Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет электроники и вычислительной техники

Кафедра «Программное обеспечение автоматизированных систем»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе**

по дисциплине «Объектно-ориентированный анализ и программирование»

на тему: «Проектирование программы с использованием объектно-ориентированного подхода»

(индивидуальное задание – вариант №01\_01)

Студент: Марченко С.С.

Группа: ПрИн-367

Работа зачтена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ « »  июня  2024 г.

Руководитель проекта, нормоконтроллер \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Литовкин Д.В.

Волгоград 2024 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет электроники и вычислительной техники

Направление 09.03.04 «Программная инженерия»   
Кафедра «Программное обеспечение автоматизированных систем»

Дисциплина «Объектно-ориентированный анализ и программирование»

Утверждаю

Зав. кафедрой \_\_\_\_\_\_\_\_\_\_ Орлова Ю.А.

**Задание**

**на курсовую работу**

Студент: Шашков Д. А.

Группа: ПрИн-367

1. Тема: «Проектированиепрограммы с использованием объектно-ориентированного подхода» (индивидуальное задание – вариант №01\_01)

Утверждена приказом от « » января 2024 г. № -ст

2. Срок представления работы к защите « 04 »   июня  2021 г.

3. Содержание пояснительной записки:

формулировка задания, требования к программе, структура программы, типовые процессы в программе, человеко-машинное взаимодействие, код программы и модульных тестов

4. Перечень графического материала:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Дата выдачи задания « » февраля 2024 г.

Руководитель проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Литовкин Д.В.

Задание принял к исполнению: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Марченко С.С.

« » февраля 2024 г.

**1 Формулировка задания**

Игра "Коза и капуста".

Правила игры:

- имеется загон NxM клеток, в котором находится коза, капуста, ящики и стены (располагаются внутри клеток);

- ящики и стены, возможно, образуют непроходимый лабиринт;

- цель козы — добраться до капусты, двигая ящики;

- коза может двигать только один ящик от себя или на себя;

- количество шагов козы ограничено.

**Подвариант 1:** Необходимо предусмотреть в программе **точки расширения**, используя которые можно реализовать вариативную часть программы (в дополнение к базовой функциональности).

**Вариативность:** на поле могут появляться объекты, которые могут или не могут быть передвинуты козой. Кроме того, неперемещаемые объекты можно активировать, применив ранее собранные объекты (неперемещаемые объекты). Результатом активации является изменение положения существующих объектов, появление новых объектов, уничтожение имеющихся.

**НЕ изменяя** ранее созданные классы, а используя **точки расширения, реализовать:** телепортатор, который активируется ключом.

# 2 Нефункциональные требования

1. Программа должна быть реализована на языке Java SE 17 с использованием стандартных библиотек, в том числе, библиотеки Swing.
2. Форматирование исходного кода программы должно соответствовать Java Code Conventions, September 12, 1997.

# 3 Первая итерация разработки

**3.1 Формулировка упрощённого варианта задания**

Игра "Коза и капуста".

Правила игры:

-         имеется загон NxM клеток, в котором находится коза, капуста, ящики и стены (располагаются внутри клеток);

-         ящики и стены, возможно, образуют непроходимый лабиринт;

-         цель козы — добраться до капусты, двигая ящики;

-         коза может двигать только один ящик от себя или на себя;

-         количество шагов козы ограничено.

**3.2 Функциональные требования (сценарии)**

**1) Сценарий** «Играть»

1. **По указанию пользователя,** Игра стартует.
2. **По указанию** Игры, Поле создаёт Ячейки и формирует из них себя. При этом Поле узнает позицию для Капусты у Загона.
3. **По указанию** Игры, создаетя Загон, после чего Загон создаёт и размещает на Поле Стены, Ящики, Капусту и Козу.
4. **В ответ на запрос** Игры, Поле **сообщает** о Козе, которая находится на нём.
5. **Делать** 
   1. **По указанию** **пользователя,** Коза двигает Ящик от себя или на себя, расходуя Шаги.
   2. **По указанию** **пользователя,** Коза перемещается на соседнюю Ячейку, расходуя Шаги.
   3. **Если** Коза находится в клетке с Капустой, она съедает Капусту
   4. Коза сообщает Игре о том, что съела Капусту
   5. Игра завершается победой пользователя

# Пока у Козы есть доступное количество шагов и Игра не завершена

1. **Сценарий завершается**.

**2) Дочерний сценарий «**Загон создаёт и размещает на Поле Стены, Ящики, Капусту и Козу**»**

1. Загон создаёт Козу.
2. Загон помещает на Поле созданную Козу.
3. Загон создаёт Капусту и помещает ее на Поле.
4. Загон создаёт и расставляет последовательности Стен по периметру Поля.
5. **Сценарий завершается**.

**3) Дочерний сценарий** «Коза перемещается на соседнюю Ячейку расходуя Шаги»

1. **В ответ на запрос** Козы, Загон **сообщает**, что Препятствия нет в направлении движения Козы
2. **Если** взята Коробка, Загон **сообщает** о том, что в соседней для Коробки Клетки в направлении движения нет Стены или Коробки
3. Если взята Коробка, Загон **извлекает** Коробку из соседней Клетки
4. **В ответ на запрос** Козы, соседняя **Клетка** в направлении движения Козы сообщает, свободна ли она
5. Коза решает, что может переместиться в соседнюю Клетку
6. **По указанию** Козы, шаг, необходимый для движения в направлении, изымается из общего доступного количества шагов
7. **По указанию** Козы, Клетка с Козой извлекает ее из себя
8. Соседняя Клетка помещает Козу в себя, так как Клетка пуста
9. Если была взята Коробка, то Коза перемещает Коробку в соседнюю свободную по направлению движения (к себе или от себя) Клетку
10. **Сценарий завершается**.

3.1) **Альтернативный сценарий** «Между соседней Клеткой и текущей находится Стена». Сценарий выполняется с п. 1 сценария 3

1. **В ответ на запрос** Козы, Соседняя Клетка сообщает, что Стена имеется в направлении движения Козы

2. Коза решает, что не может переместиться в соседнюю Клетку

3. **Сценарий завершается**

3.2) **Альтернативный сценарий** «В соседней для взятой Коробки Клетке находится Коробка или Стена». Сценарий выполняется с п. 1 сценария 2

1. **По указанию** Козы, соседняя Клетка сообщает, что в ней есть Коробка или Стена

2. Коза решает, что не может переместиться в соседнюю Клетку

3. **Сценарий завершается**

3.3) **Альтернативный сценарий** «У Козы недостаточно шагов для совершения хода». Сценарий выполняется с п. 6 сценария 3

1. Коза **сообщает**, что у нее недостаточно шагов для совершения хода

2. **Сценарий завершается**

**4) Альтернативный сценарий** «Перемещение Козы с Ящиком вдоль направления». Сценарий **выполняется с** п. 2 сценария 3.2

1. **По указанию** Пользователя, Коза сообщает наличие ящика.

2. Если направление, **указанное** Пользователем, вдоль, Коза запрашивает наличие препятствия у Клетки, в которую сместится ящик и Коза.

3. Если Клетка свободна, коза переходит в нее, коробка следом за ней.

3. **Сценарий завершается**

**5) Альтернативный сценарий** «Перемещение Козы с Ящиком не вдоль направления». Сценарий **выполняется с** п. 2 сценария 3.2

1. **По указанию** Пользователя, Коза сообщает наличие ящика.

2. Если направление, **указанное** Пользователем, не вдоль, Коза сообщает, что передвижение невозможно.

3. **Сценарий завершается**

**6) Дочерний сценарий** «Игра считает победителем Козу, забравшую Капусту»

1. Игра **запрашивает** у Поля Капусту.
2. Поле **запрашивает** Козу.
3. Поле **сообщает** Игре, что капусты нет на поле.
4. Игра считает победителем Козу, т.к. она съела Капусту.
5. **Сценарий завершается.**

**6.1) Альтернативный сценарий** «Поражение, так как Коза израсходовала все шаги, не достигнув капусту». Сценарий **выполняется с** п. 3 сценария 5

1. Игра **запрашивает** у Поля Капусту.
2. Поле **запрашивает** Козу.
3. Поле **сообщает** Игре, что капуста есть на поле.
4. Коза сообщает, что у нее нет шагов на совершение действия
5. Игра считает проигравшей Козу, т.к. она не забрала Капусту и не может двигаться.
6. **Сценарий завершается.**

**7) Альтернативный сценарий** «Досрочное завершение игры». Сценарий **выполняется в любой точке** главного сценария

1. **По указанию пользователя,** программа завершается без определения победителя.
2. **Сценарий завершается.**

**3.3 Словарь предметной области**

**Игра** – знает о Поле и Загоне. Игра инициирует создание Поля и расстановку всех сущностей на нем с помощью Загона. Игра определяет окончание игры.

**Поле** – прямоугольная область, состоящая из Ячеек. Между Ячейками может располагаться Стена. По границе Поля также могут располагаться Стены. Знает о Козе и Капусте, находящейся на Поле.

**Ячейка** – квадратная область Поля. Знает о четырёх соседних Ячейках и граничащих с ней Стенах. На ней может располагаться Коза, или Капуста, либо сразу оба.

**Стена** – непроходимое Препятствие для Козы, располагающееся между Ячейками.

**Загон** – умеет создавать Стены, Козу и Капусту и размещать их на Поле в Ячейках. Позиции этих сущностей Загон определяет самостоятельно. Он знает расположение Капусты.

**Коза** – умеет *перемещаться* и двигать одну коробку на себя или от себя. Коза перемещается в соседнюю Ячейку, затрачивая Шаги, но не может пройти через Стену.

**Капуста** – разновидность объекта взаимодействия. Когда в Ячейку, где находится Капуста, становится Коза, Капуста уничтожается и сообщает о конце игры.

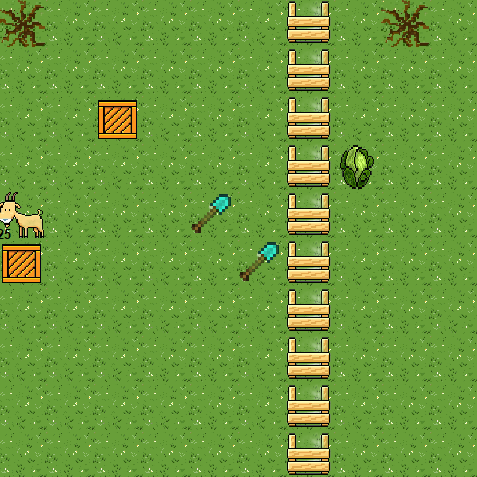
**Ящик** – разновидность объекта взаимодействия. Непреодолимое препятствие для Козы. Когда рядом Коза, она может взять Ящик и потянуть вперед или назад.

**3.4 Структура программы на уровне классов**

**3.5 Типовые процессы в программе**

**3.6 Человеко-машинное взаимодействие**

Общий вид главного экрана программы представлен ниже. На нём располагается игровой загон, на котором изображена Коза и Капуста, коробки и стены.



Управление Козой пользователь осуществляет с помощью клавиатуры.

Стрелка вверх – движение вверх.

Стрелка вниз – движение вниз.

Стрелка влево – движение влево.

Стрелка вправо – движение вправо.

Пробел – взаимодействие с объектами

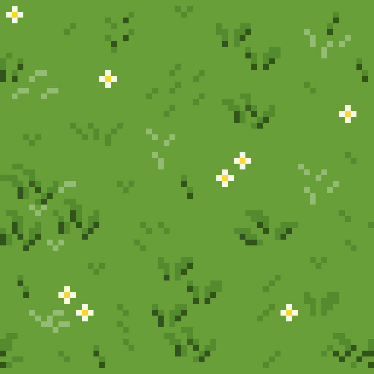
Клавиша Ctrl удерживается и при совместном применении с любой из клавиш стрелок Коза тянет коробку к себе



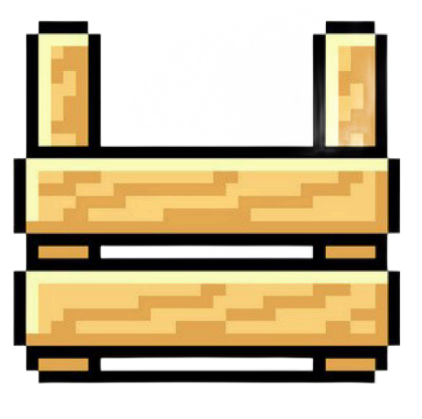
Изображение Козы

В левом нижнем углу от Козы располагается надпись, информирующая о кол-ве шагов, доступных Козе, на данный момент.

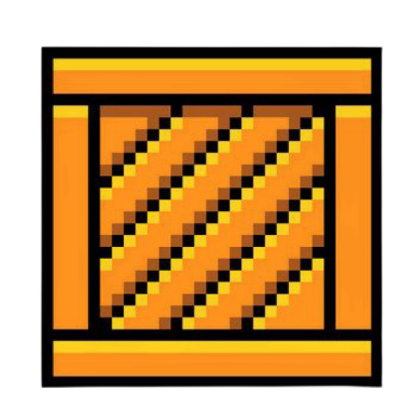
Каждая клетка имеет свое фоновое изображение



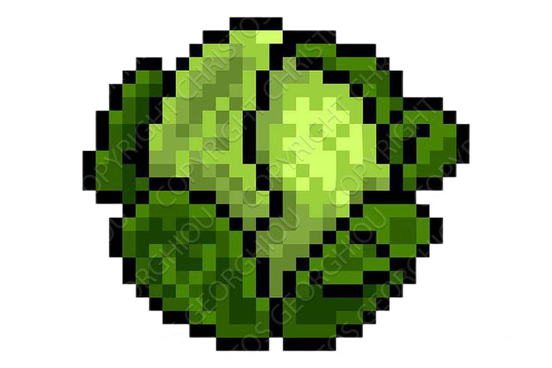
Также имеются изображения для Коробки, Стен и Капусты



Изображение Стены



Изображение Коробки



Изображение Капусты

**3.7 Реализация ключевых классов**

**3.8 Реализация ключевых тестовых случаев**

# 4 Вторая итерация разработки

**4.1 Функциональные требования (сценарии)**

1. **Сценарий** «Коза использует Яму с помощью лопаты»

1. В ответ на запрос Козы, Яма запрашивает количество Лопат у Козы

2. Если Лопаты есть, Лопата изымается у Козы

3. Яма перемещает Козу рядом с другой Ямой, связанной с текущей

* 1. **Альтернативный Сценарий** «Коза использует Яму без Лопаты с собой»

1. В ответ на запрос Козы, Яма запрашивает количество Лопат у Козы
2. Если Лопаты нет, взаимодейтсвие Козы с Ямой заканчивается

**4.2 Словарь предметной области**

**Игра** – знает о Поле и Загоне. Игра инициирует создание Поля и расстановку всех сущностей на нем с помощью Загона. Игра определяет окончание игры.

**Поле** – прямоугольная область, состоящая из Ячеек. Между Ячейками может располагаться Стена. По границе Поля также могут располагаться Стены. Знает о Козе и Капусте, находящейся на Поле.

**Ячейка** – квадратная область Поля. Знает о четырёх соседних Ячейках и граничащих с ней Стенах. На ней может располагаться Коза, или Капуста, либо сразу оба.

**Стена** – непроходимое Препятствие для Козы, располагающееся между Ячейками.

**Загон** – умеет создавать Стены, Козу и Капусту и размещать их на Поле в Ячейках. Позиции этих сущностей Загон определяет самостоятельно. Он знает расположение Капусты.

**Коза** – умеет *перемещаться* и двигать одну коробку на себя или от себя. Коза перемещается в соседнюю Ячейку, затрачивая Шаги, но не может пройти через Стену.

**Капуста** – разновидность объекта взаимодействия. Когда в Ячейку, где находится Капуста, становится Коза, Капуста уничтожается и сообщает о конце игры.

**Ящик** – разновидность объекта взаимодействия. Непреодолимое препятствие для Козы. Когда рядом Коза, она может взять Ящик и потянуть вперед или назад.

**Ключ (в документе Лопата)** – разновидность объекта взаимодействия. Непреодолимое препятствие для Козы. Когда рядом Коза, она может подобрать ключ с поля себе.

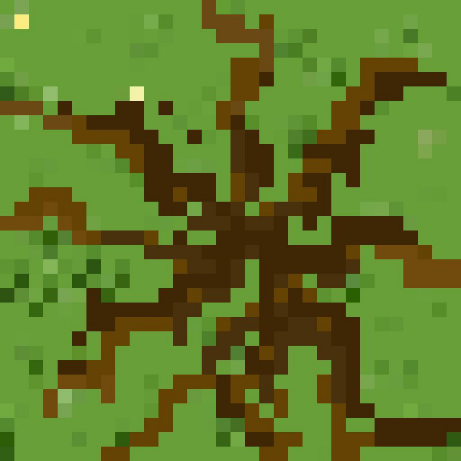
**Телепорт (в документе Яма)** – разновидность объекта взаимодействия. Непреодолимое препятствие для Козы. Когда рядом Коза, она может Переместиться от одного телепорта к другому ценой одного Ключа(Лопаты).

**4.3 Структура программы на уровне классов**

**4.4 Типовые процессы в программе**

**4.5 Человеко-машинное взаимодействие**

Есть на поле ямы и лопаты для перемещения по ним



Изображение ямы



Изображение лопаты