Санкт-Петербургский политехнический университет

Институт компьютерных наук и технологий

**Кафедра «Компьютерные системы и программные технологии»**

**КУРСОВОЙ ПРОЕКТ**

**Разработка игры "Heatos"**

По дисциплине «Технологии программирования»

Выполнила студентка гр. 3530901/10002 Смирнова А. Н.

Преподаватель Степанов Д. С.

Санкт-Петербург

2022

Санкт-Петербургский политехнический университет

**ЗАДАНИЕ**

**НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА**

студентке группы 3530901/10002 Смирновой Алине Николаевне

1. Тема проекта: создание игры «Heatos» с графическим интерфейсом.

2. Срок сдачи законченного проекта: 23 сентября

3. Исходные данные к проекту: реализовать игру Heatos

4. Содержание пояснительной записки: введение с описанием правил игры, основная часть (технологии JavaFX, MVC и их применение в приложении), описание автоматических тестов для бизнес-логики, заключение, список используемых источников.

Дата получения задания: «22» апреля 2020 г.

Руководитель Степанов Д.С.

Задание приняла к исполнению Смирнова А. Н.

22 апреля 2020

**Содержание**

ВВЕДЕНИЕ ……………………………………………………………………………………………….4

ОПИСАНИЕ ПРЕДЛОЖЕННОГО РЕШЕНИЯ…………………………………………………………5

ТЕСТИРОВАНИЕ ПРОГРАММЫ……………………………………………………………………….6

ЗАКЛЮЧЕНИЕ……………………………………………………………………………………………7

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ………………………………………………………... 8

**ВВЕДЕНИЕ**

Цель работы: создать и протестировать игру “Heatos” c графическим интерфейсом.

**Правила игры:**

Цель игры: Согреть поле, то есть избавиться от всех холодных блоков. Уровень считается проигранным, если от холодных блоков больше нельзя избавиться.

1. В игре принимает участие один игрок.
2. Передвижение блоков производится последовательными кликами на блок, который мы хотим передвинуть, и на ячейку, куда мы хотим его переместить. Двигать можно не все блоки и не на все ячейки (об этом далее), двигаться можно только вверх, вниз, влево или вправо на одну клетку.
3. Каждому уровню соответствует начальное положение всех блоков на игровом поле, размер которого варьируется. Блоки, которые встречаются в игре:
   1. Холодный блок синего цвета отрицательной температуры

Можно двигать на:

* пустые ячейки (температура уменьшится на один)
* теплые блоки (температура увеличится на указанное число)
* блоки-множители (температура блока умножится на указанное число)
  1. Теплый блок красного цвета положительной температуры

Можно двигать на:

* пустые ячейки (температура уменьшится на один; когда температура блока достигает нуля, он становится пустой ячейкой)
* холодные блоки (температура уменьшится на указанное число)
* теплые блоки (температура увеличится на указанное число)
* блоки-множители (температура блока умножится на указанное число)
  1. Блок-множитель белого цвета с индексом умножения

Нельзя двигать

* 1. Блок-камень серого цвета

Нельзя двигать

**ОПИСАНИЕ ПРЕДЛОЖЕННОГО РЕШЕНИЯ**

Для создания графического пользовательского интерфейса (GUI) использовалась библиотека JavaFX. В частности, были использованы элементы: label (лэйбл), button (кнопка) и другие простые объекты. Для отрисовки блоков использовались элементы типа rectangle. Само поле представляет собой контейнер GridPane, что позволяет легко перемещать блоки, обращаясь к конкретным ячейкам сетки. Взаимодействие пользователя с графическим элементом описывается в событиях этого элемента. В данном случае я использовала fx:id для назначения действий при взаимодействии с тем или иным элементом.

Программа была написана с использованием концепция MVC (model-view-controller) для отделения бизнес-логики от визуализации, поэтому весь код разбит на три части controller, model и view.

