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 и реализация программы с использованием объектно-ориентированного подхода»

(индивидуальное задание – вариант №06\_01)

Студент: Олейников М.А.

Группа: ПрИн-366

Работа зачтена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «      » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20      г.

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

Волгоград 2021 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: Олейников М.А.

Группа: ПрИн-366

1. Тема: «Проектирование и реализация программы с использованием объектно-ориентированного подхода» (индивидуальное задание – вариант №06\_01)

Утверждена приказом от «24» января 2021г. № 101-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 «12» февраля 2021 г.

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

Задание принял к исполнению: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Олейников М.А.

«12» февраля 2021 г.

**Содержание**

1 Формулировка задания 4

2 Нефункциональные требования 5

3 Первая итерация разработки 6

3.1 Формулировка упрощенного варианта задания 6

3.2 Функциональные требования (сценарии) 6

3.3 Словарь предметной области 10

# 1 Формулировка задания

Правила игры «Пауки»:

- имеется паутина N×N;

- на паутине живут К пауков, один из которых управляется человеком;

-        паук движется вдоль нити паутины; игра ведется **пошагово**;

-         жизнь паука – L единиц; на каждом шаге жизнь уменьшается на 1;

-         случайным образом в паутину могут попадать насекомые;

-         насекомое с некоторой вероятностью может вырваться из паутины;

- при съедании насекомого жизнь паука восстанавливается на некоторую величину;

- цель игры – паук не должен умереть как можно дольше; паук умирает, если его жизнь уменьшается до нуля;

- минимальные требования к интеллекту компьютерного игрока: паук движется к ближайшему насекомому, до которого он может добежать. Если таких насекомых несколько, то выбор осуществляется случайным способом.

2. **Дополнительные требования:**
3. - отображать текущую жизнь каждого паука;
4. - отображать факт съедения насекомого;
5. - отображать кол-во пройденных шагов.
6. **Подвариант 1:**

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

**Вариативность**: создание разновидностей насекомых, имеющих различный внешний вид, размер, способ попадания на паутину и возможность вырваться из паутины. Так же вид насекомого по-разному влияет на жизнь паука.

Используя точки расширения, **реализовать** (ранее созданные классы **НЕ изменять**):

- в паутину могут попадать "кузнечики", "мухи", "комары" и "осы". Чем больше насекомое, тем реже оно попадает на паутину;

1. -  размер насекомого определяет, с какой вероятностью оно может вырваться из паутины;
2. -  размер насекомого определяет, насколько оно восстанавливает жизнь паука; "осы" иногда могут кусаться, тогда жизнь паука уменьшается;
3. -  кузнечик может прыгать по паутине (вырываться и попадать снова, но в другой позиции), но с каждым разом вероятность вырваться из паутины уменьшается.

# 2 Нефункциональные требования

1. Программа должна быть реализована на языке Java SE 12 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

Правила игры «Сломанный робот»:

* игра происходит на прямоугольном поле W × H, разделённом на ячейки;
* на поле имеется лабиринт с одним выходом, в котором разбросаны батарейки;
* лабиринт состоит из стен, которые располагаются между ячейками поля;
* на поле имеются два робота, начальная позиция роботов произвольная;
* при перемещении роботы расходуют заряд собственной батарейки, которой хватает на M шагов;
* цель робота – выйти из лабиринта и оставить своего соперника в лабиринте;
* оба робота управляются игроками;
* необходимо анимировать расход/приход заряда.

### 3.2 Функциональные требования (сценарии)

*кому/что поручить*

**Сценарии игры «пауки».**

1. **Главный успешный сценарий** “Играть”
2. **По указанию пользователя**, *Игра* стартует.
3. *Игра* создаёт *Природу* – класс необходимый для хранения знаний о *Паутине*, *Пауках* и *Насекомых*, а также запускающий процесс создания всего перечисленного.
4. *Природа* запускает процесс создания *Паутины* – поля для игры из *Зон* *Паутины* (клеток на поле).
5. **По указанию** *Игры* на *Паутине*, в случайной *Зоне*, *Природа* вызывает процесс создания *Паука* игрока с определённым количеством очков здоровья.
6. **По указанию** *Игры* в случайных пустых *Зонах* *Паутины* *Природа* вызывает процесс создания некоторого количества *Пауков* ботов – соперников игрока с определённым количеством очков здоровья.
7. **Делать.**
   1. **По указанию** *Игры* во всех пустых *Зонах* *Паутины* с определённой вероятностью *Природа* вызывает процесс создания *Насекомых*, которыми могут питаться *Пауки*, наступив на *Зону* с ними, для увеличения очков здоровья.
   2. **По указанию** *Игры* с определённой вероятностью *Природа* вызывает процесс попадания *Насекомого* в случайную пустую *Зону Паутины*.
   3. **По указанию** *Игры* во всех *Зонах* *Паутины* с *Насекомыми* с определённой вероятностью *Природа* вызывает процесс выхода *Насекомого* из *Зоны Паутины,* в которой оно находится.
   4. Игрок передаёт *Игре* *Направление*, по которому хочет перевести своего *Паука*, затрачивая при этом 1 очко здоровья *Паука*.
   5. *Паутина* пытается перевести *Паука* игрока, в том *Направлении*, которое ему передала *Игра*.
   6. Если здоровье *Паука* игрока равно 0 – природа уничтожает паука.
   7. **Делать**
      1. *Игра* спрашивает у *Паука* бота *Направление*, в котором *Паук* бота хочет перейти.
      2. *Паук* бота определяет направление, в котором ему выгоднее переместиться.
      3. *Паук* бота передаёт *Направление*, которое посчитал наиболее выгодным для своего перемещения - *Паутине*, затрачивая при этом 1 очко здоровья, если перемещение было успешно совершено.
      4. *Паутина* пытается перевести *Паука* бота, в том *Направлении*, которое ему передал *Паук*.
      5. Если здоровье *Паука* бота равно 0 – природа уничтожает паука.

**Для** всех *Пауков* ботов

**Пока** *Паук* игрока не достиг нуля очков здоровья или игрок не завершил игру принудительно.

1. **Сценарий завершается**
2. **Дочерний успешный сценарий** – «*Паутина* пытается перевести *Паука* в определённом *Направлении*» (успешно).
3. **По запросу** *Паутины* *Паук* сообщает ей в какой *Зоне* он находится.
4. *Паутина* определяет, существует ли *Зона* в том *Направлении*, в котором паук хочет переместиться.
5. Т.к. *Зона* существует, *Паутина* проверяет, находится ли *Паук* в полученной *Зоне*.
6. Т.к. в полученной *Зоне* нет *Паука,* *Паутина* передаёт *Пауку* эту *Зону* для перемещения и возбуждает процесс попытки перемещения *Паука*.
7. **Сценарий завершается**

2.1) **Альтернативный сценарий** – «*Паутина* пытается перевести *Паука* в определённом *Направлении*» (безуспешно, так как в заданном *Направлении* не существует *Зоны)*.

1. **Начинаем** с пункта 2 сценарий 2.

2. Т.к. *Зоны* не существует, *Паутина* не возбуждает процесс попытки перемещения *Паука*.

3. **Сценарий завершается**

1. **Дочерний успешный** **сценарий** – «Процесс попытки перемещения *Паука* в определённую *Зону»* (*Зона* пуста).
2. *Паук* спрашивает у новой *Зоны*, есть ли в *Зоне*, которую он переходит *Паук*.
3. Т.к. в *Зоне* нет *Паука, Паук* спрашивает у новой *Зоны*, есть ли в *Зоне*, которую он переходит *Насекомое*.
4. *Паук* освобождает свою предыдущую *Зону* и занимает новую перед этим замечая, есть ли в клетке *Насекомое*.
5. Т.к. *Зона* пуста **сценарий завершается.**
   1. **Альтернативный сценарий** – «Процесс перемещения *Паука* в определённую *Зону»* (в *Зоне* находится *Насекомое*).
6. Начинаем с пункта 3 сценария 3.
7. Т.к. в Зоне находится *Насекомое, Насекомое* уничтожается *Пауком*.
8. Здоровье *Паука* увеличивается на размер съеденного *Насекомого*.
9. **Сценарий завершается**
   1. **Альтернативный сценарий** – «Процесс перемещения *Паука* в определённую *Зону»* (в *Зоне* находится *Паук*).
10. Начинаем с пункта 1 сценария 3.
11. Т.к. в Зоне уже находится *Паук,* процесс перемещения паука в эту Зону считается безуспешным.
12. **Сценарий завершается**
13. **Дочерний успешный** **сценарий** – «Определение ботом направления, в котором его *Паук* будет передвигаться».
14. **По запросу** *Паука* *Паутина* возвращает *Зону,* где находится ближайшее *Насекомое* к *Пауку* бота.
15. Создать список возможных направлений для перемещения *Паука*.
16. **Делать**
    1. Если *Зона* ближайшего насекомого находится в данном *Направлении* – добавим его в список возможных направлений для перемещения *Паука*.

**Для всех** *Направлений* из множества {север, юг, запад, восток}

4. Вернуть случайное *Направление* списка возможных.

5. **Сценарий завершается**

1. **Дочерний успешный** **сценарий** – «Процесс попадания *Насекомого* в случайную пустую *Зону Паутины».*
2. Создаётся *Насекомое* с определённым размером.
3. *Насекомое* кладётся в заданную *Зону*.
4. **Сценарий завершается**
5. **Дочерний успешный** **сценарий** – «Процесс выхода *Насекомого* из *Зоны Паутины,* в которой оно находится*».*

Необходимо добавить по причине переосмысления работы.

1. *Насекомое* выходит из своей *Зоны.*
2. *Насекомое* уничтожается, так как у него нет связей.
3. **Сценарий завершается**

### 3.3 Словарь предметной области

*знает/умеет/предназначена*

**Паук** – персонаж, который принадлежит одному из игроков на поле, знает о своей Зоне.

**Паук бота (умный Паук)** – паук умеющий определять относительно выгодное Направление перемещения.

**Насекомое** – персонаж, который может быть съеден пауком, для увеличения очков здоровья, знает о своей зоне.

**Паутина** – поле для игры, знает обо всех Зонах.

**Зона паутины** – клетка на поле, знает о Паутине.

**Природа** – класс необходимый для создания насекомых и пауков, знает о Паутине.

**Игра** – это класс, который создаёт и использует другие классы для возможности проведения игры, знает о Паутине и Природе.

**Направление** – это enam, позволяет определить из какой Зоны, в какую требуется совершить перемещение.