**Практическая работа 4.2. Схема создания игры змейка. Требования.**

· Размеры игрового поля - 30x30 (30клеток по ширине и 30 клеток по высоте).

· Игровое поле сделайте в виде сетки из линий (см. рис. 8).

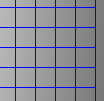


Рис. 8

· Размеры одной клетки игрового поля -20x20 (20 пикселей по ширине и 20 пикселей по высоте).

· Игра должна располагаться внутри окна - 800x650 (800 пикселей по ширине и 650пикселей по высоте).

· В заголовке окна должна быть надпись: Игра "Змейка" (см. рис.9)

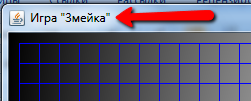


Рис. 9

· Пользователь не должен иметь возможность изменять размеры окна по ширине или высоте.

· Справа от игрового поля должны располагаться две кнопки: Новая игра, Выход(см. рис. 10).



Рис. 10

При нажатии на кнопкуНовая игра –игра начинается с начала с обнулением очков и возврата длины змейки к исходному состоянию. При нажатии на кнопку Выход –происходит выход из игры с закрытием окна игрового поля.

· Справа от игрового поля необходимо отображать количество набранных очков(см. рис. 10). При поедании змейкой объекта прибавляется десять очков.

· В начале игры размер змейки составляет три клетки (см.рис. 11). Она появляется в центре игрового поля и начинает движение влево (см. рис. 11). Голова змейки представляет собой одну клетку и отличается от клеток туловища змейки (см. рис.11):

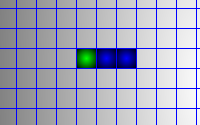


Рис. 11

· Объект для "поедания" змейкой должен появляться в случайном месте игрового поля, которое не занято змейкой. Этот объект имеет размер в одну клетку (см. рис.12). При поедании объекта –змейка должна увеличиваться на одну клетку, а объект появляется в новом случайном месте.

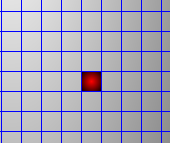


Рис. 12

При столкновении головы змейки с туловищем - игра заканчивается и выводится изображение с надписью: Конец Игры (см. рис. 13).



Рис. 13

При нажатии на кнопкуНовая игра –игра начинается с самого начала.

· Игровое поле и кнопки должно быть расположены на фоне изображения.

· Управление движением змейки происходит при помощи стрелок клавиатуры:Вверх, Вниз, Вправо, Влево. Направление змейки должно изменяться при нажатии клавиш, изменяющих направление вправо или влево от текущего направления. При выборе противоположного направления –направление змейки не меняется.

· При выходе головы змейки за пределы игрового поля –она должна появляться ровно с противоположной стороны (см.рис. 14).

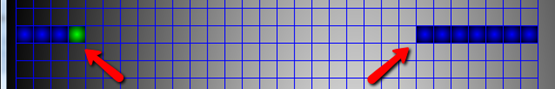


Рис. 14

· При резком развороте змейка должна проходить вплотную к своему туловищу (см.рис. 15).

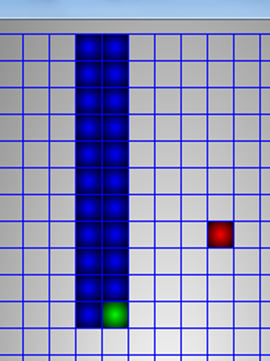


Рис. 15

· Скорость перемещения змейки необходимо подобрать так, чтобы пользователь успевал изменять направление змейки, но играть ему было максимально сложно.

· Движение туловища змейки должно происходить по пути прохождения ее головы (см. рис. 16).

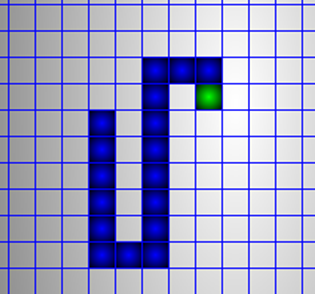


Рис. 16

· Конечный вариант игры необходимо поместить в JAR-архив.

Для создания описанного варианта игры необходимо пять изображений:фон, голова змейки, элемент тела змейки,объект для поедания, изображение конца игры. Данные изображения необходимо заготовить самостоятельно, используя любой графический редактор. При создании игры можно ограничиться самыми простыми изображениями. А в уже готовом варианте игры заменить их более привлекательными.

Первый уровень сложности содержит перечень минимальных элементов, необходимых для написания приложения:

· Создание окна необходимых размеров.

· Указание заголовка окна.

· Создание общего фона.

· Размещение двух кнопок Новая игра, Выход.

· При нажатии на кнопку Выход окно должно полностью закрываться.

· Создание сетки игрового поля30x30.

· Размещение под кнопками надписиСчет: 0.

· В начале игры в центре игрового поля должна появляться только голова змейки,голова не перемещается, тело змейки не отображается.

· В начале игры в случайном месте на игровом поле должен появляться объект для поедания змейкой.

· При нажатии на кнопку Новая игра объект для поедания змейкой должен появляться в новом случайном месте.

· Законченный вариант необходимо представить в виде JAR-архива.

Внимание!!!Подробности по каждому пункту первого уровня сложности необходимо смотреть в основном техническом задании.

Второй уровень сложности является продолжением первого уровня сложности. В дополнение к первому уровню сложности:

· Голова змейки должна перемещаться по игровому полю, тело змейки не отображается.

· Направление движения головы змейки должно изменяться с клавиатуры.

· При поедании объекта –он должен появляться в новом случайном месте, при этом счет увеличивается на десять очков.

· При выходе за пределы экрана –голова змейки должна появляться с другой стороны игрового поля.

· Законченный вариант необходимо представить в виде JAR-архива.

Внимание!!!Подробности по каждому пункту второго уровня сложности необходимо смотреть в основном техническом задании.

Третий уровень сложности является продолжением второго уровня сложности:Выполнение задания в полном объеме

План разработки проекта

Если задача, с которой сталкивается программист, не является простой, то для ее выполнения необходимо составить план реализации проекта. Прежде чем начинать писать программный код,необходимо разложить задачу на составляющие и постепенно выполнять каждый пункт. Для начала можно взять бумагу и сделать несколько рисунков игрового поля. Подумать над решением некоторых вопросов. Можно зарисовать двумерный массив в виде таблицы и вписать в ячейки числа, отвечающие за состояние игрового поля. Представить, что змейка перемещается и записать в массиве числа после перемещения. Подобные зарисовки помогают писать программный код и более ясно разобраться в некоторых вопросах. В этом нам может помочь опыт создания проекта в третьем модуле курса программирования поJava.

Принципы создания двумерных игр мы изучали на примере создания игры "Новогодний дождь".

Все эти принципы подходят и для создания игр Змейка, Морской бой,Пасьянс-Косынка.

Приемы загрузки изображений, работа с таймером, обработка событий от клавиатуры, отрисовка игрового поля, работа с графикой –всё это было продемонстрировано в игре "Новогодний дождь". Построение интерфейса пользователя, размещение кнопок и надписей, обработка событий при нажатии кнопок –всё это было изучено при создании приложений под Windows с построением интерфейса пользователя. Другими словами, все технические моменты, необходимые вам,можно найти в методической литературе по курсу Java.

Более сложная часть–это реализация алгоритма игры Змейка. Алгоритм–это точный план или схема действий, которая может привести к нужному результату. Если программист знает алгоритм, то ему остается запрограммировать его и отладить программный код. Для решения одной задачи может существовать несколько алгоритмов.Рассмотрим один из них. Для хранения информации об игровом поле можно воспользоваться двумерным массивом целых чисел. Размер игрового поля 30x30, поэтому размерность массива будет также 30x30.

//Двумерный массив для хранения игрового поля

int[][] mas;

// Создаем новый массив 30x30

mas = new int[30][30];

Решим, что свободные ячейки игрового поля будут отмечаться в массиве нулями. Для начала можно обнулить массив:

//Заполняем весь массив нулями

for (int i = 0; i < 30; i++) {

for (int j = 0; j < 30; j++) {

mas[i][j] = 0;

}

}

Клетку с объектом для поедания змейкой обозначим в массиве числом минус один: -1 (см. рис.17):

Внимание!!!Цифры на игровом поле показаны с целью демонстрации значений в двумерном массиве. Такой подход можно использовать при отладке



Рис. 17

Голову змейки обозначим в массиве числом один: 1 (см. рис. 18). Каждую последующую клетку от головы змейки будем обозначать по возрастанию числа на единицу. Получится так, что клетки змейки будут обозначены числами: 1,2,3,4,5 и т.д.(см. рис. 18), где единица –это голова змейки. При увеличении длины змейки будет добавляться еще одно число, которое на единицу больше числа в хвосте змейки.

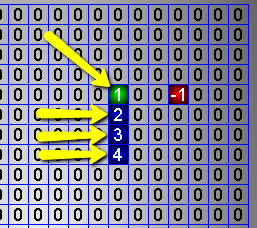


Рис. 18

При поедании объекта змейкой на (рис. 18) к змейке добавится число пять:5, которое станет новым хвостом змейки. Таким образом,рост змейки будет происходить за счет увеличения хвоста.

Теоретически можно голову змейки обозначить числом один:1, а клетки туловища числом два: 2. Но возникнет проблема, когда змейка будет согнута полностью пополам, и две её части будут полностью прилегать друг к другу и мы не сможем определить где хвост змейки (см. рис. 19).



Рис. 19

Поэтому такой метод не подходит! В процессе отрисовки игрового поля мы будем по числу определять, что именно располагается в данной клетке.Отрисовщик будет постоянно перерисовывать игровое поле на основании данных массива.Другими словами, нам предстоит перемещать числа в двумерном массиве. Тогда игровое поле начнет менять свой внешний вид и программирование игры сведется к изменению расположения чисел в массиве.Последний сложный вопрос, который заслуживает внимания –это проблема перемещения тела змейки по игровому полю.При этом тело змейки должно проходить по пути прохождения головы змейки (см. рис. 20).

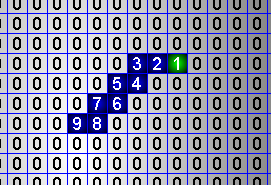


Рис. 20

Как запоминать путь движения змейки? Этот путь каждый раз изменяется и может иметь различный вид. Если тщательно подумать над этим вопросом, то можно увидеть совершенно простой выход из этой ситуации.Перемещение змейки представляет собой перемещение ее головы в одном из четырех направлений, которое на данный момент является текущим и перемещение ее хвоста.Таким образом, для перемещения змейки следует выполнять две операции:

1. Перемещать голову змейки в нужном направлении.

2. Удалять хвост змейки. Удаление хвоста равносильно записи нулевого значения в элемент массива, где находилось число, отвечающее за хвост змейки.

При таком подходе тело змейки будет автоматически проходить по нужному пути. Это и есть пример алгоритма(схемы, плана действий) для решения определенной задачи. Пока неизвестен алгоритм –бесполезно писать программный код!