути следования шара.

5. Автомат "Управление игрой"

5.1. Словесное описание

Этот автомат обеспечивает управление:

- передвижением и прыжками шаров;
- появлением и удалением шаров;
- поиском кратчайшего пути.

5.2. Схема связей

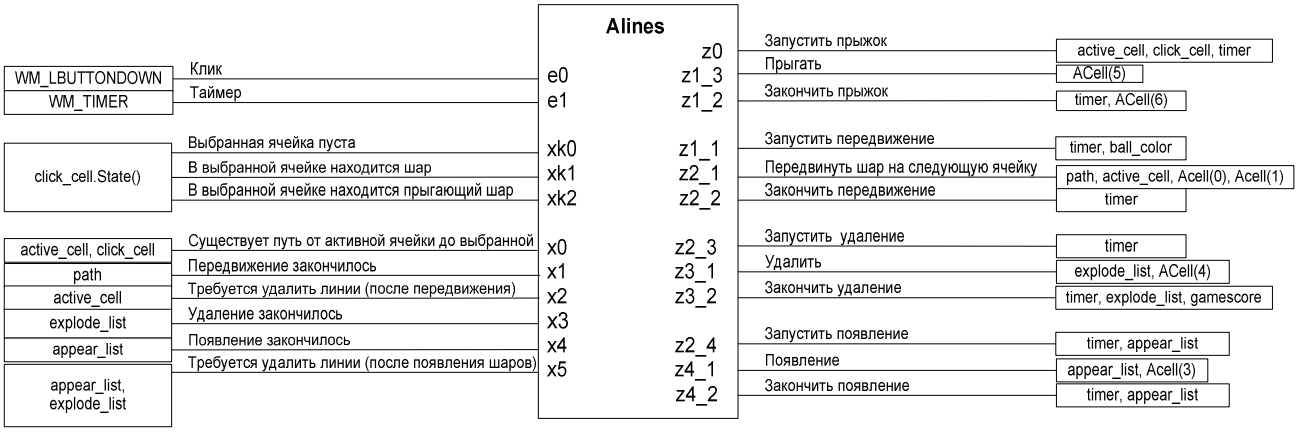

Рис. 2. Схема связей автомата "Управление игрой"

5.3. Граф переходов

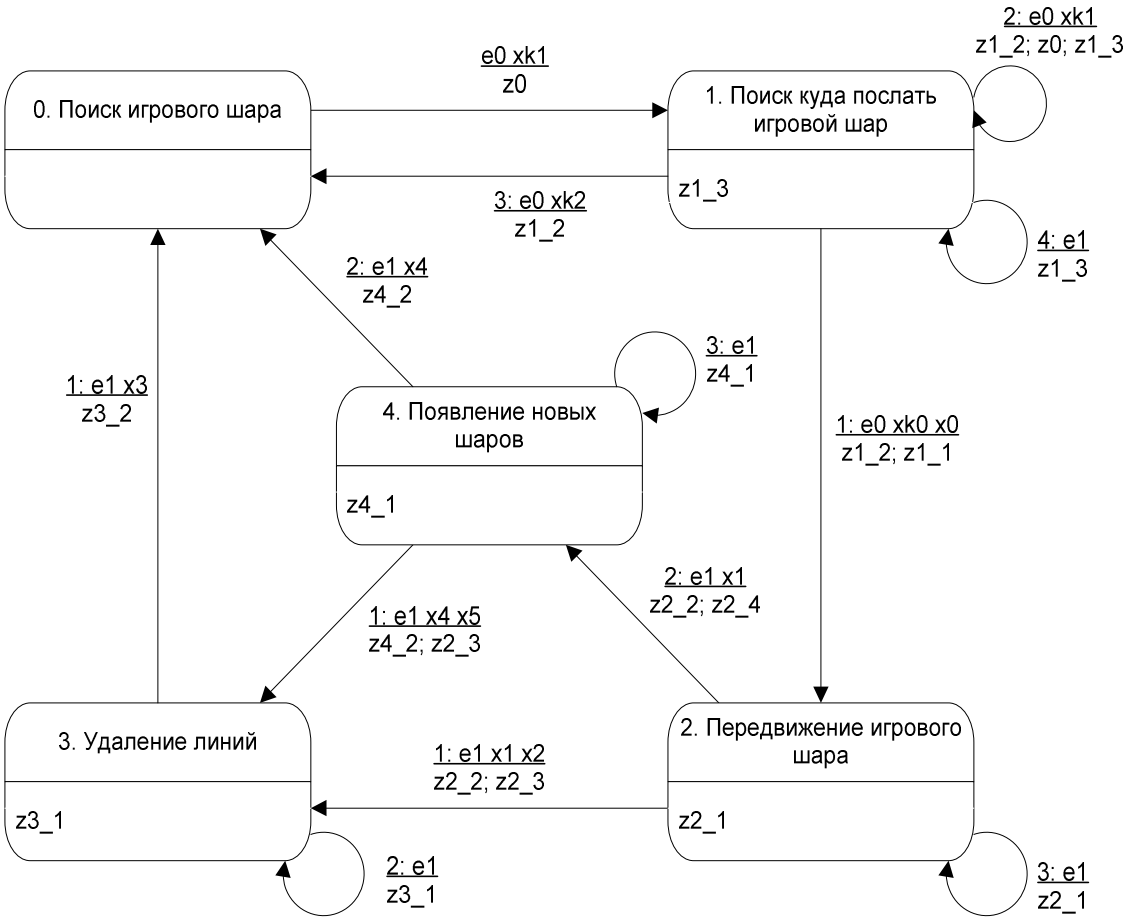

Рис. 3. Граф переходов автомата "Управление игрой" (A0)

6. Автомат "Ячейка"

6.1. Словесное описание

Этот автомат обеспечивает:

- установку шара в ячейке;
- удаление шара;
- подсказку о том, что шар на следующем ходу должен появиться;
- отображение состояния ячейки на экране.

6.2. Схема связей

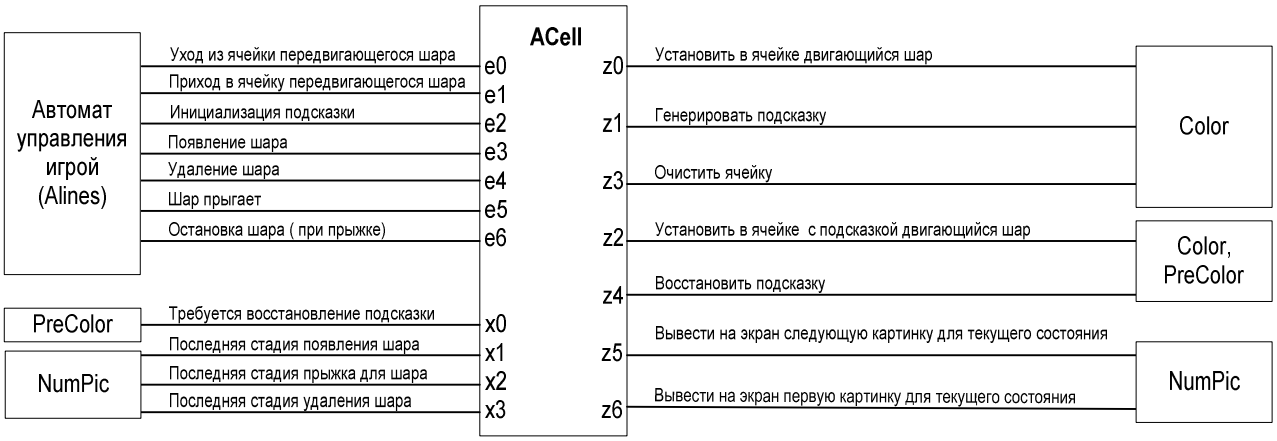

Рис. 4. Схема связей автомата "Ячейка" (A1)

