Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Национальный исследовательский университет

“Высшая школа экономики”»

Жалкова Наталия Евгеньевна

**Мини Курсовая Работа**

Отчет по лабораторной работе №7 студента

2 курса бакалавриата группы БИВ151

Москва 2016

**Формулировка задания**

Создать игру жанра roguelike.

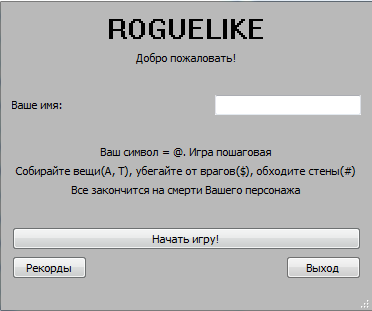
Roguelike (/roʊɡlaɪk/; буквально «Rogue-подобные» (игры), сленг «рогалик») — жанр компьютерных игр. Характерными особенностями классического roguelike являются генерируемые случайным образом уровни, пошаговость и необратимость смерти персонажа — в случае его гибели игрок не может загрузить игру и должен начать её заново.

**Постановка задачи**

Согласно формулировке, необходимо создать игру с генерируемой картой, пошаговым исполнением, описать управляемых компьютером противников, которые рано или поздно приведут к смерти персонажа игрока. Кроме того, необходимо наличие списка рекордов, в который будут записываться результаты каждой игровой сессии. Сама карта будет представлять собой случайно генерируемый массив символов, где каждый символ будет обозначать определенный объект – стена, противник, игрок, ключ, яблоки и так далее. Необходимо создать “выигрышную” стратегию для объекта, управляемого компьютером, учесть наличие стен, через которые проходить нельзя. Файл, в который будут записываться результаты игровых сессий, необходимо защитить от изменения.

**Руководство пользователя**

Вы видите перед собой окно главного меню



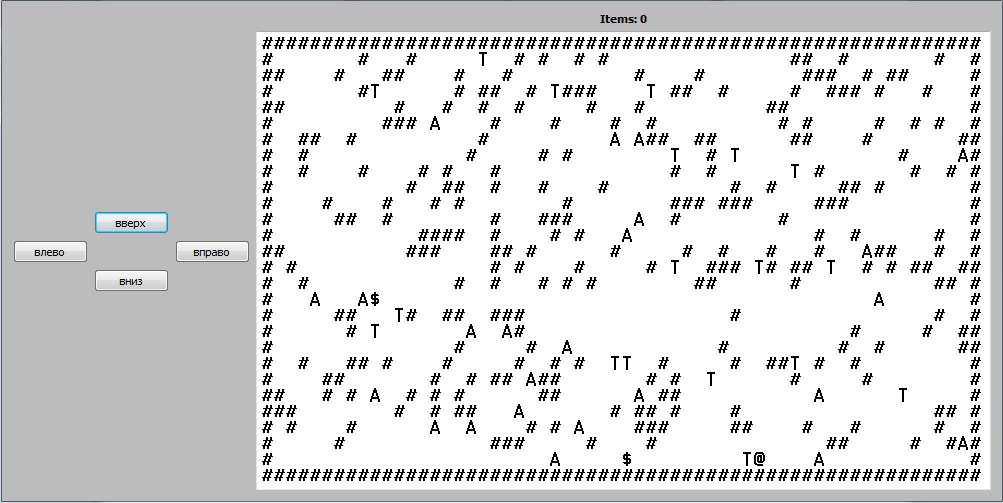
Сверху находится название игры и приветственное сообщение. В поле ввода ниже необходимо вписать имя, которое будет затем записано в список рекордов. Затем идет небольшое описание смысла игры, правила, а так же объяснение обозначения символов.

Нажатие на кнопку “Начать игру!” приведет к открытию нового окна с начавшейся игрой.

Нажатие на кнопку “Рекорды” приведет к открытию нового окна с данными об записанных рекордах.

Нажатие на кнопку “Выход” приведет к закрытию главного меню и выходу из игры.

Начать игру!



Открывается новое окно. Слева находятся четыре кнопки, обозначающие направления. Справа находится карта, по которой передвигается Ваш персонаж. Над картой можно увидеть количество вещей, которых Вы подобрали за игру. После каждого “поднятия” символа A или T счетчик увеличивается.

Стены обозначаются символом #, символы, которые можно подобрать – A, T, символ вашего персонажа - @, символ врага - $.

Передвигаться вы можете с помощью четырех кнопок. При нажатии на кнопку “вверх” персонаж делает 1 шаг вверх. При нажатии на кнопку “вниз” персонаж делает 1 шаг вниз. При нажатии на кнопку “влево” персонаж делает 1 шаг влево. При нажатии на кнопку “вправо” персонаж делает 1 шаг вправо.

