Лабораторная работа 2

Тема: «Разработка консольной игры в парадигме ООП на python»

Цель: «Реализовать консольное приложение - игру с применением принципов объектно-ориентированного программирования (работа с классами, интерфейсами, принципами ООП)»

Содержание

Задача на самостоятельную реализацию

Реализовать программу для тестирования (работа с консолью и ООП). Реализовать структуру курсовой работы. Выполнить следующие 3 задания в рамках лабораторной работы и 4 задание в рамках курсового проекта.

Задача 1. Основные классы

Разработать и реализовать набор классов:

- Класс игрового поля
- Набор классов юнитов

Игровое поле является контейнером для объектов представляющим прямоугольную сетку. Основные требования к классу игрового поля:

- Создание поля произвольного размера
- Контроль максимального количества объектов на поле
- Возможность добавления и удаления объектов на поле
- Возможность копирования поля (включая объекты на нем)

Юнит является объектов, размещаемым на поля боя. Один юнит представляет собой отряд. Основные требования к классам юнитов:

- Все юниты должны иметь как минимум один общий интерфейс
- Реализованы 3 типа юнитов (например, пехота, лучники, конница)
- Реализованы 2 вида юнитов для каждого типа (например, для пехоты могут быть созданы мечники и копейщики)
- Юниты имеют характеристики, отражающие их основные атрибуты, такие как здоровье, броня, атака.
- Юнит имеет возможность перемещаться по карте

Пример игрового поля на рис. 1.

Рис. 1. Пример игрового поля для задания 1 22 на 22

Задача 2. Интерфейсы классов, взаимодействие классов, перегрузка операций

Разработать и реализовать набор классов:

- Класс базы
- Набор классов ландшафта карты
- Набор классов нейтральных объектов поля

Класс базы должен отвечать за создание юнитов, а также учитывать юнитов, относящихся к текущей базе. Основные требования к классу база:

- База должна размещаться на поле
- Методы для создания юнитов
- Учет юнитов, и реакция на их уничтожение и создание

- База должна обладать характеристиками такими, как здоровье, максимальное количество юнитов, которые могут быть одновременно созданы на базе, и т.д.

Набор классов ландшафта определяют вид поля. Основные требования к классам ландшафта:

Должно быть создано минимум 3 типа ландшафта

- Все классы ландшафта должны иметь как минимум один интерфейс
- Ландшафт должен влиять на юнитов (например, возможно пройти по клетке с определенным ландшафтом или запрет для атаки определенного типа юнитов)
- На каждой клетке поля должен быть определенный тип ландшафта

Набор классов нейтральных объектов представляют объекты, располагаемые на поле и с которыми могут взаимодействие юнитов. Основные требования к классам нейтральных объектов поля:

- Создано не менее 4 типов нейтральных объектов
- Взаимодействие юнитов с нейтральными объектами, должно быть реализовано в виде перегрузки операций
- Классы нейтральных объектов должны иметь как минимум один общий интерфейс

Пример карты после выполнения задания 2 на рис. 2.

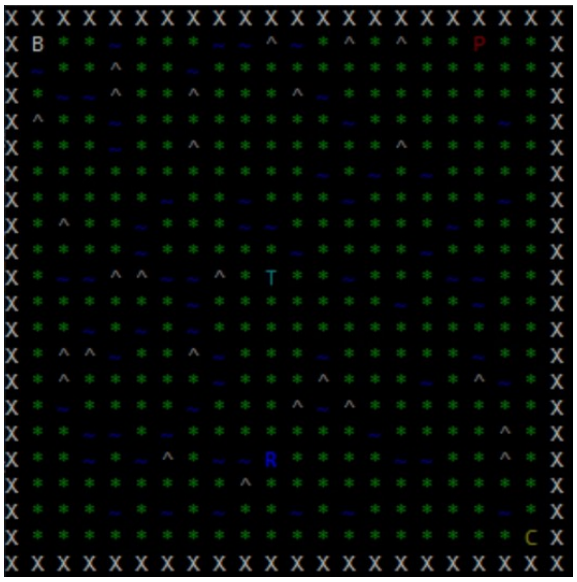

Рис. 2. Пример внешнего вида консольной карты, после выполнения задания 2

Задача 3. Взаимодействие пользователя с игрой

Разработать и реализовать набора классов для взаимодействия пользователя с юнитами и базой. Основные требования:

- Должен быть реализован функционал управления юнитами
- Должен быть реализован функционал управления базой

Пример выполнения программы по управлению базой в консоли представлен на рис. 3.

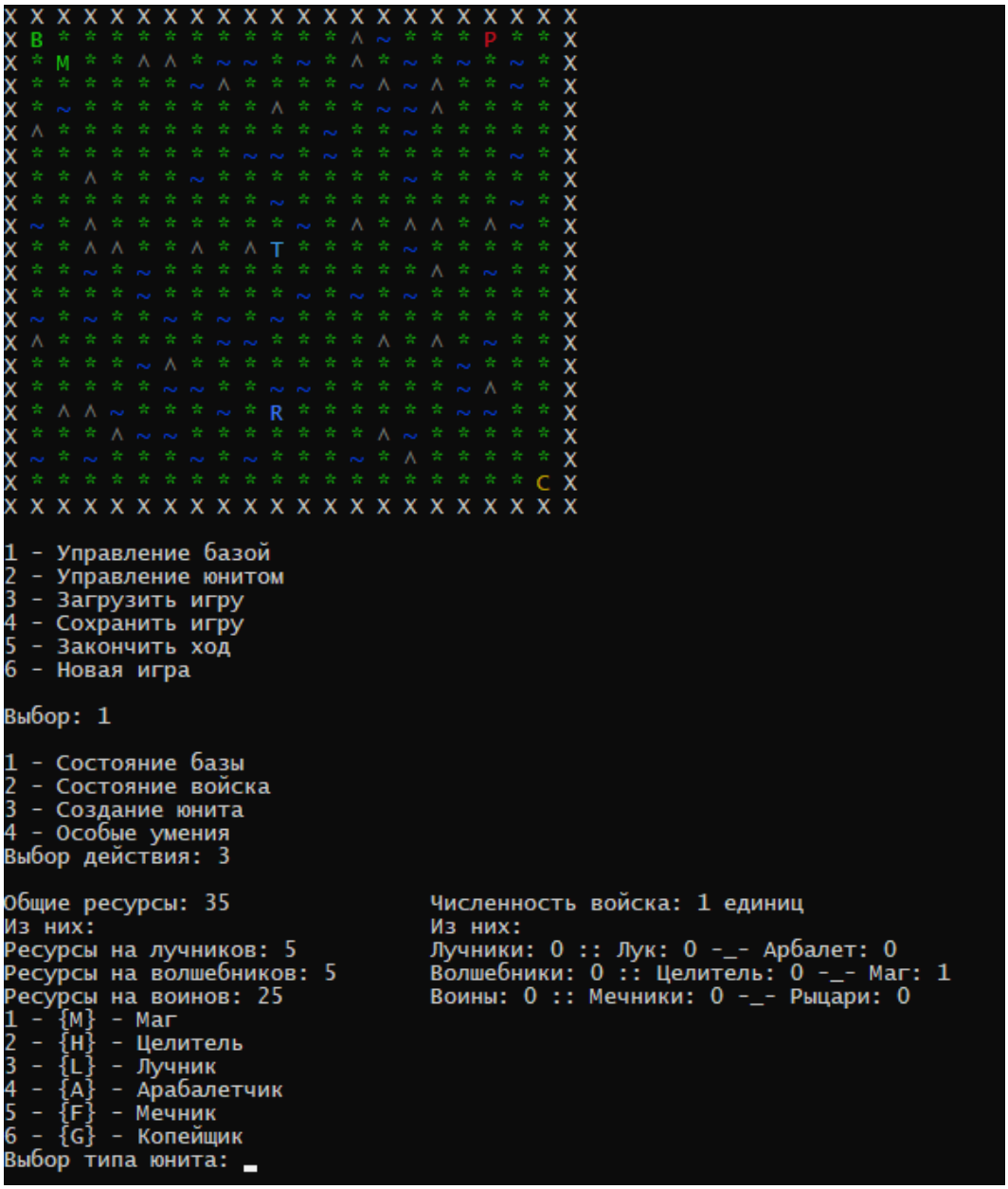

Рис. 3. Пример консольного интерфейса к заданию 3

Пример управлением юнитом представлен на рис. 4.

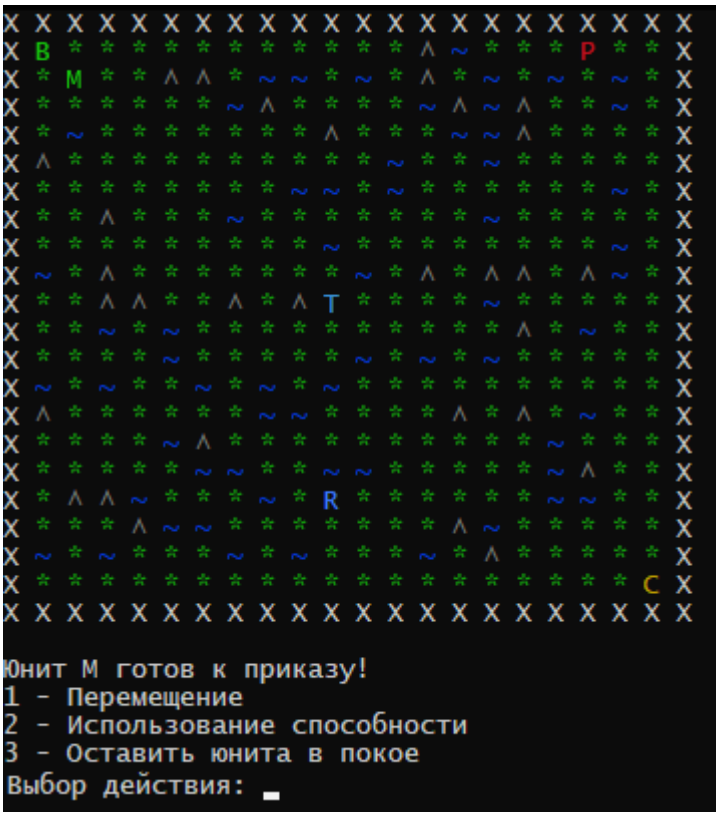

Рис. 4. Интерфейс управления юнитом из консоли

Пример состояния базы представлен на рис. 5.

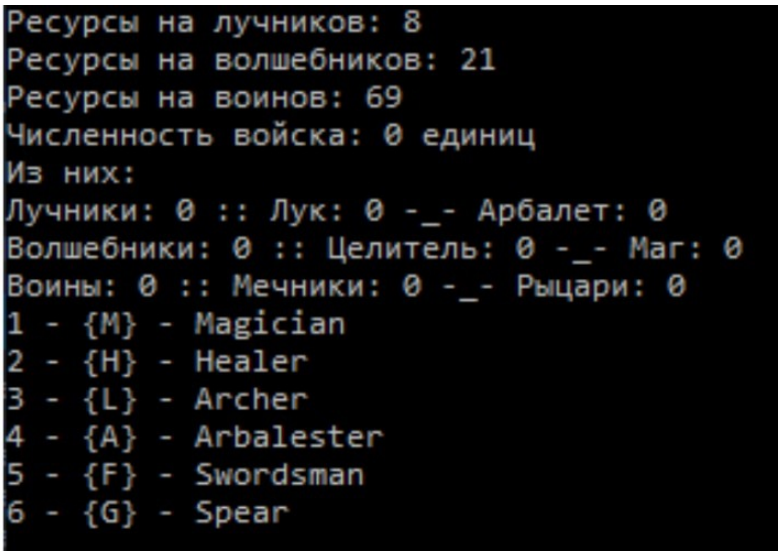

Рис. 5. Пример отображения состояния базы из консольного интерфейса

Задача 4. Реализация программы курсовой

Разработать и реализовать набор классов по теме курсового проекта. Продумать структуру и целесообразность реализации в парадигме ООП.

Технические требования

A. Реализация на Python 3.*.

B. Обратить внимание на структуру проекта (разделить проект по файлам, выделить миксины, использовать декораторы, где возможно).

Полезные ссылки

1. Kettler R. A Guide to Python's Magic Methods. URL: https://rszalski.github.io/magicmethods/
2. Hettinger R. Descriptor HowTo Guide. URL: https://docs.python.org/3.7/howto/descriptor.html
3. Understanding Python metaclasses. URL: https://blog.ionelmc.ro/2015/02/09/understanding-python-metaclasses
4. Timeouts and cancellation for humans. URL: https://vorpus.org/blog/timeouts-and-cancellation-for-humans/
5. Notes on structured concurrency, or: Go statement considered harmful. URL: https://vorpus.org/blog/notes-on-structured-concurrency-or-go-statement-considered-harmful/
6. Пользовательские атрибуты в Python. URL: https://habr.com/ru/post/137415/
7. Классы в языке Python. Документация на русском. URL: https://docs-python.ru/tutorial/klassy-jazyke-python/

8. Настройка доступа к атрибутам класса Python. URL: https://docs-python.ru/tutorial/klassy-jazyke-python/nastrojka-dostupa-atributam-klassa/

9. Modules and Packages: Live and Let Die! URL: http://dabeaz.com/modulepackage/

10. Программирование на Python Санкт-Петербург, осень 2018. URL: https://compscicenter.ru/courses/python/2018-autumn/

Требования к отчёту по работе

1. Титульный лист;
2. Описание постановки задачи;
3. Схема программы;
4. Описания классов и программных реализаций;
5. Скриншоты работы программы – запуск, ручное тестирование.