6.3. Граф переходов

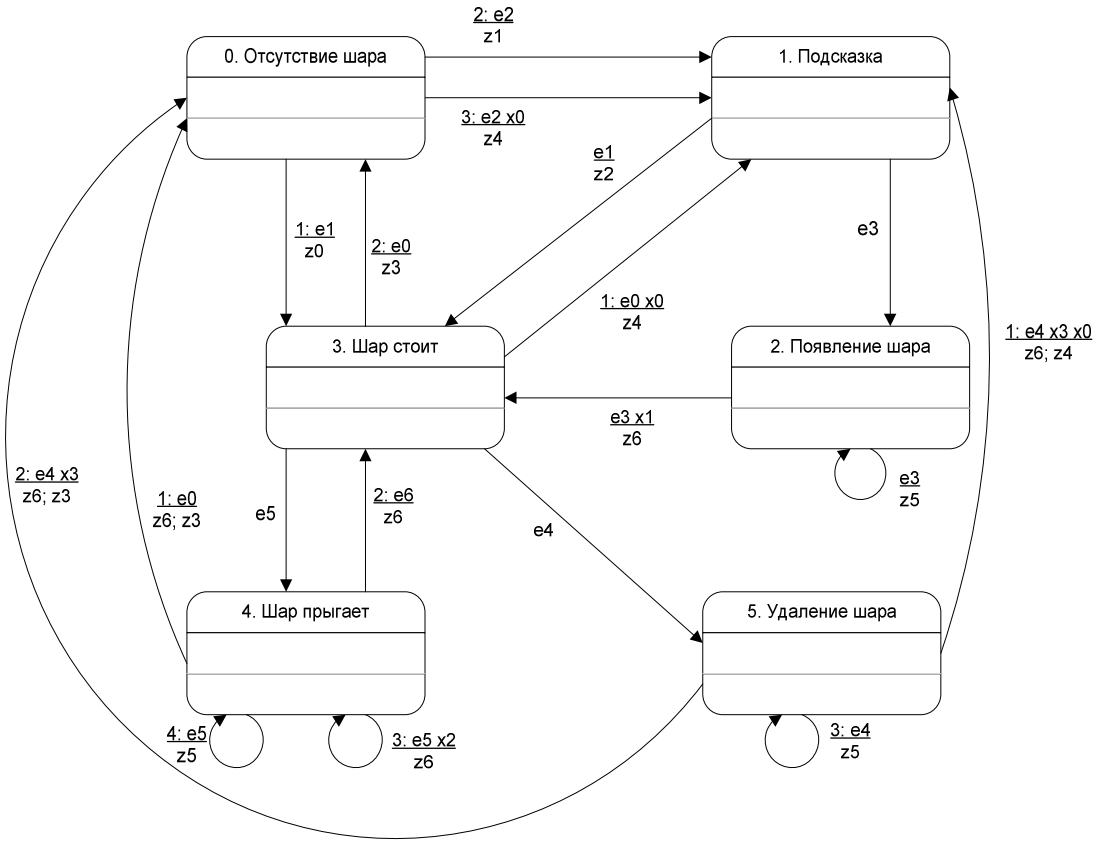

Рис. 5. Граф переходов автомата "Ячейка"

7. Пример отладочного протокола

В процессе разработки программы была обеспечена возможность создания отладочных протоколов (вывод их в файл). На рис.6 приведены скриншоты игры в начале и в конце протоколирования.

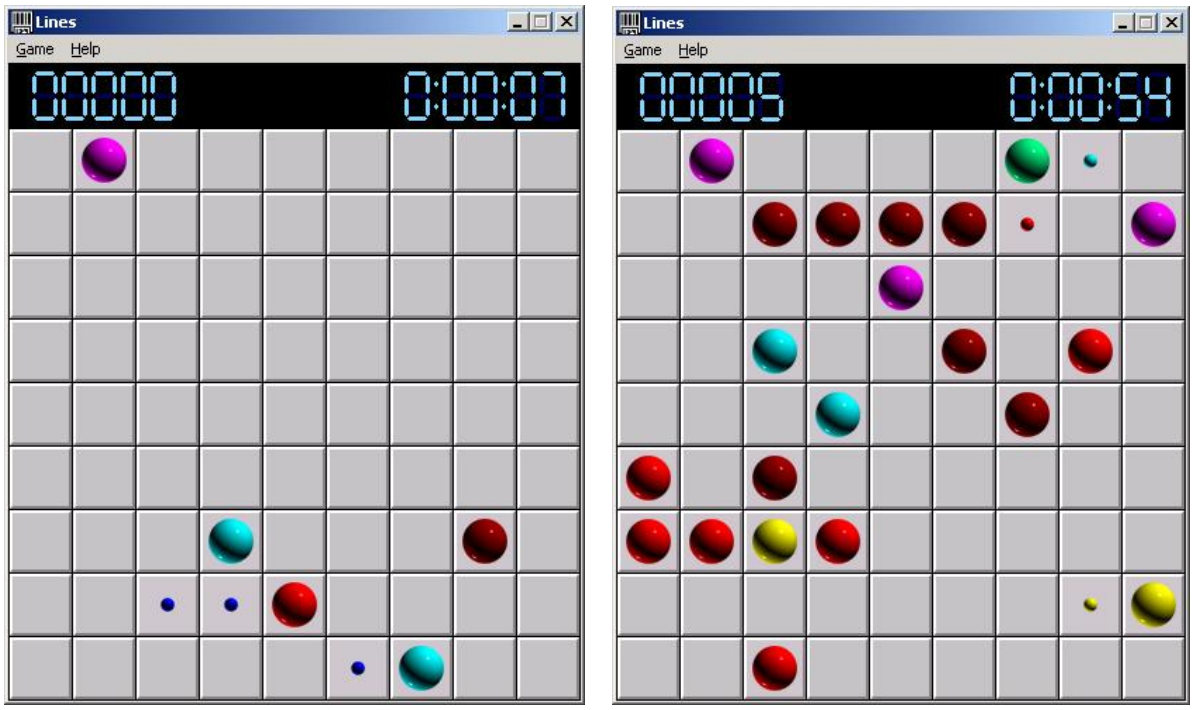

Рис. 6. Состояние в начале и конце отладки

Этот протокол имеет следующий вид:

```
[12:44:33] Новая игра
[12:44:33] => Ячейка (5,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:33] => Ячейка (8,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:33] => Ячейка (2,1) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:33] => Ячейка (4,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:33] => Ячейка (7,9) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:33] => Ячейка (3,8) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:33] => Ячейка (4,8) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:33] => Ячейка (6,9) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:41] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск
куда послать игровой шар".
[12:44:41] => Ячейка (4,7) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:44:42] => Ячейка (4,7) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:44:42] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в
состояние "Передвижение игрового шара".
[12:44:42] => Ячейка (4,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:42] => Ячейка (5,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:42] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:42] => Ячейка (5,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:42] => Ячейка (5,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:42] => Ячейка (5,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (5,5) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:43] => Ячейка (5,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (5,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:43] => Ячейка (5,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (5,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:43] => Ячейка (5,2) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
```