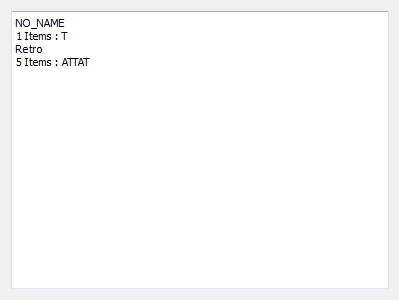
Игра является пошаговой, поэтому при каждом совершенном Вами шаге, два врага также делают один шаг, по направлению к Вам.

Проходить через стены (#) невозможно как Вам, так и противникам. Подбирать символы можете только Вы.

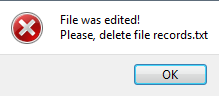
Цель игры – собрать как можно больше символов, перед тем, как один из противников настигнет Вашего персонажа.

Концом игры считается момент, когда позиции Вашего персонажа и любого из противников становятся равны.

Рекорды

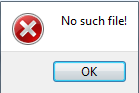


Открывается небольшое окошко, где можно видеть результаты предыдущих сыгранных игр. В нечетных строках находится имя сыгравшего, в четных – количество подобранных им символов, а так же сами символы. Если игрок забыл при начале игры написать имя, оно по умолчанию записывается как NO\_NAME. Если файл был поврежден, отображается ошибка:



Необходимо удалить повреждённый текстовый файл records.txt.

Если файл с рекордами не был найден, отображается ошибка:



Файл был удален, или же еще не было ни одной игры. Это можно исправить начинанием новой игры.

Выход

Игра закрывается. Рекорды, записанные в файл, не удаляются.

**Руководство программиста**

Ключевым элементом является заголовочный файл units.h.

В него входят описания классов enemy – класс, описывающий противника в игре, и player – класс, описывающий игрока, являющийся наследником enemy.

Enemy:

У данного класса существуют private поля, отвечающие за координаты X и Y(PositionX и PositionY соответственно), а так же Symbol, отвечающее за отображаемый символ. Кроме того, в данном классе есть конструкторы, селекторы и модификаторы private полей(get… и set… соответственно), а так же метод moving, отвечающий за движение противника относительно игрока.

В конструкторе по умолчанию координаты объекта задаются случайным образом, а символ задается как $. В конструкторе инициализации данные переменные становятся равными переданным величинам. Селекторы и модификаторы выполняют свою основную роль – позволяют иметь доступ к private полям объекта (получать их величины/изменять их). В методе Moving происходит изменение позиции противника в зависимости от расположения игрока относительно него (в метод передаются игрок и противник). Передвижение осуществляется по следующему алгоритму:

Если координата X врага не равна координате X игрока, то в зависимости от того, больше она или меньше, координата X врага меняется в меньшую или большую сторону соответственно. В ином случае, если координата Y врага не равна координате Y игрока, то в зависимости от того, больше она или меньше, координата Y врага меняется в меньшую или большую сторону соответственно. Каждый раз случайным образом определяется важность координаты (идет ли сначала проверка координаты X или же Y и соответственно, какая будет меняться). Стоит заметить, что движение осуществляется только в одном направлении и только на одну единицу, что обуславливается пошаговостью разрабатываемой игры.

Player:

Данный класс является наследником класса enemy и, соответственно, имеет те же private поля, что и родитель (координаты и символ), а так же собственные: PickedItems[50] – массив символов, отвечающий за подобранные игроком символы и PickedItemsCount – их количество. Так же присутствуют селекторы и модификаторы, работающие аналогично с вышеприведенным классом enemy, а так же конструкторы и метод moving.

С помощью конструктора по умолчанию, у объекта случайным образом генерируются координаты, символом становится @, а количество подобранных объектов становится равным нулю. В методе moving осуществляется перемещение игрока по следующему алгоритму:

В метод передаются изменения координат X и Y соответственно. Внутри метода с помощью модификаторов позиции изменяются на переданные величины, относительно нынешних координат, которые получаются с помощью селекторов.

В классе mainwindow осуществляется работа с окном главного меню, которое появляется при запуске. У данного класса имеется метод шифровки main\_shifer, private слоты для нажатия на кнопки “Начать игру” и “Рекорды”, а так же объекты классов других окон(game, record)

Слот нажатия на кнопку “Начать игру”. При нажатии, создается объект класса game, открывается окно. Из поля, доступного для редактирования (playerName), в конец файла “records.txt” записывается введенное пользователем имя, предварительно зашифровываясь вместе с кодовым словом “lying”. Слот нажатия на кнопку “Рекорды”. При нажатии, создается объекта класса records, открывается окно. В методе main\_shifer получаемая строка шифруется с помощью XOR (каждый символ изменяется с помощью ^ 1001) и возвращается из метода.

В классе records осуществляется работа с окном, где выводится список рекордов, считанных из файла. У данного класса имеются методы: дешифратор (unshifer), работа с файлом(openFile).