В view (находится в recourses) содержатся классы hello-view, level-sample, menu-view и rules-view которые отвечают за задание параметров главного окна и окна меню, уровней и правил, где расположены все элементы интерфейса. Также в в recourses хранится css-файл, в котором содержится большая часть стиля игры (шрифты, цвета, размеры объектов и так далее).

В пакете controller содержатся классы, отвечающие за обработку действий пользователя.

Пакет model содержит классы, отвечающие за переменные и методы, предназначенные для работы бизнес-логики. Для описания разных типов блоков были использованы абстрактные классы. Класс поле хранит в себе матрицу блоков, которая является игровым полем.

В соответствии с выбранным шаблоном разработки, пользователь взаимодействует с view, все команды от пользователя обрабатывает controller, который в свою очередь обращается к model.

**ТЕСТИРОВАНИЕ ПРОГРАММЫ**

Для тестирования были написаны три автоматических теста, использующих возможности библиотеки JUnit.

1. Проверяется правильность добавления блока (его расположение в матрице, тип, характеристики).
2. Проверяется правильность осуществления передвижения блока (крайние случаи, обозначение блока “выбранным”, отсутствие побочных блоков на поле).
3. Проверяется правильность определения выигрыша.

**ЗАКЛЮЧЕНИЕ**

Было создано приложение с графическим интерфейсом. Также были разработаны автоматические тесты для проверки работоспособности кода. В ходе выполнения этого задания мною были изучены библиотека JavaFX и шаблон MVC.

Исходные файлы приложения лежат в репозитории на GitHub:

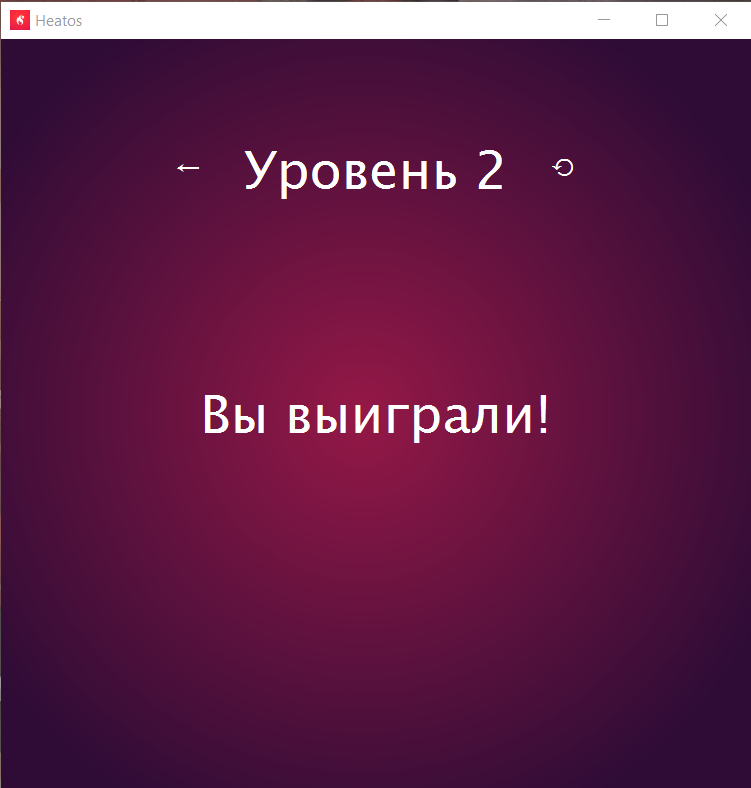
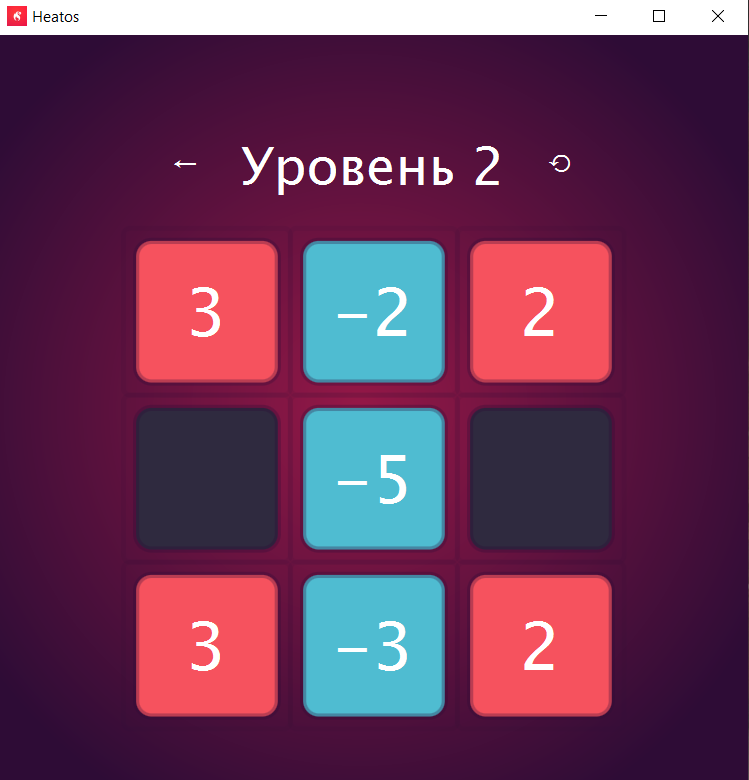
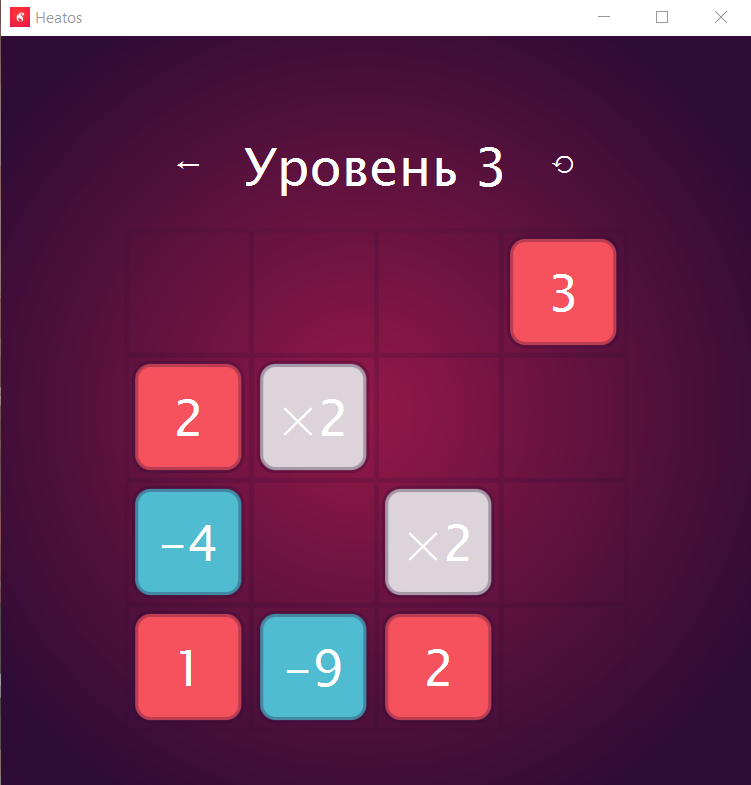
<https://github.com/IrReiku/ConsoleApp>

Рисунок 1. Скриншоты игрового процесса

**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

<https://habr.com/ru/company/redmadrobot/blog/275515/> - настройка зависимостей Gradle

<https://habr.com/ru/post/520172/> - JavaFX (fx:id, View-Controller)

<https://habr.com/ru/post/255561/> - архитектура игр, наследование

<https://gradients.app/en> - создание градиентов, палитр

<https://habr.com/ru/post/474292/> - учебник по JavaFX

<https://openjfx.io/javadoc/19/> - документация JavaFX

<https://www.tutorialspoint.com/javafx/javafx_event_handling.htm> - обработка событий

<https://developer.mozilla.org/ru/docs/Learn/CSS/First_steps> - CSS туториал