[12:44:43] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:44:43] => Ячейка (3,8) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:43] => Ячейка (4,8) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:43] => Ячейка (6,9) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:43] => Ячейка (3,8) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (4,8) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (6,9) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:43] => Ячейка (9,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:43] => Ячейка (3,6) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:43] => Ячейка (3,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:43] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:44:43] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:44:43] => Ячейка (7,9) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:44:44] => Ячейка (7,9) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:44:44] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:44:44] => Ячейка (7,9) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:44] => Ячейка (7,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:44] => Ячейка (7,8) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:44] => Ячейка (6,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:44] => Ячейка (6,8) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:44] => Ячейка (6,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:44] => Ячейка (6,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:44] => Ячейка (5,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:44] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:44] => Ячейка (4,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (4,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:45] => Ячейка (3,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (3,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:45] => Ячейка (3,6) из состояния "Подсказка" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (3,6) из состояния "Шар стоит" перешла в состояние "Подсказка".
[12:44:45] => Ячейка (3,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (3,5) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:44:45] => Ячейка (3,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:44:45] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:44:45] => Ячейка (9,2) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:45] => Ячейка (3,6) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:45] => Ячейка (3,2) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:44:45] => Ячейка (9,2) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (3,6) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (3,2) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:44:45] => Ячейка (7,1) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:45] => Ячейка (4,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:45] => Ячейка (2,7) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:44:45] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:13] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:45:13] => Ячейка (5,2) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:14] => Ячейка (5,2) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:14] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:14] => Ячейка (5,2) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:14] => Ячейка (4,2) из состояния "Подсказка" перешла в состояние "Шар стоит".
[12:45:14] => Ячейка (4,2) из состояния "Шар стоит" перешла в состояние "Подсказка".
[12:45:14] => Ячейка (4,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:14] => Ячейка (4,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:14] => Ячейка (4,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:14] => Ячейка (4,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:14] => Ячейка (4,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:14] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:45:14] => Ячейка (7,1) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:14] => Ячейка (4,2) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:14] => Ячейка (2,7) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:15] => Ячейка (7,1) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:15] => Ячейка (4,2) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:15] => Ячейка (2,7) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:15] => Ячейка (3,7) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:15] => Ячейка (5,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:15] => Ячейка (3,9) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:15] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:16] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".

[12:45:16] => Ячейка (8,7) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:17] => Ячейка (8,7) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:17] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:17] => Ячейка (8,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (7,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (7,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (6,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (6,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (5,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (5,5) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (5,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (5,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:17] => Ячейка (5,2) из состояния "Подсказка" перешла в состояние "Шар стоит".
[12:45:17] => Ячейка (9,6) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:17] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:45:17] => Ячейка (3,7) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:17] => Ячейка (9,6) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:17] => Ячейка (3,9) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:18] => Ячейка (3,7) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:18] => Ячейка (9,6) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:18] => Ячейка (3,9) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:18] => Ячейка (6,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:18] => Ячейка (4,7) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:18] => Ячейка (1,7) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:18] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:18] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:45:18] => Ячейка (9,6) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:19] => Ячейка (9,6) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:19] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:19] => Ячейка (9,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:19] => Ячейка (8,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:19] => Ячейка (8,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:19] => Ячейка (7,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:19] => Ячейка (7,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:19] => Ячейка (6,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:19] => Ячейка (6,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:19] => Ячейка (6,7) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:19] => Ячейка (6,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:19] => Ячейка (6,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:19] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:45:19] => Ячейка (6,2) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:19] => Ячейка (4,7) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:19] => Ячейка (1,7) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:20] => Ячейка (6,2) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:20] => Ячейка (4,7) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:20] => Ячейка (1,7) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:20] => Ячейка (5,7) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:20] => Ячейка (7,5) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:20] => Ячейка (8,4) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:20] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:20] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:45:20] => Ячейка (5,8) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:21] => Ячейка (5,8) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:21] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:21] => Ячейка (5,8) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (5,7) из состояния "Подсказка" перешла в состояние "Шар стоит".
[12:45:21] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Подсказка".
[12:45:21] => Ячейка (5,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:21] => Ячейка (5,6) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (5,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:21] => Ячейка (5,5) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (5,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:21] => Ячейка (5,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (4,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".

[12:45:21] => Ячейка (4,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (4,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:21] => Ячейка (4,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:21] => Ячейка (3,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (3,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:22] => Ячейка (2,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (2,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:22] => Ячейка (1,3) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (1,3) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:22] => Ячейка (1,4) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (1,4) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:22] => Ячейка (1,5) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (1,5) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:22] => Ячейка (1,6) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:22] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:45:22] => Ячейка (5,7) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:22] => Ячейка (7,5) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:22] => Ячейка (8,4) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:22] => Ячейка (5,7) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (7,5) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (8,4) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:22] => Ячейка (5,3) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:22] => Ячейка (6,4) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:22] => Ячейка (9,8) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:22] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:23] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:45:23] => Ячейка (6,9) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:23] => Ячейка (6,9) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:23] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:23] => Ячейка (6,9) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:23] => Ячейка (7,9) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:23] => Ячейка (7,9) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:23] => Ячейка (7,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:23] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Появление новых шаров".
[12:45:23] => Ячейка (5,3) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:23] => Ячейка (6,4) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:23] => Ячейка (9,8) из состояния "Подсказка" перешла в состояние "Появление шара".
[12:45:24] => Ячейка (5,3) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:24] => Ячейка (6,4) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:24] => Ячейка (9,8) из состояния "Появление шара" перешла в состояние "Шар стоит".
[12:45:24] => Ячейка (7,2) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:24] => Ячейка (8,1) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:24] => Ячейка (8,8) из состояния "Отсутствие шара" перешла в состояние "Подсказка".
[12:45:24] Автомат управления игрой из состояния "Появление новых шаров" перешел в состояние "Поиск игрового шара".
[12:45:24] Автомат управления игрой из состояния "Поиск игрового шара" перешел в состояние "Поиск куда послать игровой шар".
[12:45:24] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Шар прыгает".
[12:45:25] => Ячейка (5,7) из состояния "Шар прыгает" перешла в состояние "Шар стоит".
[12:45:25] Автомат управления игрой из состояния "Поиск куда послать игровой шар" перешел в состояние "Передвижение игрового шара".
[12:45:25] => Ячейка (5,7) из состояния "Шар стоит" перешла в состояние "Отсутствие шара".
[12:45:25] => Ячейка (5,8) из состояния "Отсутствие шара" перешла в состояние "Шар стоит".
[12:45:25] Автомат управления игрой из состояния "Передвижение игрового шара" перешел в состояние "Удаление линий".
[12:45:25] => Ячейка (7,8) из состояния "Шар стоит" перешла в состояние "Удаление шара".
[12:45:25] => Ячейка (6,8) из состояния "Шар стоит" перешла в состояние "Удаление шара".
[12:45:25] => Ячейка (4,8) из состояния "Шар стоит" перешла в состояние "Удаление шара".
[12:45:25] => Ячейка (3,8) из состояния "Шар стоит" перешла в состояние "Удаление шара".
[12:45:25] => Ячейка (5,8) из состояния "Шар стоит" перешла в состояние "Удаление шара".
[12:45:25] => Ячейка (7,8) из состояния "Удаление шара" перешла в состояние "Отсутствие шара".
[12:45:25] => Ячейка (6,8) из состояния "Удаление шара" перешла в состояние "Отсутствие шара".
[12:45:25] => Ячейка (4,8) из состояния "Удаление шара" перешла в состояние "Отсутствие шара".
[12:45:25] => Ячейка (3,8) из состояния "Удаление шара" перешла в состояние "Отсутствие шара".
[12:45:25] => Ячейка (5,8) из состояния "Удаление шара" перешла в состояние "Отсутствие шара".
[12:45:25] Автомат управления игрой из состояния "Удаление линий" перешел в состояние "Поиск игрового шара".

Заключение

Применение автоматного программирования в этой задаче показало, что оно достаточно эффективно при спецификации задачи, ее реализации и протоколирования.

Литература

1. *Шалыто А.А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998.
2. *Шалыто А.А., Туккель Н.И.* Программирование с явным выделением состояний // Мир ПК, 2001. №8, 9.
3. *Наумов А.С., Шалыто А.А.* Система управления лифтом. http://is.ifmo.ru , раздел "Проекты".
4. *Ла Мот А., Ратклифф Д., Семинаторе М. и др.* Секреты программирования игр. СПб.: Питер, 1995.
5. *Туккель Н.И., Шалыто А.А.* Система управления танком для игры "Robocode". Объектно-ориентированное программирование с явным выделением состояний. http://is.ifmo.ru , раздел "Проекты".
6. *Ларман К.* Применение UML и шаблонов проектирования. М.: Вильямс, 2001.

Приложение. Исходные коды программы

lines.cpp:

```
#include "windows.h"
#include "resource.h"

#include "random.h"

#include <list>
#include <queue>
#include <stack>
#include <strstrea.h>

#define MENU_HEIGHT 45

#define TOP_HEIGHT 46 // Высота табло (где отображаются очки) в пикселах
#define CELL_SIZE 45  //Размер стороны ячейки в пикселах (они все квадратные)

#define N1 4   //Количество картинок для появления шарика
#define N2 12  //Количество картинок для прыжка шарика
#define N3 9   //Количество картинок для удаления шарика

#define MAX_MAP_X 20  //Максимальный размер поля по X
#define MAX_MAP_Y 12  //Максимальный размер поля по Y


struct
{
	//Состояние автомата для этой ячейки
	int y;

	//Цвет шарика в этой ячейке
	int color;
```

```
        //Цвет шарика-подсказки в этой ячейке
        //(когда игровой шарик проходит через эту ячейку, тогда color != pre_color)
        int pre_color;

        //Номер картинки, выводимой в текущий момент, при появлении, удалении, прыжке шарика
        int num_pic;
}

// Игровое поле
map[MAX_MAP_X][MAX_MAP_Y];

// Цвет игрового шарика
int ball_color;


struct info
{
        int score;
        int time;
        char name[30];
}

// Лучшие результаты
leaders[3];

// Минимальный результат, чтобы попасть в таблицу
const info null_leader={100,3600,"noname"};

// Состояние автомата управления игрой
int y_lines;

// Время игры
int gametime;

// Очки
int gamescore;

// Тип игры: 0-easy, 1-normal, 2-hard, 3-custom
int gametype;

// Текущий размер поля
int max_x; // по x
int max_y; // по y

//Количество появляющихся шаров
int app_balls;

//Количество удаляемых шаров
int del_balls;

#define LOGGING //Включаем логирование

#ifdef LOGGING
FILE *log;

const char* aCell_states[6] =
{
  "Отсутствие шара",
  "Подсказка",
  "Появление шара",
  "Шар стоит",
  "Шар прыгает",
  "Удаление шара"
};

const char* aLines_states[5] =
{
  "Поиск игрового шара",
  "Поиск куда послать игровой шар",
  "Передвижение игрового шара",
  "Удаление линий",
  "Появление новых шаров"
};

#endif
```

```
HANDLE bmp_0,bmp_prestand, bmp_stand, bmp_jump[N2], bmp_explode[N3], bmp_appear[N1], bmp_numbers,
bmp_points;
HANDLE *bmp[6];

HDC hDC;
HDC hCompatibleDC;
HWND hWnd;
HINSTANCE hInst;
TCHAR szTitle[] = "Lines";
TCHAR szWindowClass[] = "LINES";

RECT clRect;

// Класс "Ячейка"
class cell
{
public:
	int posx; // Позиция ячейки на поле (слева направо от 0)
	int posy; // Позиция ячейки на поле (сверху вниз от 0)
	bool operator ==(const cell & b) const
	{return (b.posx == posx && b.posy == posy);}
	bool operator !=(const cell & b) const
	{return (!(*this == b));}
	int & State() const
	{return map[posx][posy].y;}
	int & Color() const
	{return map[posx][posy].color;}
	int & PreColor() const
	{return map[posx][posy].pre_color;}
	int & NumPic() const
	{return map[posx][posy].num_pic;}


	// Автомат "Ячейка"
	void ACell(int e) const
	{
		int &y=State();

		#ifdef LOGGING
		int y_old = y;
		#endif

		switch(y)
		{
		// Отсутствие шара
		case 0:
			if (e==1)		{z0();		y=3;}
			if (e==2 && x0())	{z4();		y=1;}
			else if (e==2)		{z1();		y=1;}
			break;

		// Подсказка
		case 1:
			if (e==1)		{z2();		y=3;}
			if (e==3)				y=2;
			break;

		// Появление шара
		case 2:
			if (e==3 && x1())	{z6();		y=3;}
			else if (e==3)		{z5();}
			break;

		// Шар стоит
		case 3:
			if (e==0 && x0() )	{z4();		y=1;}
			else if (e==0)		{z3();		y=0;}
			if (e==4)				y=5;
			if (e==5)				y=4;
			break;

		// Шар прыгает
		case 4:
			if (e==0)		{z6();z3();	y=0;}
			if (e==6)		{z6();		y=3;}
			if (e==5 && x2())	{z6();}
			else if (e==5)		{z5();}
			break;
```

```
                // Удаление шара
                case 5:
                        if (e==4 && x3() && x0())     {z6();z4();           y=1;}
                        else if (e==4 && x3())        {z6();z3();           y=0;}
                        else if (e==4)                {z5();}
                        break;
                }

                #ifdef LOGGING
                if (y!=y_old)
                {
                        char s[30];
                        time_t t;
                        time(&t);
                        strftime(s,30,"%X", gmtime(&t));
                        fprintf(log,"[%s] => Ячейка (%d,%d) из состояния \"%s\" перешла в состояние
                        \"%s\".\n",s,posx+1,posy+1,aCell_states[y_old],aCell_states[y]);
                }
                #endif

                DrawState();
        }

        void DrawState() const
        {
                SelectObject(hCompatibleDC, bmp[State()][NumPic()]);
                BitBlt(hDC,posx*CELL_SIZE, TOP_HEIGHT+posy*CELL_SIZE, CELL_SIZE+1, CELL_SIZE+1,
                hCompatibleDC, Color()*CELL_SIZE, 0, SRCCOPY);
        }

private:

        //ВХОДНЫЕ ПЕРЕМЕННЫЕ

        //Требуется восстановление подсказки
        bool x0() const
        {return PreColor()!=-1;}

        //Последняя стадия появления шара
        bool x1() const
        {return (NumPic()==N1-1);}

        //Последняя стадия прыжка для шара
        bool x2() const
        {return (NumPic()==N2-1);}

        //Последняя стадия удаления шара
        bool x3() const
        {return (NumPic()==N3-1);}


        //ВЫХОДНЫЕ ВОЗДЕЙСТВИЯ

        //Установить в ячейке двигающийся шар
        void z0() const
        {Color()=ball_color;}

        //Генерировать подсказку
        void z1() const
        {Color()=random(7);}

        //Очистить ячейку
        void z2() const
        {PreColor()=Color();Color()=ball_color;}

        //Установить в ячейке с подсказкой двигающийся шар
        void z3() const
        {Color()=0;}

        //Восстановить подсказку
        void z4() const
        {Color()=PreColor();PreColor()=-1;}

        //Вывести на экран следующую картинку для текущего состояния
        void z5() const
        {++NumPic();}
```

```
        //Вывести на экран первую картинку для текущего состояния
        void z6() const
        {NumPic()=0;}
};

//Игровой шар
cell ball;

//Выбранный шар (ткнули мышкой)
cell click_ball;

//Список появляющихся шаров
std::list<cell> appear_list;

//Список удаляющихся шаров
std::list<cell> explode_list;

//Путь по которому проходит двигающийся шар
std::stack<cell> path;

std::list<cell>::iterator itr;

// Прототипы функций, встречаемых далее в программе
void ALines(int);
void z0();
void z1_1();
void z1_2();
void z1_3();
void z2_1();
void z2_2();
void z2_3();
void z2_4();
void z3_1();
void z3_2();
void z4_1();
void z4_2();

bool xk0();
bool xk1();
bool xk2();
bool x0();
bool x1();
bool x2();
bool x3();
bool x4();
bool x5();

bool FindEmptyCell(cell &);
void GenerateAppearList();
void CheckAppearList();
bool FindPath(const cell &, const cell &);
bool CheckLines(const cell &);
bool Valid(const cell &);

void GameOver();
void NewGame();
void DrawTime();
void DrawScore();
void DrawTop();

void CheckCustomParameters();
void GetInfo();
void WriteInfo();


ATOM                        MyRegisterClass(HINSTANCE);
BOOL                        InitInstance(HINSTANCE,int);
LRESULT CALLBACK      WndProc(HWND, UINT, WPARAM, LPARAM);
LRESULT CALLBACK      About(HWND, UINT, WPARAM, LPARAM);
LRESULT CALLBACK      Custom(HWND, UINT, WPARAM, LPARAM);
LRESULT CALLBACK      BestResults(HWND, UINT, WPARAM, LPARAM);
LRESULT CALLBACK      GetName(HWND, UINT, WPARAM, LPARAM);


//Главная функция окна
int APIENTRY WinMain(HINSTANCE hInstance,
                     HINSTANCE hPrevInstance,
                     LPSTR     lpCmdLine,
                     int       nCmdShow)
```

```
{
        MSG msg;
        HACCEL hAccelTable;

        randomize();

        hInst=hInstance;
        MyRegisterClass(hInstance);

        if (!InitInstance (hInstance,nCmdShow))
        {
                return FALSE;
        }

        hAccelTable = LoadAccelerators(hInstance, (LPCTSTR)IDC_LINES);

        //Главный цикл сообщений Windows
        while (GetMessage(&msg, NULL, 0, 0))
        {
                if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
                {
                        TranslateMessage(&msg);
                        DispatchMessage(&msg);
                }
        }

        return msg.wParam;
}

//Регистрация класса окна
ATOM MyRegisterClass(HINSTANCE hInstance)
{
        WNDCLASSEX wcex;

        wcex.cbSize = sizeof(WNDCLASSEX);

        wcex.style                      = CS_HREDRAW | CS_VREDRAW;
        wcex.lpfnWndProc        = (WNDPROC)WndProc;
        wcex.cbClsExtra                 = 0;
        wcex.cbWndExtra                 = 0;
        wcex.hInstance          = hInstance;
        wcex.hIcon                      = LoadIcon(hInst, (LPCTSTR)IDI_LINES);
        wcex.hCursor            = LoadCursor(NULL, IDC_ARROW);
        wcex.hbrBackground      = (HBRUSH)(COLOR_WINDOW+1);
        wcex.lpszMenuName       = (LPCSTR)IDC_LINES;
        wcex.lpszClassName      = szWindowClass;
        wcex.hIconSm            = LoadIcon(wcex.hInstance, (LPCTSTR)IDI_SMALL);

        return RegisterClassEx(&wcex);
}

// Создание и отображение окна
BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
{
   hWnd = CreateWindow(szWindowClass, szTitle,
          WS_OVERLAPPED|WS_CAPTION|WS_SYSMENU|WS_MINIMIZEBOX,// WS_OVERLAPPEDWINDOW,
          CW_USEDEFAULT, 0, CW_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);

   if (!hWnd)
   {
      return FALSE;
   }

   ShowWindow(hWnd, nCmdShow);
   UpdateWindow(hWnd);

   return TRUE;
}

// Главная функция окна
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
        static PAINTSTRUCT ps;
        static RECT Rect;
        static HMENU hMenu;
        static HKEY hKey;
        static cell l;
        static int temp;
        static  FILE *f;
```

```
switch (message)
{
// Обработка сообщения при создании окна
case WM_CREATE:
	randomize();

	bmp[0] = &bmp_0;
	bmp[1] = &bmp_prestand;
	bmp[2] = &bmp_appear[0];
	bmp[3] = &bmp_stand;
	bmp[4] = &bmp_jump[0];
	bmp[5] = &bmp_explode[0];

	bmp_0 = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_0));
	bmp_prestand = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_PRESTAND));

	bmp_appear[0] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_APPEAR_1));
	bmp_appear[1] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_APPEAR_2));
	bmp_appear[2] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_APPEAR_3));
	bmp_appear[3] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_APPEAR_4));

	bmp_stand = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_STAND));

	bmp_jump[0] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_3));
	bmp_jump[1] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_2));
	bmp_jump[2] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_1));
	bmp_jump[3] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_2));
	bmp_jump[4] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_3));
	bmp_jump[5] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_STAND));
	bmp_jump[6] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_4));
	bmp_jump[7] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_5));
	bmp_jump[8] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_6));
	bmp_jump[9] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_5));
	bmp_jump[10] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_JUMP_4));
	bmp_jump[11] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_STAND));

	bmp_explode[0] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_1));
	bmp_explode[1] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_2));
	bmp_explode[2] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_3));
	bmp_explode[3] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_4));
	bmp_explode[4] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_5));
	bmp_explode[5] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_6));
	bmp_explode[6] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_7));
	bmp_explode[7] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_8));
	bmp_explode[8] = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_EXPLODE_9));

	bmp_numbers = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_NUMBERS));
	bmp_points = LoadBitmap(hInst,MAKEINTRESOURCE(IDB_POINTS));

	hDC = GetDC(hWnd);
	hCompatibleDC = CreateCompatibleDC(hDC);

	#ifdef LOGGING
	log = fopen("lines.log","wt");
	#endif

	GetInfo();
	hMenu = GetSubMenu(GetMenu(hWnd),0);
	CheckMenuItem(hMenu, gametype+IDM_EASY, MF_CHECKED);
	GetWindowRect(hWnd,&Rect);
	MoveWindow(hWnd,Rect.left,Rect.top,CELL_SIZE*max_x+7,TOP_HEIGHT+CELL_SIZE*max_y+MENU_
	HEIGHT, TRUE);
	NewGame();

	SetTimer(hWnd,0,1000,NULL);
	break;

// Обработка сообщения от нажатия левой кнопки мыши
case WM_LBUTTONDOWN:
	click_ball.posx = LOWORD(lParam)/CELL_SIZE;
	click_ball.posy = (HIWORD(lParam)-TOP_HEIGHT)/45;
	ALines(0);
	break;
```

```
// Обработка сообщения от таймера
case WM_TIMER:
	switch (LOWORD(wParam))
	{
	case 0:
		gametime++;
		DrawTime();
		break;
	case 1:
		ALines(1);
		break;
	}
	break;

// Обработка сообщения WM_COMMAND
case WM_COMMAND:
	switch (LOWORD(wParam))
	{
	case IDM_EASY:
		max_x=9;
		max_y=6;
		app_balls = 2;
		del_balls = 4;
		break;

	case IDM_NORMAL:
		max_x=9;
		max_y=9;
		app_balls = 3;
		del_balls = 5;
		break;

	case IDM_HARD:
		max_x=20;
		max_y=12;
		app_balls = 10;
		del_balls = 4;
		break;

	case IDM_CUSTOM:
		DialogBox(hInst, (LPCTSTR)IDD_CUSTOMBOX, hWnd, (DLGPROC)Custom);
		CheckCustomParameters();
		break;
	}

	switch (LOWORD(wParam))
	{
	case IDM_ABOUT:
		DialogBox(hInst, (LPCTSTR)IDD_ABOUTBOX, hWnd, (DLGPROC)About);
		break;

	case IDM_EXIT:
		DestroyWindow(hWnd);
		break;

	case IDM_EASY:
	case IDM_NORMAL:
	case IDM_HARD:
	case IDM_CUSTOM:
		CheckMenuItem(hMenu, gametype+IDM_EASY, MF_UNCHECKED);
		CheckMenuItem(hMenu,LOWORD(wParam), MF_CHECKED);
		gametype=LOWORD(wParam)-IDM_EASY;
		GetWindowRect(hWnd,&Rect);
		MoveWindow(hWnd,Rect.left,Rect.top,CELL_SIZE*max_x+7,TOP_HEIGHT+CELL_SIZE*max_
		y+MENU_HEIGHT, TRUE);
	case IDM_NEW:
		NewGame();
		InvalidateRect(hWnd,NULL,FALSE);
		break;

	case IDM_BESTRESULTS:
		DialogBox(hInst, (LPCTSTR)IDD_BESTRESULTSBOX, hWnd, (DLGPROC)BestResults);
		break;

	default:
		return DefWindowProc(hWnd, message, wParam, lParam);

	}
	break;
```

```
        // Обработка сообщения при отрисовке окна
        case WM_PAINT:
                BeginPaint(hWnd, &ps);

                DrawTop();
                for (l.posx=0;l.posx<max_x;l.posx++)
                for(l.posy=0;l.posy<max_y;l.posy++)
                        l.DrawState();

                EndPaint(hWnd, &ps);
                break;

        // Обработка сообщения при уничтожении окна
        case WM_DESTROY:
                DeleteObject(bmp_0);
                DeleteObject(bmp_prestand);
                DeleteObject(bmp_appear[0]);
                DeleteObject(bmp_appear[1]);
                DeleteObject(bmp_appear[2]);
                DeleteObject(bmp_appear[3]);
                DeleteObject(bmp_stand);
                DeleteObject(bmp_jump[0]);
                DeleteObject(bmp_jump[1]);
                DeleteObject(bmp_jump[2]);
                DeleteObject(bmp_jump[3]);
                DeleteObject(bmp_jump[4]);
                DeleteObject(bmp_jump[5]);
                DeleteObject(bmp_jump[6]);
                DeleteObject(bmp_jump[7]);
                DeleteObject(bmp_jump[8]);
                DeleteObject(bmp_jump[9]);
                DeleteObject(bmp_jump[10]);
                DeleteObject(bmp_jump[11]);
                DeleteObject(bmp_explode[0]);
                DeleteObject(bmp_explode[1]);
                DeleteObject(bmp_explode[2]);
                DeleteObject(bmp_explode[3]);
                DeleteObject(bmp_explode[4]);
                DeleteObject(bmp_explode[5]);
                DeleteObject(bmp_explode[6]);
                DeleteObject(bmp_explode[7]);
                DeleteObject(bmp_explode[8]);
                DeleteObject(bmp_numbers);
                DeleteObject(bmp_points);

                DeleteDC(hCompatibleDC);
                ReleaseDC(hWnd,hDC);

                WriteInfo();

                #ifdef LOGGING
                fclose(log);
                #endif

                PostQuitMessage(0);
                break;

        // Обработка сообщения по умолчанию
        default:
                return DefWindowProc(hWnd, message, wParam, lParam);
        }
        return 0;
}

// Главная функция окна диалога “About”
LRESULT CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)
{
        switch (message)
        {
                case WM_INITDIALOG:
                                return TRUE;

                case WM_COMMAND:
                        if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)
                        {
                                EndDialog(hDlg, LOWORD(wParam));
                                return TRUE;
                        }
                        break;
```

```
        }
    return FALSE;
}

// Главная функция окна диалога "Custom"
LRESULT CALLBACK Custom(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)
{
        static char szVal[10];

        switch (message)
        {
                case WM_INITDIALOG:
                        ostrstream(szVal,sizeof(szVal)) << max_x << ends;
                        SendDlgItemMessage(hDlg, IDC_EDIT1, WM_SETTEXT, (WPARAM)0,(LPARAM)szVal);
                        ostrstream(szVal,sizeof(szVal)) << max_y << ends;
                        SendDlgItemMessage(hDlg, IDC_EDIT2, WM_SETTEXT, (WPARAM)0,(LPARAM)szVal);
                        ostrstream(szVal,sizeof(szVal)) << app_balls << ends;
                        SendDlgItemMessage(hDlg, IDC_EDIT3, WM_SETTEXT, (WPARAM)0,(LPARAM)szVal);
                        ostrstream(szVal,sizeof(szVal)) << del_balls << ends;
                        SendDlgItemMessage(hDlg, IDC_EDIT4, WM_SETTEXT, (WPARAM)0,(LPARAM)szVal);
                        return TRUE;

                case WM_COMMAND:
                        if (LOWORD(wParam) == IDOK)
                        {
                                SendDlgItemMessage(hDlg, IDC_EDIT1, WM_GETTEXT,
                                (WPARAM)sizeof(szVal),(LPARAM)szVal);
                                istrstream(szVal,sizeof(szVal)) >> max_x;
                                SendDlgItemMessage(hDlg, IDC_EDIT2, WM_GETTEXT,
                                (WPARAM)sizeof(szVal),(LPARAM)szVal);
                                istrstream(szVal,sizeof(szVal)) >> max_y;
                                SendDlgItemMessage(hDlg, IDC_EDIT3, WM_GETTEXT,
                                (WPARAM)sizeof(szVal),(LPARAM)szVal);
                                istrstream(szVal,sizeof(szVal)) >> app_balls;
                                SendDlgItemMessage(hDlg, IDC_EDIT4, WM_GETTEXT,
                                (WPARAM)sizeof(szVal),(LPARAM)szVal);
                                istrstream(szVal,sizeof(szVal)) >> del_balls;
                                EndDialog(hDlg, LOWORD(wParam));
                        }
                        if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)
                        {
                                EndDialog(hDlg, LOWORD(wParam));
                                return TRUE;
                        }
                        break;
        }
    return FALSE;
}

// Главная функция окна диалога BestResult
LRESULT CALLBACK BestResults(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)
{
        static char szVal[50];
        static int i, h ,m1,m2,s1,s2;

        switch (message)
        {
                case WM_INITDIALOG:
                        for (i=0;i<3;i++)
                        {
                                h=leaders[i].time;
                                s2=h%60; h/=60;
                                s1=s2%10; s2/=10;
                                m2=h%60; h/=60;
                                m1=m2%10; m2/=10;
                                ostrstream(szVal,sizeof(szVal)) << leaders[i].name<< ends;;
                                SendDlgItemMessage(hDlg, IDC_EDIT1+i, WM_SETTEXT,
                                (WPARAM)0,(LPARAM)szVal);
                                ostrstream(szVal,sizeof(szVal))<< leaders[i].score<<"
                                "<<h<<':'<<m2<<m1<<':'<<s2<<s1<< ends;
                                SendDlgItemMessage(hDlg, IDC_EDIT4+i, WM_SETTEXT,
                                (WPARAM)0,(LPARAM)szVal);
                        }
                        return TRUE;
```

```
                case WM_COMMAND:
                        if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)
                        {
                                EndDialog(hDlg, LOWORD(wParam));
                                return TRUE;
                        }
                        break;
        }
    return FALSE;
}

// Главная функция окна диалога “GetName”
LRESULT CALLBACK GetName(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)
{
        static char szVal[30];

        switch (message)
        {
                case WM_INITDIALOG:
                                return TRUE;

                case WM_COMMAND:
                        if (LOWORD(wParam) == IDOK)
                        {
                                SendDlgItemMessage(hDlg, IDC_EDIT1, WM_GETTEXT, (WPARAM)sizeof(szVal),
                                (LPARAM)szVal);
                                ostrstream(leaders[gametype].name,sizeof(leaders[gametype].name)) <<
                                szVal << ends;
                        }
                        if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)
                        {
                                EndDialog(hDlg, LOWORD(wParam));
                                return TRUE;
                        }

                        break;
        }
    return FALSE;
}

//Автомат "Управление игрой"
void ALines(int e)
{
        int y_old = y_lines;

        switch(y_lines)
        {
        //Поиск игрового шара
        case 0:
                if (e==0 && xk1() )          { z0();                      y_lines=1;}
                break;

        //Поиск куда послать игровой шар
        case 1:
                if (e==0 && xk0() && x0() )  { z1_2(); z1_1();            y_lines=2;}
                else
                if (e==0 && xk1() )          { z1_2(); z0(); z1_3();}
                else
                if (e==0 && xk2() )          { z1_2();                    y_lines=0;}
                else
                if (e==1)                    { z1_3(); }
                break;

        //Передвижение игрового шара
        case 2:
                if     (e==1 && x1() && x2() ){ z2_2(); z2_3();           y_lines=3;}
                else if (e==1 && x1() )      { z2_2(); z2_4();           y_lines=4;}
                else if (e==1)               { z2_1(); }
                break;

        //Удаление линии
        case 3:
                if (e==1 && x3() )           { z3_2();                   y_lines=0;}
                else if (e==1)               { z3_1(); }
                break;
```

```
        //Появление новых шаров
        case 4:
               if (e==1 && x4() && x5() )    { z4_2(); z2_3();            y_lines=3;}
               else if (e==1 && x4() )       { z4_2();                    y_lines=0;}
               else if (e==1)                { z4_1(); }
               break;
        }

        #ifdef LOGGING
        if (y_lines!=y_old)
        {
               char s[30];
               time_t t;
               time(&t);
               strftime(s,30,"%X", gmtime(&t));
               fprintf(log,"[%s] Автомат управления игрой из состояния \"%s\" перешел в состояние
               \"%s\".\n",s,aLines_states[y_old],aLines_states[y_lines]);
        }
        #endif

        if (y_old!=y_lines)

        switch(y_lines)
        {
        case 1:
               z1_3();
               break;
        case 2:
               z2_1();
               break;
        case 3:
               z3_1();
               break;
        case 4:
               z4_1();
               break;
        }
}


//ВХОДНЫЕ ПЕРЕМЕННЫЕ

//Выбранная ячейка пуста
bool xk0()
{
        return (click_ball.State()==0 || click_ball.State()==1);
}

//В выбранной ячейке находится шар
bool xk1()
{
        return (click_ball.State()==3);
}

//В выбранной ячейке находится прыгающий шар
bool xk2()
{
        return (click_ball.State()==4);
}

//Существует путь от активной ячейки до выбранной
bool x0()
{
        return FindPath(ball,click_ball);
}

//Передвижение закончилось
bool x1()
{
        return path.empty();
}

//Требуется удалить линии (после передвижения)
bool x2()
{
        return CheckLines(ball);
}
```

```
//Удаление закончилось
bool x3()
{
	itr=explode_list.begin();
	return (itr->State()==0);//N3
}

//Появление закончилось
bool x4()
{
	itr=appear_list.begin();
	return (itr->State()==3);//N1
}

//Требуется удалить линии (после появления шаров)
bool x5()
{
	itr=appear_list.begin();
	while (itr!=appear_list.end())
		CheckLines(*itr++);

	return (explode_list.size()!=0);
}


// ВЫХОДНЫЕ ВОЗДЕЙСТВИЯ

//Запустить прыжок
void z0()
{
	SetTimer(hWnd,1,50,NULL);
	ball=click_ball;
}

//Запустить передвижение
void z1_1()
{
	SetTimer(hWnd,1,50,NULL);
	ball_color=ball.Color();
}

//Закончить прыжок
void z1_2()
{
	KillTimer(hWnd,1);
	ball.ACell(6);
}

//Прыгать
void z1_3()
{
	ball.ACell(5);
}

//Передвинуть шар на следующую ячейку
void z2_1()
{
	ball.ACell(0);
	ball=path.top();
	path.pop();
	ball.ACell(1);
}

//Закончить передвижение
void z2_2()
{
	KillTimer(hWnd,1);
}

//Запустить  удаление
void z2_3()
{
	SetTimer(hWnd,1,20,NULL);
}
```

```
//Запустить появление
void z2_4()
{
	CheckAppearList();
	SetTimer(hWnd,1,50,NULL);
}

//Удалить
void z3_1()
{
	itr=explode_list.begin();
	while (itr!=explode_list.end())
		(*itr++).ACell(4);
}

//Закончить удаление
void z3_2()
{
	KillTimer(hWnd,1);

	gamescore += (explode_list.size() - del_balls + 1) * explode_list.size();
	DrawScore();

	explode_list.clear();
}

//Появление
void z4_1()
{
	itr=appear_list.begin();
	while (itr!=appear_list.end())
		(*itr++).ACell(3);
}

//Закончить появление
void z4_2()
{
	KillTimer(hWnd,1);
	GenerateAppearList();
}

//Проверка того, что через ячейку in проходит подлежащая удаления линия (или несколько линии)
bool CheckLines(const cell &in)
{
	int x=in.posx;
	int y=in.posy;
	int c=in.Color();
	int i,j;
	cell l;
	bool b=false;

	i=1;while ((x+i<max_x)&&(map[x+i][y].y == 3)&&(map[x+i][y].color == c)) ++i;
	j=1;while((x-j>=0)&&(map[x-j][y].y == 3)&&(map[x-j][y].color == c)) ++j;
	if (j+i-1>=del_balls)
	{
		l.posx=x+i;
		l.posy=y;
		for(int k=0;k<i+j-1;k++)
		{
			l.posx--;
			explode_list.push_back(l);
		}
		b=true;
	}

	i=1;while ((y+i<max_y)&&(map[x][y+i].y == 3)&&(map[x][y+i].color == c)) ++i;
	j=1;while((y-j>=0)&&(map[x][y-j].y == 3)&&(map[x][y-j].color == c)) ++j;
	if (j+i-1>=del_balls)
	{
		l.posx=x;
		l.posy=y+i;
		for(int k=0;k<i+j-1;k++)
		{
			l.posy--;
			explode_list.push_back(l);
		}
		b=true;
	}
```

```
	i=1;while ((x+i<max_x)&&(y+i<max_y)&&(map[x+i][y+i].y == 3)&&(map[x+i][y+i].color == c)) ++i;
	j=1;while((x-j>=0)&&(y-j>=0)&&(map[x-j][y-j].y == 3)&&(map[x-j][y-j].color == c)) ++j;
	if (j+i-1>=del_balls)
	{
		l.posx=x+i;
		l.posy=y+i;
		for(int k=0;k<i+j-1;k++)
		{
			l.posx--;
			l.posy--;
			explode_list.push_back(l);
		}
		b=true;
	}

	i=1;while ((x+i<max_x)&&(y-i>=0)&&(map[x+i][y-i].y == 3)&&(map[x+i][y-i].color == c)) ++i;
	j=1;while((x-j>=0)&&(y+j<max_y)&&(map[x-j][y+j].y == 3)&&(map[x-j][y+j].color == c)) ++j;
	if (j+i-1>=del_balls)
	{
		l.posx=x+i;
		l.posy=y-i;
		for(int k=0;k<i+j-1;k++)
		{
			l.posx--;
			l.posy++;
			explode_list.push_back(l);
		}
		b=true;
	}

	if (b)
	{
		explode_list.remove(in);
		explode_list.push_back(in);
	}

	return b;

}

//Поиск пути следования шарика из ячейки from в in
bool FindPath(const cell &from, const cell &in)
{
	struct
	{
		cell pred;
		int mark;
	}
	v[MAX_MAP_X][MAX_MAP_Y];

	cell k,l;

	std::queue<cell> q;

	for (int i=0;i<max_x;i++)
	for(int j=0;j<max_y;j++)
		v[i][j].mark = 0;

	v[from.posx][from.posy].mark=1;
	q.push(from);

	while (!q.empty())
	{
		k=q.front();
		for (int i=0;i<4;i++)
		{
			l=k;
			switch(i)
			{
			case 0:
				l.posx--;
				break;

			case 1:
				l.posx++;
				break;
```

```
                        case 2:
                                l.posy--;
                                break;

                        case 3:
                                l.posy++;
                                break;
                        }

                        if (Valid(l) && !v[l.posx][l.posy].mark)
                        {
                                v[l.posx][l.posy].mark=1;
                                v[l.posx][l.posy].pred=k;
                                q.push(l);

                                if (l==in)
                                {
                                        do
                                        {
                                                path.push(l);
                                                l = v[l.posx][l.posy].pred;

                                        } while (l!=from);
                                        return true;
                                }
                        }
                }

                q.pop();
        }

        return false;
}

// Проверить список появляющихся шаров на предмет занятости соответствующей ячейки
void CheckAppearList()
{
        int tmp;
        itr=appear_list.begin();
        while (itr!=appear_list.end())
        {
                if (itr->State()==3)
                {
                        tmp=itr->PreColor();
                        itr->PreColor()=-1;
                        FindEmptyCell(*itr);
                        itr->PreColor()=tmp;
                        (*itr).ACell(2);
                }
                ++itr;
        }

}

//Создать список появляющихся шаров
void GenerateAppearList()
{
        appear_list.clear();
        cell l;
        for(int i=0;i<app_balls;i++)
                if (FindEmptyCell(l))
                {
                        appear_list.push_back(l);
                        l.ACell(2);
                }
                else return;

}

//Найти пустую ячейку
bool FindEmptyCell(cell &in)
{
        cell l;
        l.posx=random(max_x);
        l.posy=random(max_y);
```

```
	if (l.State()==0)
	{
		in=l;
		return true;
	};

	for (int i=0;i<max_x*max_y;i++)
	{
		if (l.posx!=max_x-1) l.posx++;
		else if (l.posy!=max_y-1) {l.posy++;l.posx=0;}
			else {l.posx=0;l.posy=0;};

		if (l.State()==0)
		{
			in=l;
			return true;
		}
	}

	GameOver();
	return false;
}

//Проверить можно ли через ячейку in двигаться шару
bool Valid(const cell &in)
{
	return (in.posx >= 0) && (in.posx < max_x) && (in.posy >= 0) && (in.posy < max_y) &&
	(in.State() == 0 || in.State() == 1);
}

//Подготовка ресурсов для новой игры
void NewGame()
{
	for (int i=0;i<max_x;i++)
	for(int j=0;j<max_y;j++)
	{
		map[i][j].y=0;
		map[i][j].color=0;
		map[i][j].pre_color=-1;
		map[i][j].num_pic=0;
	}

	y_lines =0;

	gamescore = 0;
	gametime = 0;

	#ifdef LOGGING
	char s[30];
	time_t t;
	time(&t);
	strftime(s,30,"%X", gmtime(&t));
	fprintf(log,"[%s] Новая игра\n",s);
	#endif

	SetTimer(hWnd,0,1000,NULL);

	cell l;

	for(i=0;i<del_balls;i++)
	{
		FindEmptyCell(l);
		ball_color=random(7);
		l.ACell(1);
	}
	GenerateAppearList();

}

//Окончание игры
void GameOver()
{
	KillTimer(hWnd,0);

	#ifdef LOGGING
	char s[30];
	time_t t;
	time(&t);
	strftime(s,30,"%X", gmtime(&t));
```

```
	fprintf(log,"[%s] Конец игры\n",s);
	#endif

	if (gametype<3)
	{
		if (leaders[gametype].score<gamescore || (leaders[gametype].score==gamescore &&
		leaders[gametype].time>gametime))
		{
			DialogBox(hInst, (LPCTSTR)IDD_GETNAMEBOX, hWnd, (DLGPROC)GetName);
			leaders[gametype].score=gamescore;
			leaders[gametype].time=gametime;
		}
		else DialogBox(hInst, (LPCTSTR)IDD_BESTRESULTSBOX, hWnd, (DLGPROC)BestResults);
	} else MessageBox(hWnd,"Your custom game is over...", "Condolences",MB_OK);
	NewGame();
	InvalidateRect(hWnd,NULL,FALSE);
}

//Отразить на табло продолжительность игры
void DrawTime()
{
	SelectObject(hCompatibleDC, bmp_numbers);
	int h=gametime;
	int s2=h%60; h/=60;
	int s1=s2%10; s2/=10;
	int m2=h%60; h/=60;
	int m1=m2%10; m2/=10;
	BitBlt(hDC,max_x*CELL_SIZE-29,5,20,37,hCompatibleDC,s1*19,0,SRCCOPY);
	BitBlt(hDC,max_x*CELL_SIZE-50,5,20,37,hCompatibleDC,s2*19,0,SRCCOPY);
	BitBlt(hDC,max_x*CELL_SIZE-77,5,20,37,hCompatibleDC,m1*19,0,SRCCOPY);
	BitBlt(hDC,max_x*CELL_SIZE-98,5,20,37,hCompatibleDC,m2*19,0,SRCCOPY);
	BitBlt(hDC,max_x*CELL_SIZE-125,5,20,37,hCompatibleDC,h*19,0,SRCCOPY);
	SelectObject(hCompatibleDC, bmp_points);
	BitBlt(hDC,max_x*CELL_SIZE-56, 5,5,37,hCompatibleDC,0,0,SRCCOPY);
	BitBlt(hDC,max_x*CELL_SIZE-104, 5,5,37,hCompatibleDC,0,0,SRCCOPY);
}

//Отразить на табло текущие очки
void DrawScore()
{
	SelectObject(hCompatibleDC, bmp_numbers);
	int t=gamescore;
	for (int i=0; i<5; i++)
	{
		BitBlt(hDC,100-21*i,5, 20, 37, hCompatibleDC, (t%10)*19, 0, SRCCOPY);
		t /=10;
	}
}

//Отрисовать табло
void DrawTop()
{
	PatBlt(hDC,0, 0, 46*max_x, 46, BLACKNESS);
	DrawScore();
	DrawTime();
}

//Проверить заданные параметры игры
void CheckCustomParameters()
{
	if (max_x<6) max_x=6;
	if (max_x>20) max_x=20;
	if (max_y<2) max_y=2;
	if (max_y>12) max_y=12;

	if (del_balls<2) del_balls=2;

	if (del_balls>(max_x>max_y?max_x:max_y)) del_balls = (max_x>max_y?max_x:max_y);

	if (app_balls+del_balls > max_x*max_y)
	{
		app_balls=max_x*max_y-del_balls;
	}
}
```

```
//Получение информации о лучших игроках
void GetInfo()
{
	FILE *in;
	if (in = fopen("leaders.dat","rb"))
	{
		fread(&gametype,sizeof(int),1,in);
		fread(&max_x,sizeof(int),1,in);
		fread(&max_y,sizeof(int),1,in);
		fread(&app_balls,sizeof(int),1,in);
		fread(&del_balls,sizeof(int),1,in);
		for (int i=0;i<3;i++)
			fread(&leaders[i],sizeof(info),1,in);
		fclose(in);
	}
	else
	{
		gametype=0;
		max_x=9;max_y=9;
		app_balls=3;del_balls=5;
		for (int i=0;i<3;i++)
			leaders[i]=null_leader;
	}
}


//Сохранить информацию о лучших игроках
void WriteInfo()
{
	FILE *out;
	if (out = fopen("leaders.dat","wb"))
	{
		fwrite(&gametype,sizeof(int),1,out);
		fwrite(&max_x,sizeof(int),1,out);
		fwrite(&max_y,sizeof(int),1,out);
		fwrite(&app_balls,sizeof(int),1,out);
		fwrite(&del_balls,sizeof(int),1,out);
		for (int i=0;i<3;i++)
			fwrite(&leaders[i],sizeof(info),1,out);
		fclose(out);
	}
}
```