Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа «Компьютерных технологий и информационных систем»

**ОТЧЕТ**

по дисциплине «Системный подход в разработке программного обеспечения»

**Выполнил:**

студент группы 5130902/20201 \_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Сафонов

подпись

**Проверил:**

Кандитад тех. Наук., доцент \_\_\_\_\_\_\_\_\_\_\_\_ С. А. Нестеров

подпись

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024г.

Санкт-Петербург, 2024

Оглавление

[1. Смотрим версии Java Puzzle Ball 3](#_Toc177209015)

[16.08.2013 5](#_Toc177209016)

[22.08.2013 6](#_Toc177209017)

[27.09.2013 7](#_Toc177209018)

[16.10.2013 7](#_Toc177209019)

[21.11.2013 8](#_Toc177209020)

[2. Упражнения из занятия 2-2 Задачи программы 10](#_Toc177209021)

[Упражнение 1. 10](#_Toc177209022)

[Упражнение 2. 14](#_Toc177209023)

[Упражнение 3. 15](#_Toc177209024)

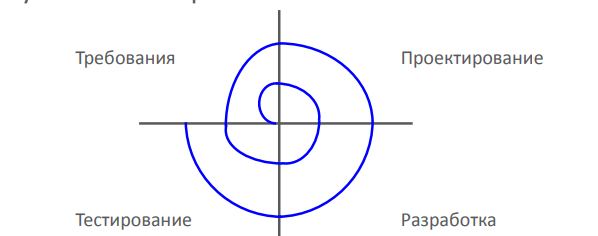
[3. Упражнения – Раздел 2 19](#_Toc177209025)

[Проблема 1: Простая ASCII-графика 19](#_Toc177209026)

[Проблема 2: Исходная ASCII-графика 19](#_Toc177209027)

[Проблема 3: The Snake Box Factory 20](#_Toc177209028)

# Упражнение 1

****

Спиральная модель — это модель, ориентированная на риск, что означает, что основное внимание уделяется управлению рисками посредством нескольких итераций процесса разработки программного обеспечения. Он состоит из следующих этапов:

1. **Определены цели:** напервом этапе спиральной модели мы уточняем, на достижение чего направлен проект, включая функциональные и нефункциональные требования.
2. **Анализ рисков:** на этапе анализа рисков выявляются и оцениваются риски, связанные с проектом.
3. **Проектирование:** на этапе проектирования программное обеспечение разрабатывается на основе требований, собранных на предыдущей итерации.
4. **Разработка:** на этапе разработки происходит написание программного кода, на основе результата проектирования программы.
5. **Оценка:** на этапе оценки оценивается программное обеспечение, чтобы определить, соответствует ли оно требованиям заказчика и высокого ли оно качества.
6. **Планирование:** Следующая итерация спирали начинается с нового этапа планирования, основанного на результатах оценки.

После завершения итерации процесс повторяется, начиная с планирования и продолжая по спирали до тех пор, пока проект не будет завершен. Каждая новая итерация представляет собой дополнительный виток спирали, и в результате проект постепенно совершенствуется и уточняется в соответствии с требованиями заказчика и оценкой рисков.

Основными преимуществами разработки по спиральной модели являются гибкость и адаптивность к изменениям в процессе разработки, а также возможность управления рисками на протяжении всего проекта.

**Достоинства спиральной модели:**

1. Управляемость и контролируемость. Каждый этап разработки имеет четкие критерии оценки, что позволяет лучше контролировать процесс разработки.

2. Адаптивность. Модель позволяет быстро реагировать на изменение требований и вносить коррективы на любом этапе разработки.

3. Уменьшение рисков. Модель позволяет раннее выявление и устранение ошибок и рисков благодаря циклическому характеру разработки.

4. Постепенное наращивание функциональности. Модель позволяет реализовывать проект поэтапно, предоставляя заказчику возможность получать работающие версии продукта на ранних стадиях.

5. Прозрачность. Благодаря структуре модели, все заинтересованные стороны имеют четкое представление о текущем состоянии проекта.

**Недостатки спиральной модели:**

1. Сложность. Модель является более сложной по сравнению с каскадной моделью, так как требует тщательного планирования и управления каждым циклом.

2. анализа рисков и создания прототипов на ранних этапах увеличивает начальные затраты на проект.

3. Сложность оценки сроков и бюджета. Из-за итеративного характера разработки и необходимости реагировать на изменяющиеся требования, сложно точно оценить сроки и бюджет проекта.

4. Сложность документирования. Из-за итеративного характера разработки модель не всегда хорошо документируется, что может затруднять поддержку и сопровождение системы в будущем.

Смотрим версии Java Puzzle Ball

16.08.2013

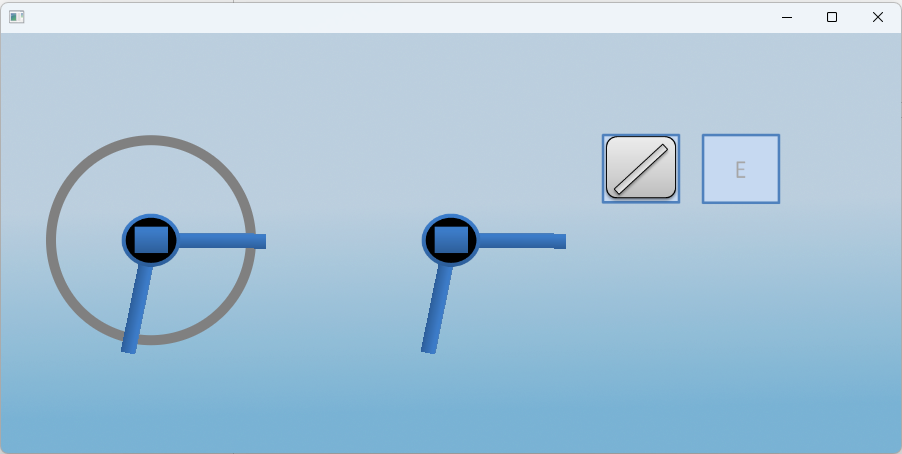


Рисунок 1.1 – Версия игры Java Puzzle Ball 08-16-13

Функции:

* Отображение графических объектов на экране.
* Распознавание событий мыши.
* Вращение объектов BlueBumper.
* Перетаскивание значка в области (N, E).

22.08.2013

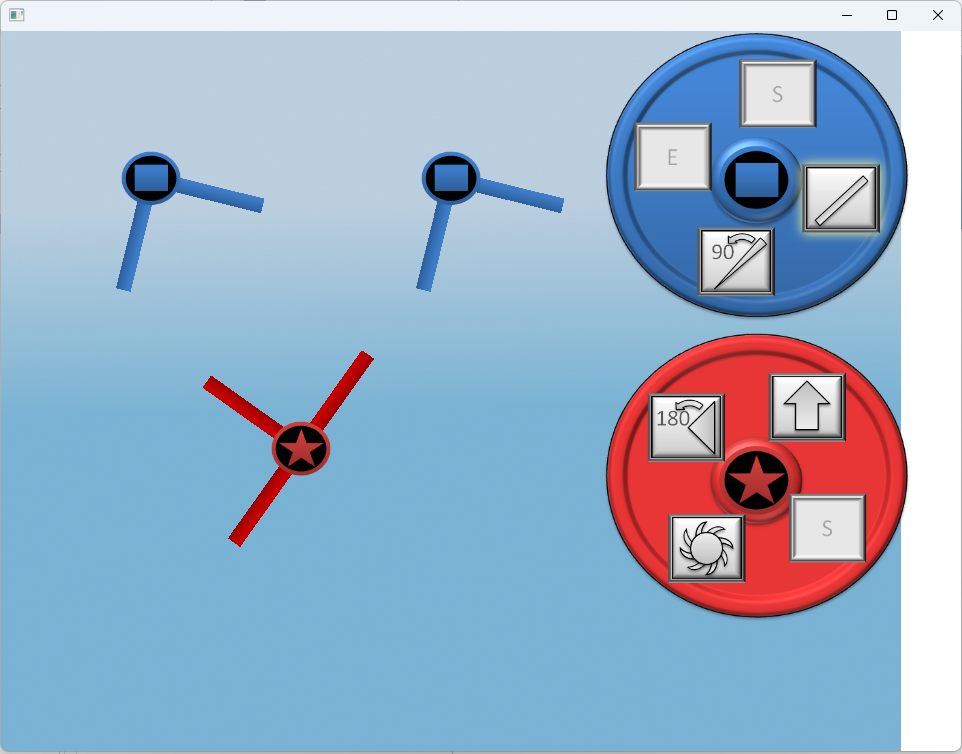


Рисунок 1.2 – Версия игры Java Puzzle Ball 09-16-13

Функции:

* Элементы пользовательского интерфейса (UI).
* Колесики и значки — расположены справа.
* Объект RedBumper.
* Цветные вложения.
* Дополнительные значки для перетаскивания.

27.09.2013

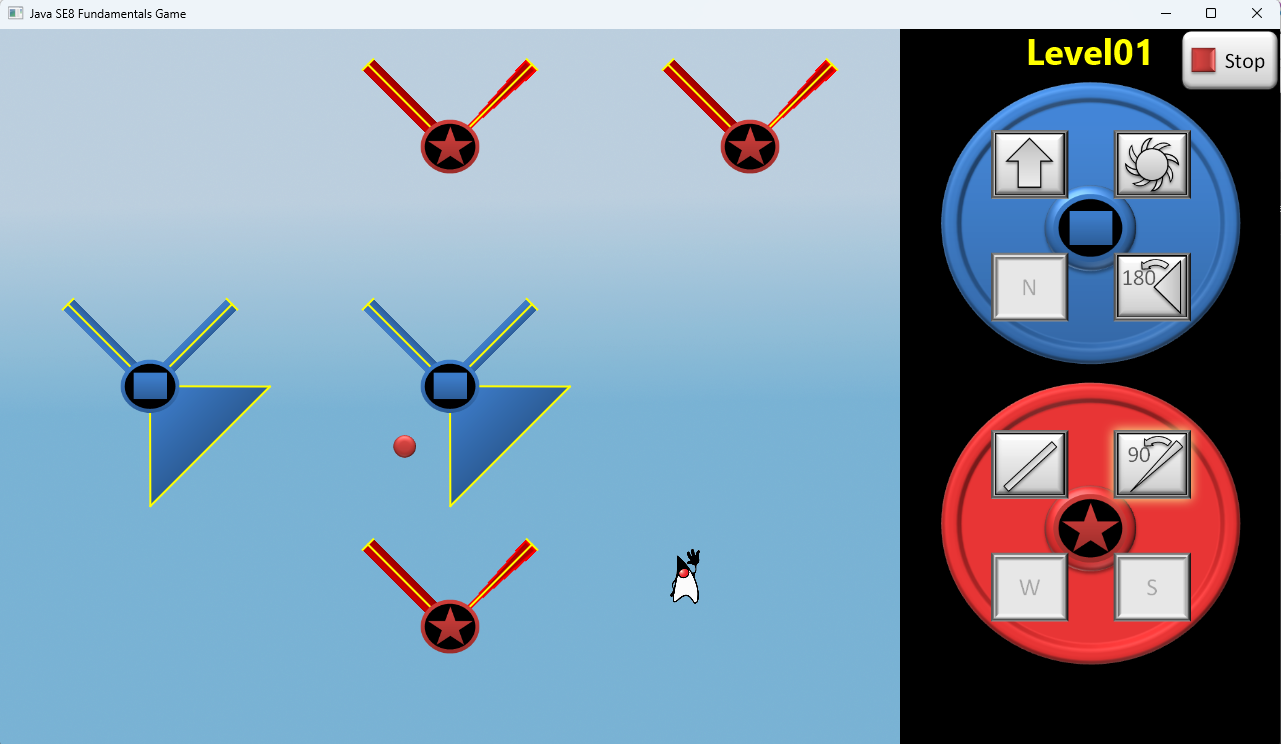


Рисунок 1.3 – Версия игры Java Puzzle Ball 09-27-13

Задача — перенаправить мяч Дюку.

Функции:

* Кнопка Play и цель (Дюк)
* Возможность менять направление движущегося мяча
* Возможность прикреплять дополнительные фигуры
* Желтые линии (для распознавания столкновений)
* Колесики, которые занимают ближайшее положение с шагом 45 градусов

16.10.2013

Появилось всплывающее меню для выбора уровней.

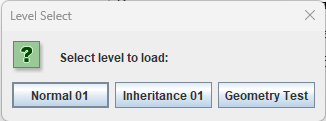


Рисунок 1.4 – Вспылывающее меню для выбора уровней

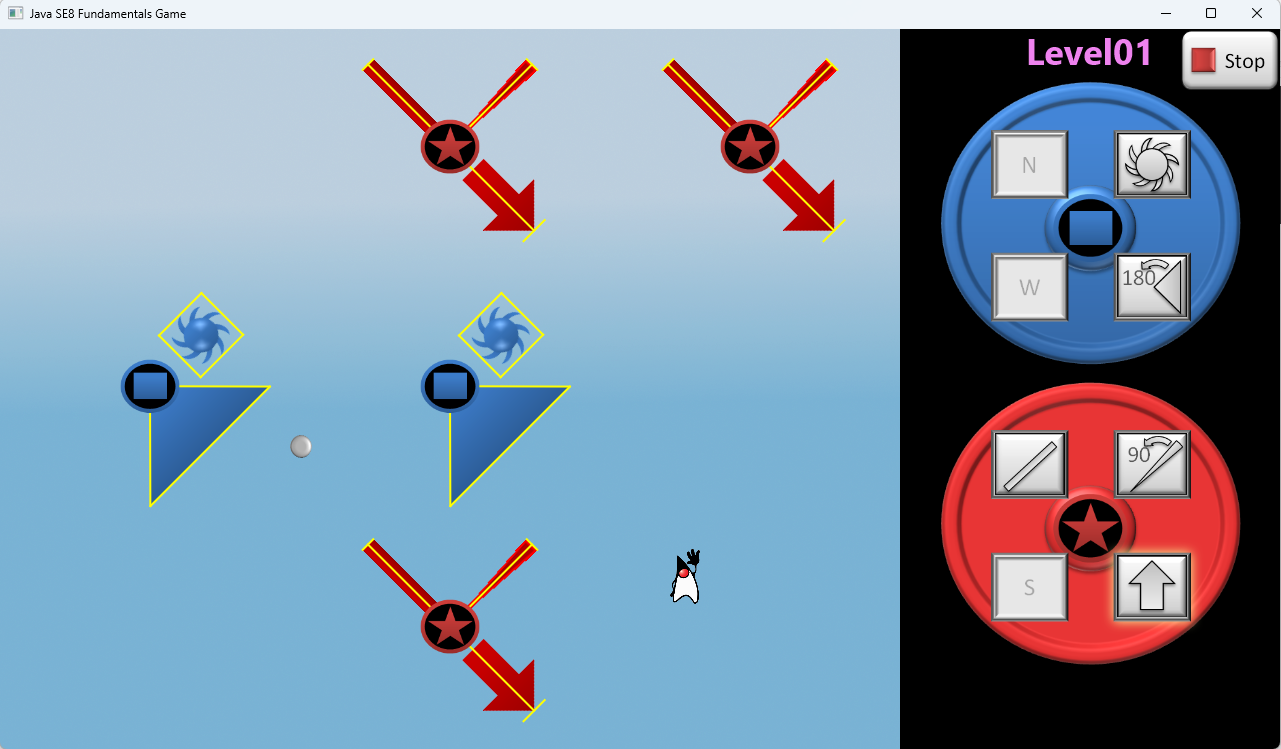


Рисунок 1.5 – Версия игры Java Puzzle Ball 10-16-13

Функции:

* Геометрия уровня
* Инструкции по созданию уровней считываются из текстового файла

21.11.2013

Уровней стало больше, теперь выбор уровня реализован в виде всплывающего списка.

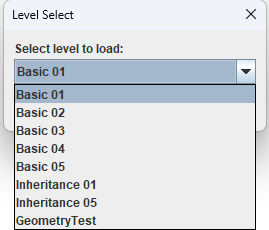


Рисунок 1.6 – Вспылывающее меню для выбора

уровней в виде всплывающего списка

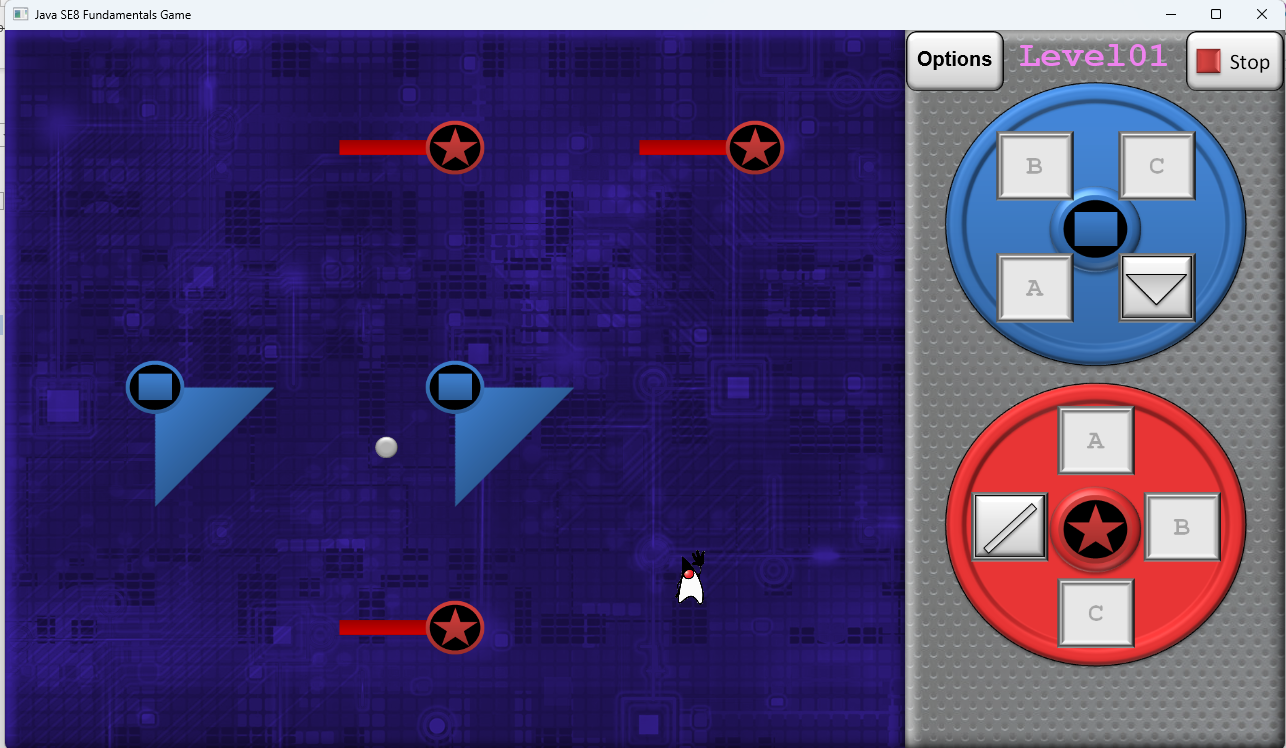


Рисунок 1.7 – Версия игры Java Puzzle Ball 11-21-13

Функции:

* Уровни можно выбирать с помощью кнопки Options
* Новые красивые фоновые изображения
* Дополнительные уровни
* Области теперь называются ABCD вместо NESW

# Упражнения из занятия 2-2 Задачи программы

Упражнение 1.

Установите точку останова в строке 5. Запустите программу в обычном режиме.

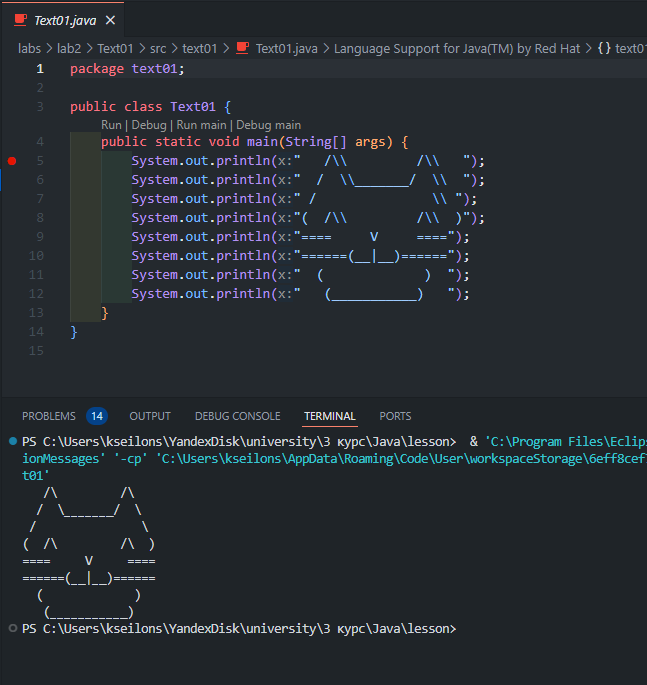


Рисунок 2.1– Запуск программы в обычном режиме

Запустите программу с отладчиком.

* Должно появиться окно "Output".
* Нажмите "Step Over", чтобы перейти на следующую строку.

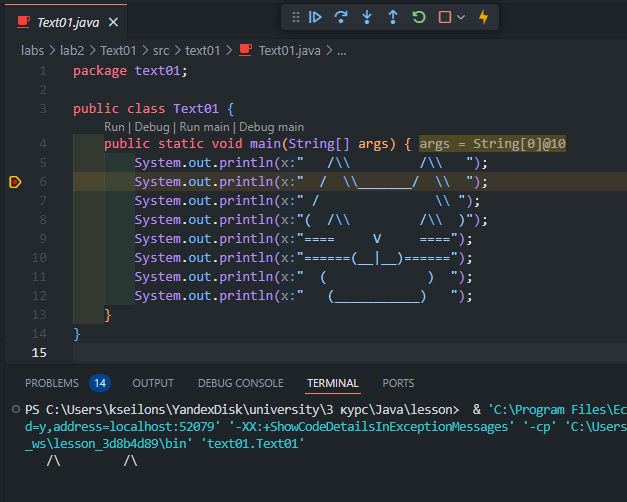


Рисунок 2.2– Запуск программы в debug mode до нажатия на “ Step Over ”

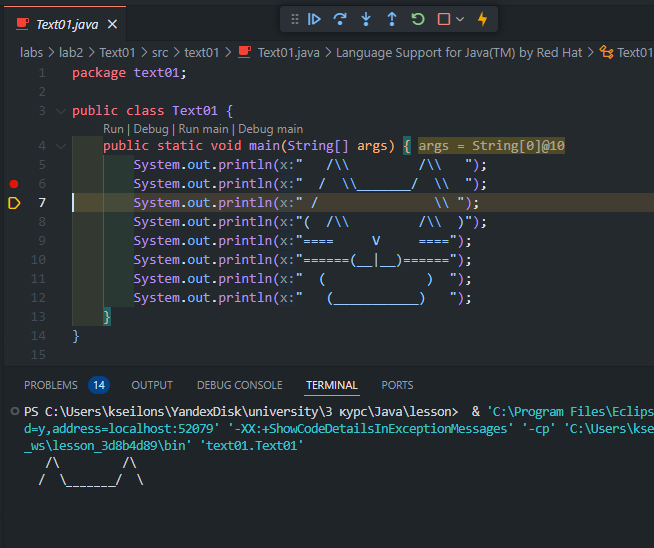


Рисунок 2.3– Запуск программы в debug mode после нажатия на “ Step Over ”

* Измените код, чтобы первые три оператора print отображались в строке 5. (Это называется "удаление пробелов".)

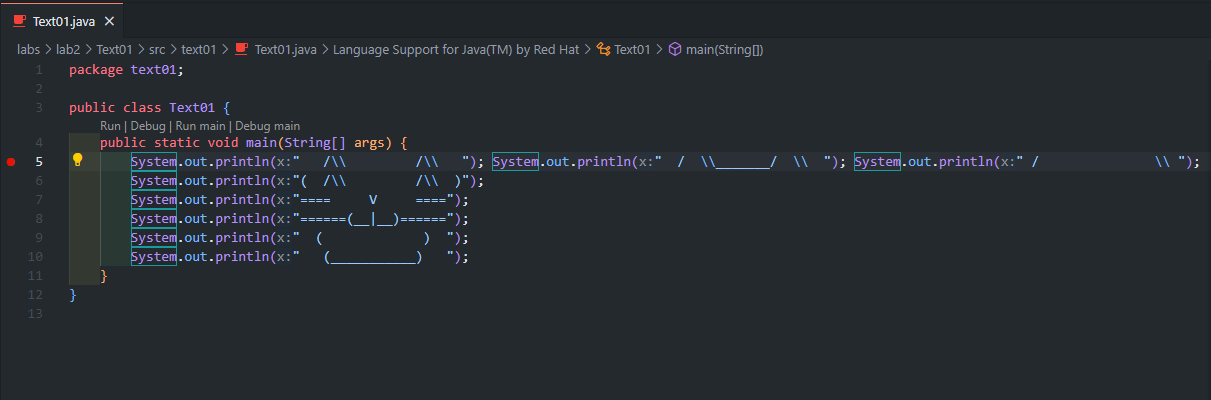


Рисунок 2.4– Изменение кода: отступы первых трех операторов print в строке 5 убраны

* Запустите программу с отладчиком.
  + Должно появиться окно "Output".
  + Нажимайте кнопку "Step Over Expression", чтобы перейти к каждой следующей строке.

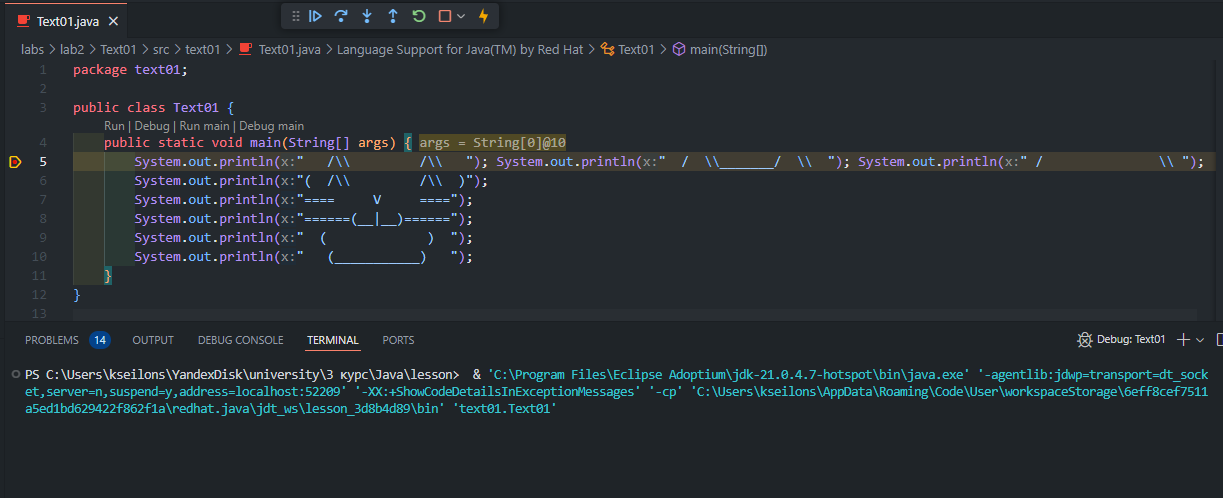


Рисунок 2.5– Запуск программы в debug mode перед нажатием на “ Step Out ”

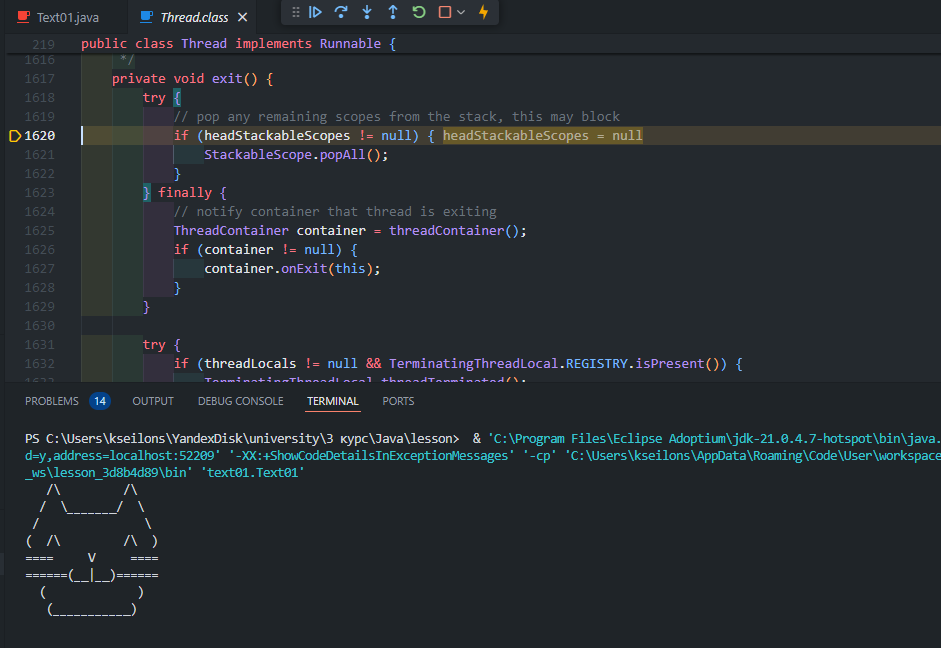


Рисунок 2.6– Запуск программы в debug mode после нажатия на “ Step Out ”

* Наблюдайте за тем, как появляется изображение кошки.
* Попробуйте удалить точку с запятой во время отладки программы.

Если удалить точку с запятой во время отладки программы ничего не произойдет, это не повлияет на текущую отладку. Чтобы изменения вступили в силу, нужно перекомпилировать программу. Если мы удалим точку с запятой перед запуском, программа не скомпилируется.

Упражнение 2.

Попробуйте исправить эту программу:

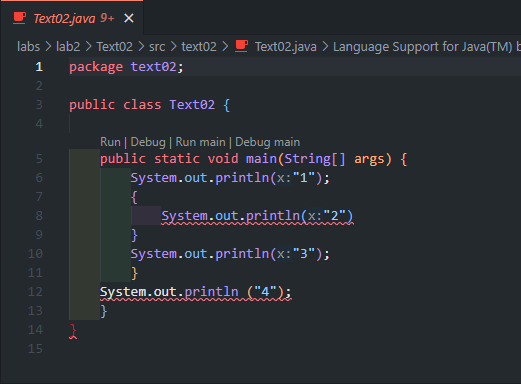


Рисунок 2.7– Неисправная программа из проекта Text02

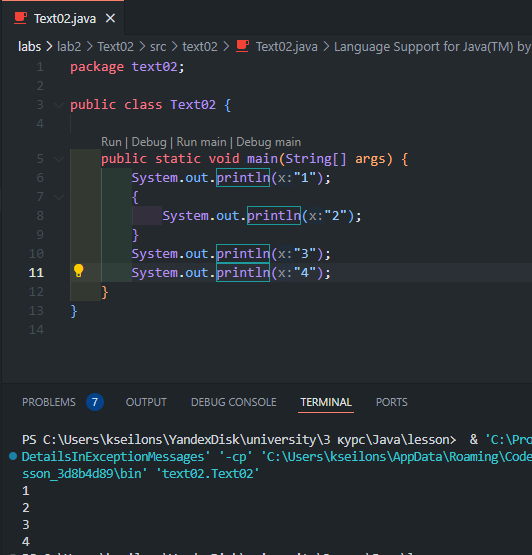


Рисунок 2.8– Исправленая программа из проекта Text02

Упражнение 3.

Установите точку останова в строке 11.

Запустите программу с отладчиком.

* Убедитесь, что выбрано окно "Output".
* Нажмите "Step Over", чтобы перейти на следующую строку

Введите drawLegs(); в строке 19 и запустите программу в режиме отладки.

* Где можно добавить точку останова, чтобы увидеть, как рисуются отдельные линии лап?
* Какой получится результат, если закомментировать строки?

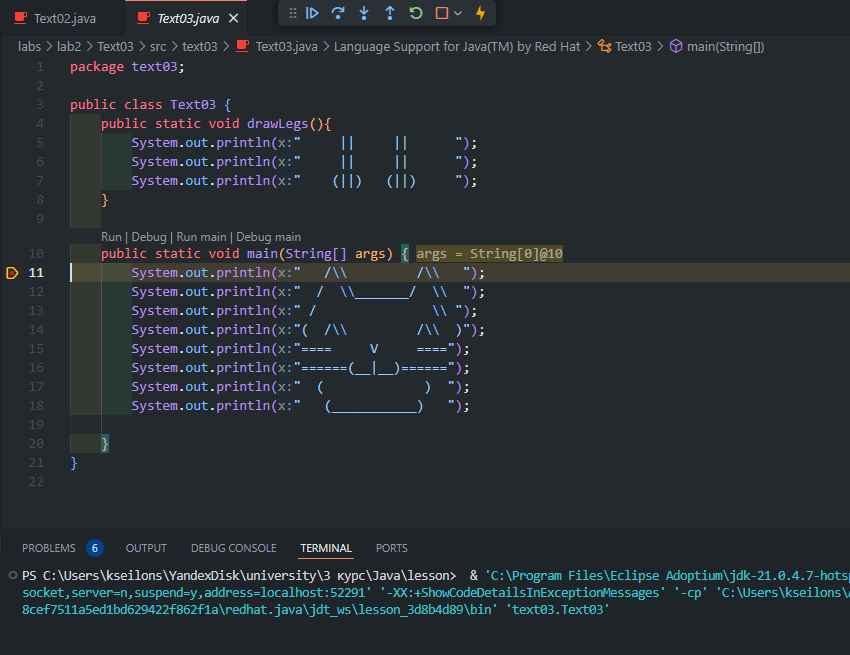


Рисунок 2.9– Запуск программы Text03

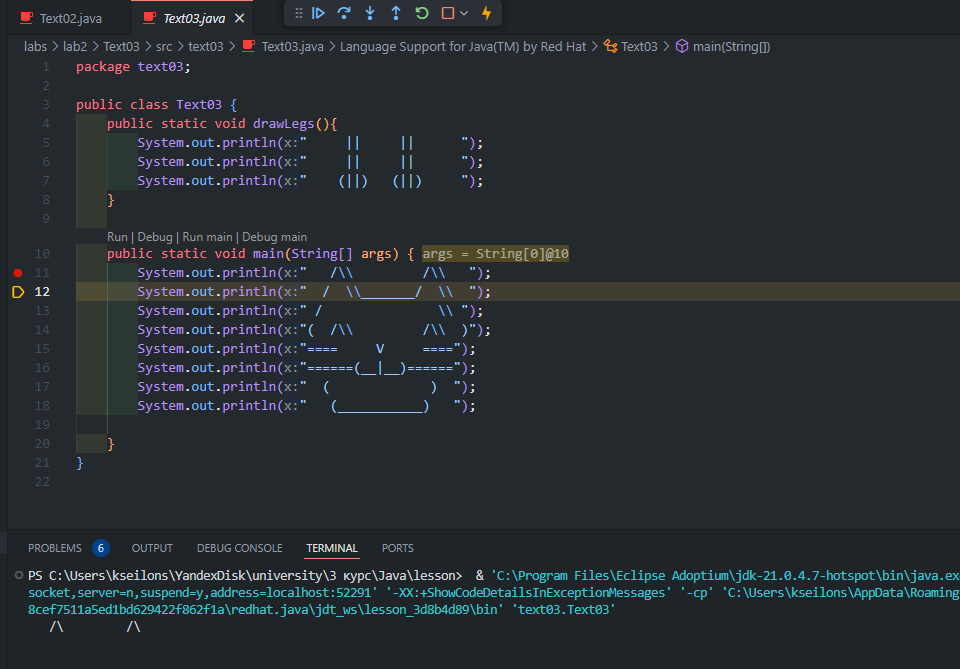


Рисунок 2.10– Запуск программы Text03 нажали на кнопку “Step over”

Чтобы увидеть, как рисуются лапы, точку останова нужно установить в пятой строчке:

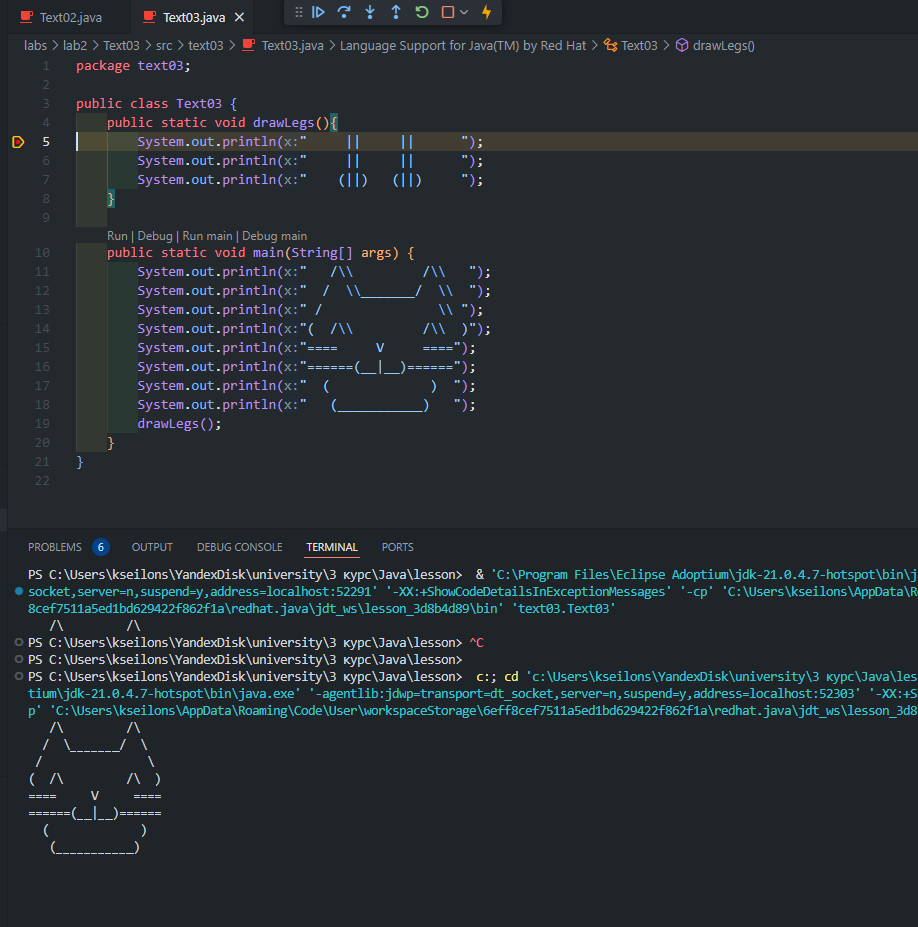


Рисунок 2.11– Запуск программы Text03 точка останова на лапах кота

Если закомментировать строки с выводом мордочки кота, например, то она будет игнорироваться:

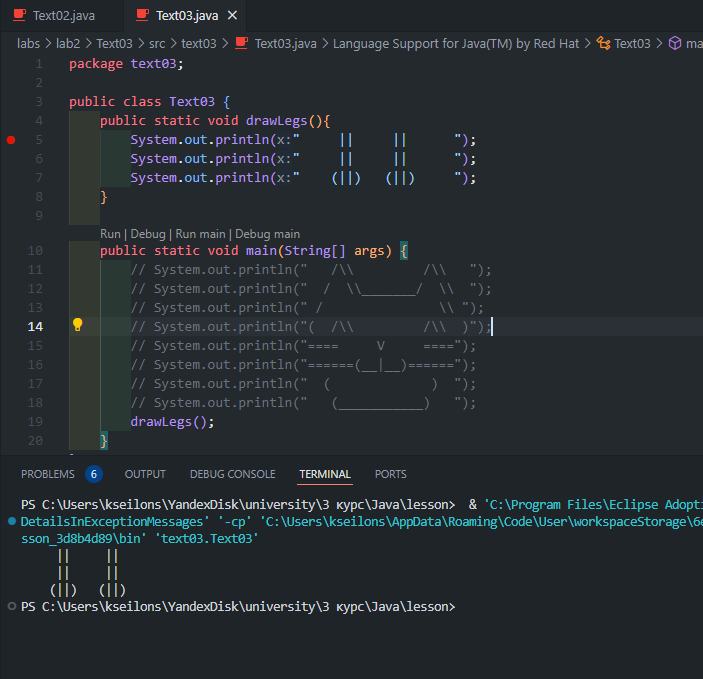


Рисунок 2.12– Запуск программы Text03 кот без мордочки

# Упражнения – Раздел 2

Проблема 1: Простая ASCII-графика

Воспроизвести приведенное выше улыбающееся лицо, используя 8 операторов вывода. Помимо пробела ваш рисунок будет основан на одном символе, таком как X или #.

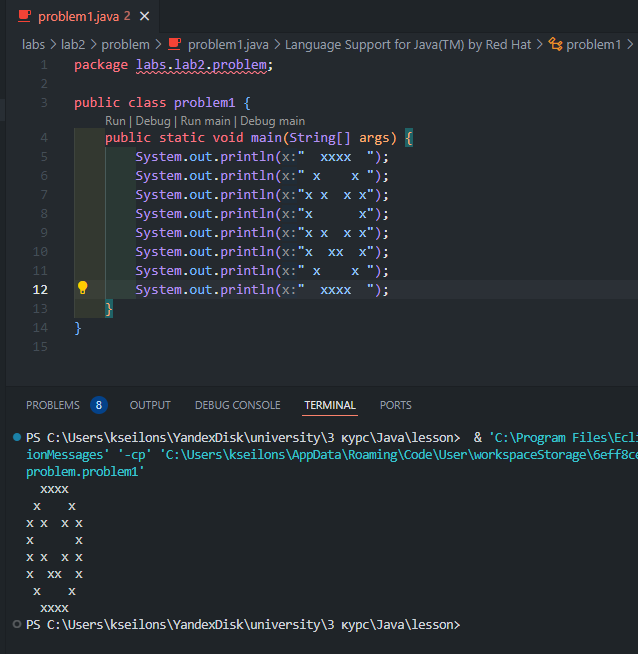


Рисунок 3.1– Улыбающийся смайлик

Проблема 2: Исходная ASCII-графика

Создать собственную оригинальную ASCII-графику, используя операторы вывода. С помощью комментариев опишите, что показано на изображении.

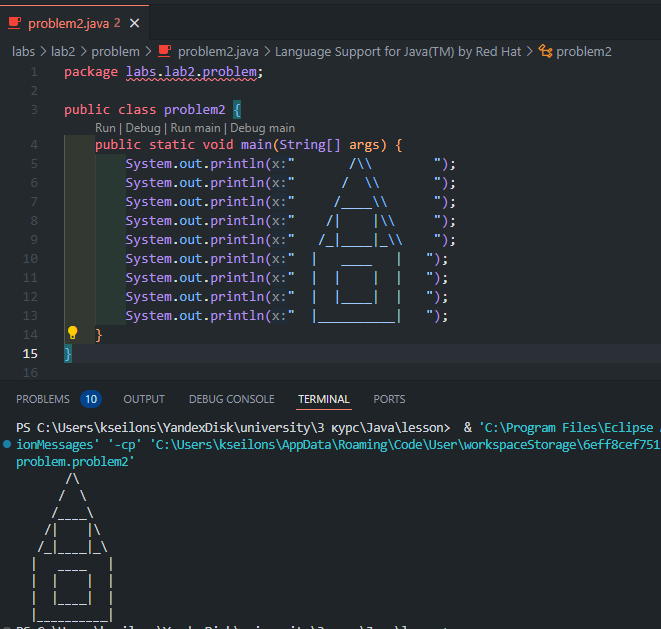


Рисунок 3.1– Дом нарисованный с помощью ASCII-графики

Проблема 3: The Snake Box Factory

Прочтите сценарий в обзоре и подумайте, какие объекты можно смоделировать в рамках создания программного решения. Определите 3 объекта в этом сценарии (помните, объекты могут быть материальными или абстрактными). Для каждого объекта перечислите 3 свойства и 3 варианта поведения.

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Box | Order | Deliveryman |
| Свойства объекта | size | orderNumber | deliveryAreaId |
| material | orderStatus | energy |
| id | id | id |
| Варианты поведения | pack | create | move |
| take | updateStatus | interactWithOtherSnakes |
| unpack | cancel | deliver |