

Московский государственный университет имени М.В. Ломоносова

Факультет Вычислительной математики и кибернетики

Кафедра Алгоритмических языков

Отчет по заданию практикума по объектно-ориентированному программированию

**Система поддержки бронирования и заселения гостиницы**

**Выполнил:**

студент 425 группы

Рогожин Денис   
Андреевич

Москва, 2022

**Оглавление**

1. Уточненная постановка задачи…….………………………………..……3
2. Диаграмма классов……………………………………………………..….4
3. Спецификации интерфейса…………………………………………..…5-7
4. Диаграмма объектов…………………………………………………….…7
5. Инструментальные средства……………………………………………....8
6. Файловая структура…………………………………………………….….8
7. Пользовательский интерфейс……………………..……………….….8-10

# **Уточнение постановки задачи**

Задача – создать компьютерную среду, автоматизирующую управление занятостью номеров гостиницы. Программа обрабатывает входящий поток заявок на бронирование и заселение, хранит информацию о занятости номеров и считает статистику занятости гостиницы и полученную прибыль.

В гостинице есть номера разных типов. При нехватке номеров определенного типа можно использовать номера большей комфортности по меньшей цене.

Для тестирования необходимо смоделировать входящий поток заявок на бронирование и заселение. Вид и параметры заявки определяются случайным образом.

В ходе моделирования система должна предоставлять всю информацию о занятости номеров. По окончании моделирования выводится собранная статистика.

В параметры моделирования были включены:

1. Число дней моделирования(от 12 до 30)
2. Число номеров каждого типа(от 1 до 9)
3. Цена номеров каждого типа
4. Минимальное(min\_t) и максимальное(max\_t) время генерации заявок

Заявки на бронирование и заселение генерируются как случайная величина из равномерного распределения на [min\_t, max\_t].

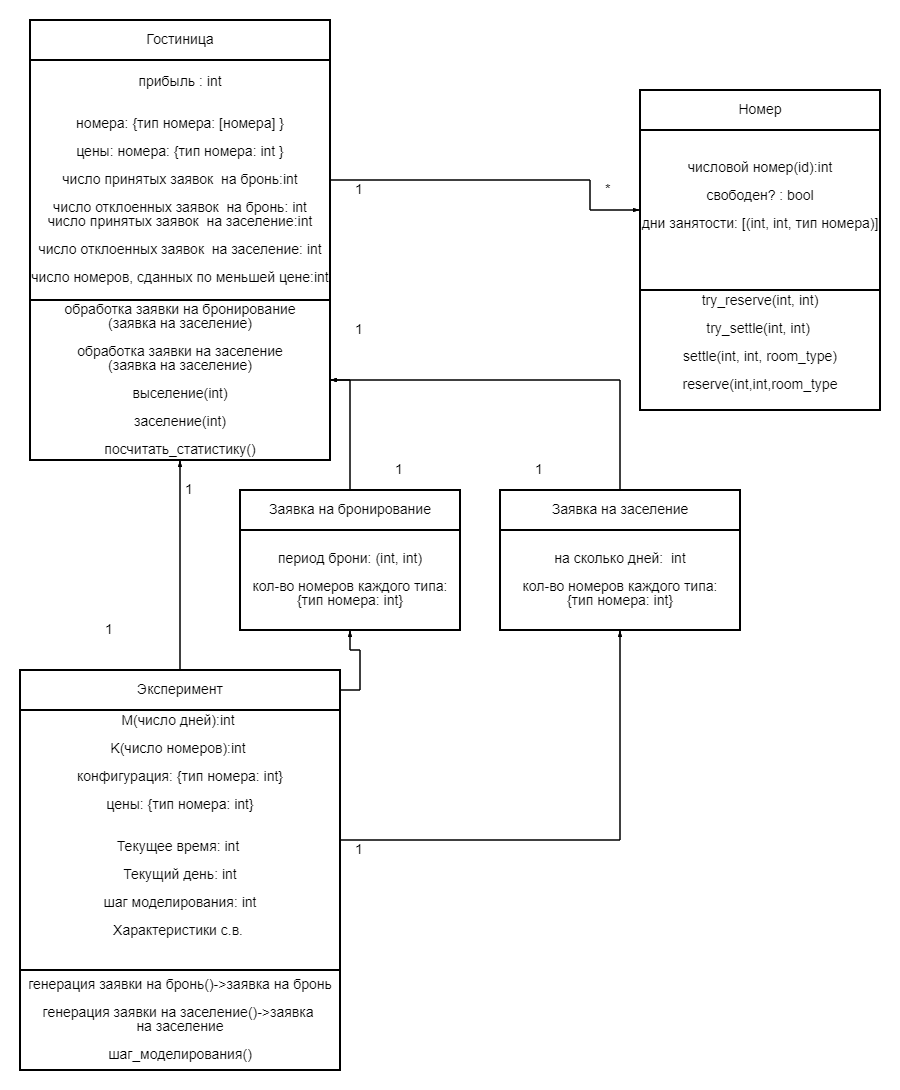
Выселение из гостиницы происходит в 10 утра каждые сутки.