Сразу же при открытии окна, используется метод openFile, куда передается название файла “records.txt”. Внутри метода происходит работа с файлом, название которого было передано. Файл открывается на чтение. Если указанный файл не был найден, то отображается ошибка в виде QMessageBox. В ином случае строки считываются из файла в одну строку. Идет расшифровка полученной строки с помощью метода unshifer. Затем производится проверка, был ли файл поврежден. В полученной расшифрованной строке по символам ищется первый или второй встречающийся символ ‘\n’ (нужно помнить, что ‘\n’ после ключевого слова не считается). Если после него НЕ находится слово “lying” – ключевое слово, значит файл был поврежден, и программа сообщает об этом и выходит из цикла проверки. В ином случае часть строки “lying\n” пропускается. Когда встречается третий символ ‘\n’, то это значит, что мы подошли к информации о следующем игроке, следовательно количество ‘\n’ обнуляется. Если, дойдя до конца строки, флаг checking, отображающий, был ли изменен файл, остался равным true, то строка, избавленная от ключевых слов, выводится в textBrowser окна с рекордами. Если же файл был изменен, выводится сообщение об этом. Алгоритм метода расшифровщика unshifer точно такой же, как у шифратора (с помощью XOR).

В классе game осуществляется работа с окном, содержащим игру и, соответственно, с картой. Класс имеет шесть свойств – один объект игрока You, два объекта врага CoolGuy и CoolGuyFriend, один объект карты myMap[28][60], представляющий собой двумерный массив QChar, а так же QChar массивы stopingSymbols и pickingSymbols, представляющие собой наборы останавливающих и подбираемых символов соответственно. Имеющиеся методы: GenerateMap – генерация карты myMap, showMap – вывод myMap в окне, CheckMoving – проверка возможности сдвинуться, IsEnding – проверка на конец игры, UnitsMoving – перемещение всех юнитов, game\_shifer – шифратор. А так же 4 слота: нажатие на четыре разные кнопки перемещения. Кроме того имеется функция to\_string, использующая шаблоны, которая переводит переменную любого типа в string.

При открытии окна, убирается крестик, закрывающий окно, с помощью функции setWindowFlags. Начинает проигрываться музыка, генерируется карта с помощью метода GenerateMap, которая затем отображается с помощью showMap.

Во время генерации карты GenerateMap, сначала отрисовываются стенки по краям. Затем карта заполняется внутри случайным образом – сначала отрисовываются случайные стенки, затем, если стенка не была отрисована, то место либо не заполняется, либо заполняется одним из возможных для подбирания символов. Вывод карты showMap производится следующим образом: символы записываются в одну строку типа QString. Циклы идут по всему пространству карты. Если на полученных координатах находится враг или игрок, то в строку записываются их их символы, если же нет – пишется соответствующий элемент из карты. Так же производится переход на новую строку (в строку добавляется ‘\n’). При нажатии на любую из кнопок передвижения (слоты), используется один и тот же алгоритм. Вызывается метод UnitsMoving, куда передаются значения, на которые будут изменяться позиции, затем карта заново отображается (с учетом проведенных изменений), после чего проверяется, окончена ли игра с помощью метода IsEnding.

В методе UnitsMoving сначала проверяется возможность игрока пройти туда, куда он хочет. Для этого оценивается значение булевой функции CheckMoving, внутри которой проверяется, какой символ находится там, куда игрок намеревался направиться, по умолчанию булева переменная имеет значение true. Если там находится блокирующий символ, метод принимает значение false. Если там находится поднимаемый символ, то он добавляется в массив поднятых игроком символов, количество поднятых символов увеличивается, кроме того данная величина выводится для пользователя на экран, вместо поднятого символа на карте остается пустое место. Если CheckMoving возвращает true, то вызывается метод класса player moving – игрок сдвигается. Затем для врагов используется следующий алгоритм:

Запоминаются первоначальные позиции каждого из врагов. Затем у каждого врага вызывается метод Moving. Если получилось так, что враг сдвинулся на символ стены, то его координаты изменяются на те, что были до движения. В методе IsEnding, куда передается конкретный враг и игрок, происходит следующий алгоритм: если координаты игрока и врага равны, то появляется соответствующее сообщение, а данные об игровой сессии зашифровываются вместе с ключевым словом и записываются в конец файла “records.txt”. Окно с игрой закрывается, оставляя только окно с главным меню. Метод шифратора game\_shifer работает аналогично с шифратором класса mainWindow.

**Выводы**

Таким образом, мною была написана игра типа roguelike. Реализованы классы игрока и врага, создан интерфейс игры, использующий для удобства пользователя несколько окон, просчитана тактика для выигрыша компьютера. Учитывается невозможность проходить сквозь стены как игроку, так и компьютеру. Во многом использована случайная генерация различных условий и происходящих действий. Происходит регистрация игрока, запись его результатов игры в файл. Текстовый файл защищен от изменения, а так же зашифрован. Шаблоны и наследование были использованы. Написана документация. Выполняется различная работа с картой – ее генерация, вывод в окно, изменение. Вся игра работает в режиме пошаговости. Все пункты ТЗ были выполнены.