Заселение в гостиницу происходит в 12 утра каждые сутки.

Случайная величина, отвечающая за количество номеров определенного типа в заявке, генерируется из дискретного распределения:

|  |  |  |  |
| --- | --- | --- | --- |
| Значение | 0 | 1 | 2 |
| Вероятность | 0.5 | 0.4 | 0.1 |

# **Диаграмма классов**



**Спецификации интерфейса**

**# интерфейс класса, представляющего комнату**

**class Room():**

#конструктор, принимает на вход номер создаваемой комнаты

def \_\_init\_\_(self, id\_):

#функция, проверяющая возможность забронировать данный номер в данный период

def try\_reserve(self, start\_day, end\_day):

#функция, бронирующая номер в данный период по цене комнаты типа room\_type

def reserve(self, start\_day, end\_day, room\_type):

#функция, проверяющая возможность заселиться в данный номер

def try\_settle(self, current\_day, day\_count):

#функция, проводящая заселение в номер

def settle(self, current\_day, day\_count, room\_type):

**# интерфейс класса, представляющего из себя заявку на бронирование**

**class Reserve\_application():**

# конструктор, принимающий конфигурацию заявки и период бронирования

def \_\_init\_\_(self, configuration, first\_day, last\_day):

**# интерфейс класса, представляющего из себя заявку на заселение**

**class Settle\_application():**

# конструктор, принимающий конфигурацию заявки и период заселения

def \_\_init\_\_(self, configuration, current\_day, day\_count):

**# интерфейс класса, представляющего из себя отель**

**class Hotel():**

# конструктор, принимающий конфигурацию отеля и цены номеров разного типа

def \_\_init\_\_(self, configuration, prices):

# функция, принимающая заявку на бронирование

def accept\_reserve\_application(self, reserve\_application):

# функция, принимающая заявку на заселение

def accept\_settle\_application(self, settle\_application):

# функция, проводящая выселение из гостиницы

def leaving(self, curr\_day):

# функция, проводящая заселение в гостиницу

def settling(self, curr\_day):

# функция, считающая статистику занятости в данный момент времени

def count\_stats(self):

**# интерфейс класса, представляющего из себя отель**

**class Experiment():**

# конструктор, принимающий число дней моделирования, кол-во комнат разного типа, цены на комнаты разного типа и коэффиценты генерации заявок

def \_\_init\_\_(self, M, room\_conf, prices\_conf, generation\_conf):

#функция для генеарации заявки на бронирование

def generate\_reserve\_application(self)

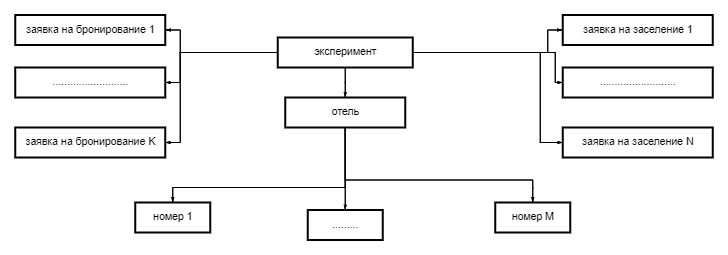
#функция для генеарации заявки на заселение

def generate\_settle\_application(self)

#функция, совершающая шаг моделирования

def step(self)

**Диаграмма объектов**



**Инструментальные средства**

Язык разработки – Питон.

Среда разработки – Jupyter notebook

Используемые библиотеки – Tkinter, numpy, enum, random

**Файловая структура**

applications.py – классы заявок на бронирование и заселение

experiment.py – класс эксперимент

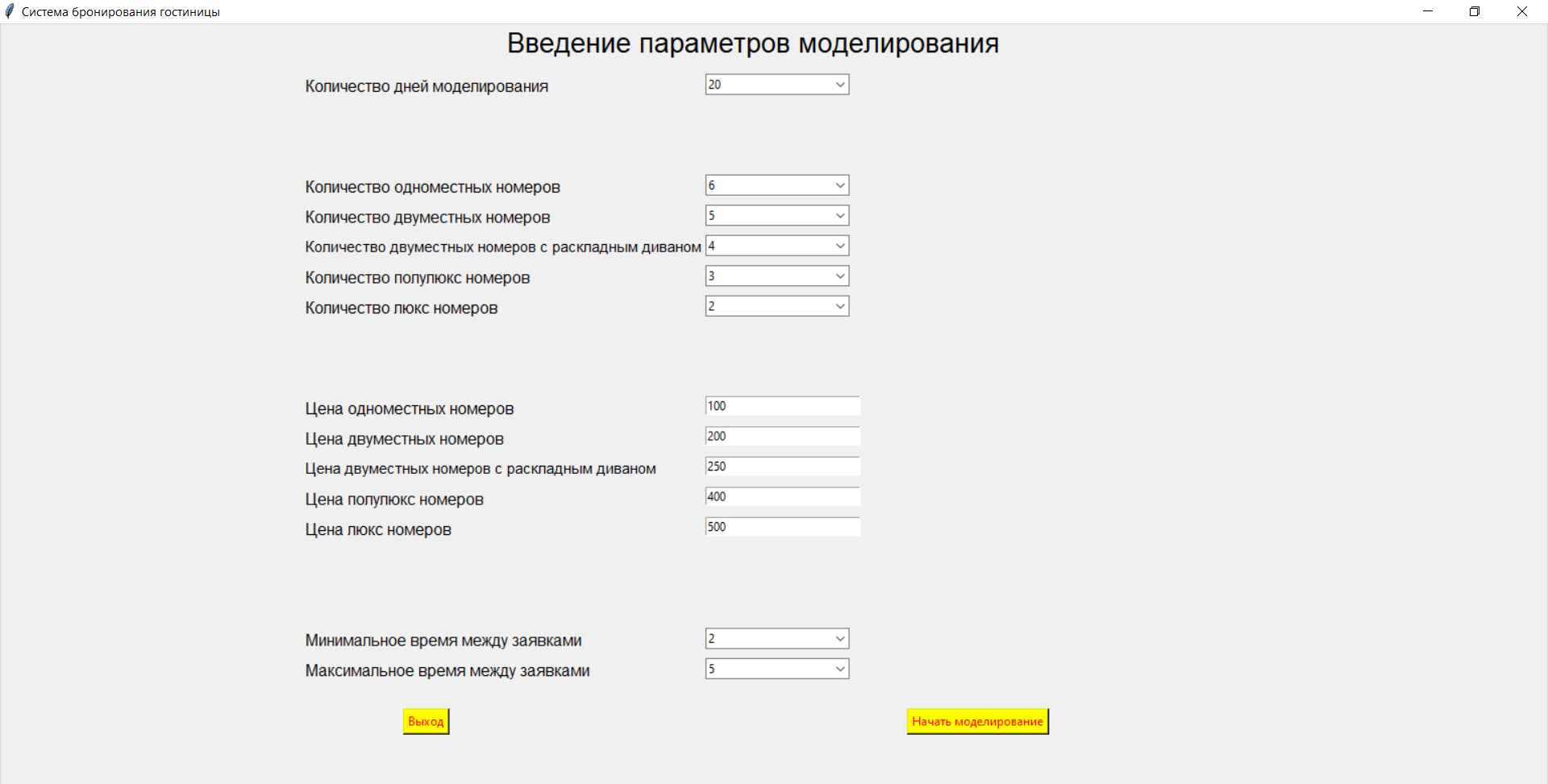
hotel.py – класс отель

interface.py – класс с интерфейсом и точка входа для запуска программы

room.py – класс комната

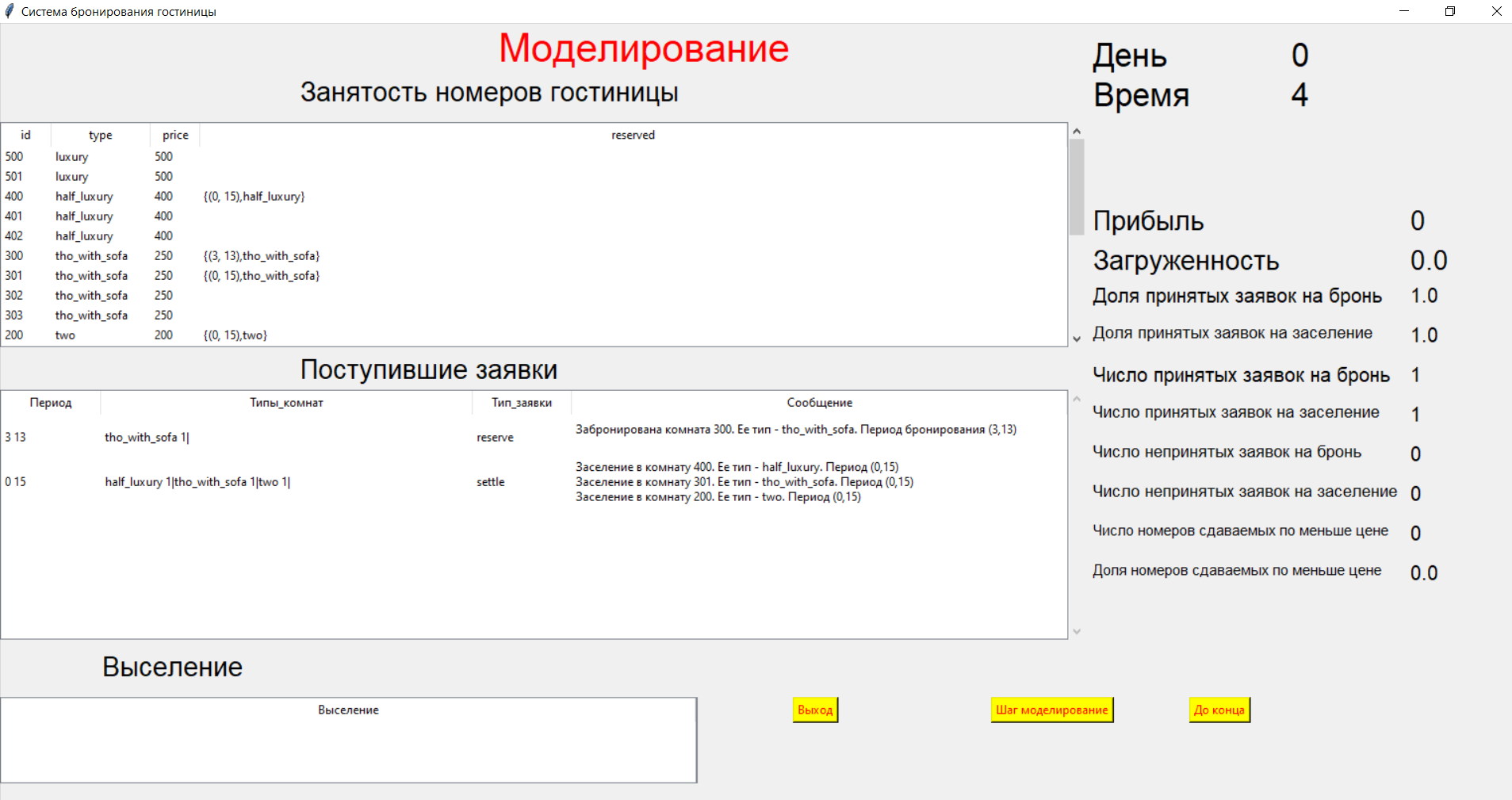
room\_type.py – типы комнат

**Пользовательский интерфейс**



При запуске программы выводится такое окно, в котором задаются параметры моделирования. При нажатии кнопки «выход» происходит окончание работы программы, при нажатии кнопки «начать моделирование» начинается процесс моделирования. Если при нажатии этой кнопки данные заданы неправильно(например, вместо чисел написаны сторонние символы или минимальное время генерации больше максимального), то пользователю выводится сообщение об ошибке и дается возможность исправить ввод.

После правильного ввода и нажатия кнопки «начать моделирование появляется такое окно:

Здесь выводится информация о текущей занятости номеров, текущий день и время, статистика занятости отеля, информация о выселении из гостиницы, а также список поступивших заявок за последние 2 часа.

При нажатии «выход» происходит завершение работы программы, при нажатии «шаг моделирования» происходит шаг моделирования. Если заданное время моделирования истекает, то выводится окно со статистикой за все время моделирования. Такое же окно выводится по нажатии кнопки «до конца».

Интерфейс окна со статистикой выглядит следующим образом:

