Московский государственный университет имени
М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра алгоритмических языков

Отчёт о выполнении задания практикума

«Ассистент в бронировании»

Студент 325 группы
М. А. Гулак

Москва, 2023

1 Постановка задачи

Требуется реализовать программу, в диалоге с которой пользователь может забронировать номер в гостинице. Для гостиницы известны вид номеров и их количество, посуточная оплата, а также их текущая занятость на ближайшую неделю. Вся эта информация о гостинице задается в текстовом файле и включает данные об уже забронированных номерах гостиницы.

Исходный запрос пользователя на бронирование может быть определен частично (например, не задан вид номера гостиницы), и в ходе диалога ему предлагаются возможные варианты бронирования и уточняются все его детали (в случае ошибок ввода делаются подсказки). В конце диалога выводится детальное описание произведенного бронирования.

Диалог может допускать возможность отмены или изменения бронирования. Также по специальному запросу может быть выведена (для администрации гостиницы) вся информация о забронированных на неделю номерах.

1.1 Базовые требования

1. Загрузка внутренного представления отеля из файла.
2. Выявление ошибок в файле.
3. Обработка пользовательского запроса, в том числе ошибок в нём.
4. Возможности отменить или изменить запрос.
5. Отображение занятости отеля на неделю по специальному запросу.

1.2 Идивидуальные части

1. Обновление текстового файла после окончания работы в программе.
2. Проверка существования файла с данными об отеле.

2 Модули проекта

Проект состоит из следующих модулей:

- `DataTypes.hs` — объявление основных типов;
- `Constants.hs` — константы.

- `TypesHandle.hs` — реализация функций для работы с новыми типами данных;
- `Book.hs` — реализация функций для диалога с пользователем и логики самого диалога;
- `Main.hs` — модуль с главной функцией, с которой начинается работа.

В модуле `DataTypes.hs` описаны следующие типы:

- `Day` — описывает день недели, соответственно, содержит 7 значений;
- `RoomType` — описывает тип комнаты: одиночная или двойная;
- `Room` — структура для описания комнаты, имеющая поля:
  - `roomId` — номер комнаты;
  - `rommType` — тип комнаты;
  - `roomPrice` — стоимость комнаты;
  - `roomBookWeek` — занятость комнаты на неделю.
- `Hotel` — структура для описания отеля, имеющая поля:
  - `hotelName` — название отеля;
  - `hotelTypes` — типы комнат отеля;
  - `hotelRoomAmount` — количество комнат;
  - `hotelBusy` — занятость отеля на неделю.
- `Request` — структура для описания запроса пользователя, имеющая поля:
  - `roomtype` — тип комнаты;
  - `days` — список дней для бронирования.

В модуле `Constants.hs` описаны следующие константы:

- `week` — список дней недели;
- `weekS1`, `weekS2` и `weekS` — списки дней недели в виде строк в полном и сокращённом виде соответственно, а также объединение этих списков;
- `typeS` — список типов комнаты в виде строк;

- `assocWeekSD` — ассоциативный список для перевода дня недели из строки в соответствующий тип данных;
- `requestPrompts` — список подсказок при получении запроса;
- `help` — список для проверки правильности запроса.

В модуле `TypesHandle.hs` описаны следующие функции:

- `transformFile` — перевод файла в виде строки в список комнат и структуру `Hotel`;
- `parseFile` — разбиение списка строк файла на имя отеля и список комнат;
- `initializeHotel` — инициализация типа данных Hotel по его имени и списку комнат;
- `initializeRooms` — инициализация типа данных Room по списку строк;
- `weekBusy` и `weekBusyHelp` — создание списка занятости отеля на неделю;
- `addToBusy` — обновление списка занятости;
- `stringToRoom`, `roomToString` — перевод строки, соответствующей номеру отеля из файла, в тип данных `Room` и обратно;
- `transformDaysToBool`,`transformDaysFromBool` — перевод списка дней недели из списка строк или дней в формате `Day` в список True/False и обратно;
- `updateRoom` — обновление одной комнаты в соответствии с запросом пользователя;
- `updateRooms` — обновление списка комнат;
- `updateHotel` — обновление отеля по одной комнате;
- `findRooms` — поиск списка комнат, подходящих под запрос пользователя;
- `isRoomBusy` — проверка занятности комнаты;
- `showRoom` — преобразование комнаты в строку для вывода на экран;

- `showRooms` — преобразование списка комнат в строку для вывода на экран;
- `showBusy`, `showBusyHelp` и `showBusyDay` — преобразование списка занятости отеля в строку для вывода на экран;
- `transform` — преобразование запроса из списка строк в список запросов;
- `transformDaysIntervalToBool` и `transformOneDay` — преобразование интервала дней недели в список True/False.

В модуле `Book.hs` описаны следующие функции:

- `fullRequestHandle` — обработка запроса пользователя;
- `getInitialRequest` — получение начального запроса от пользователя (может быть как полным, так и неполным);
- `processReq` — уточнение запроса и поиск комнат по запросу;
- `getRequest` — получение уточнённого запроса от пользователя;
- `book` — бронирование комнаты;
- `run` — основная функция обработки всех команд.

3 Используемые библиотеки

При реализации использовались следующие библиотеки:

- `Data.Maybe`
- `Text.Read`

4 Сценарии работы с приложением

Для начала работы необходимо запустить в интерпретаторе ghci файл `Main.hs`, а в нём — функцию `main`. После запуска пользователю будет предложено ввести название файла, в котором хранится информация об отеле. После этого

В случае ввода названия несуществующего файла будет выведено следующее сообщение:

```
ghci> main
> To start booking process type file with hotel info.
hotel3.txt
Error while reading the file.
```

Если же файл существует, но в нём есть ошибка, не позволяющая инициализировать внутреннее представление, то будет выведено сообщение:

```
ghci> main
> To start booking process type file with hotel info.
hotel2.txt
Error in the file.
```

,

либо

```
ghci> main
> To start booking process type file with hotel info.
hotel1.txt
Error: Empty list of rooms
```

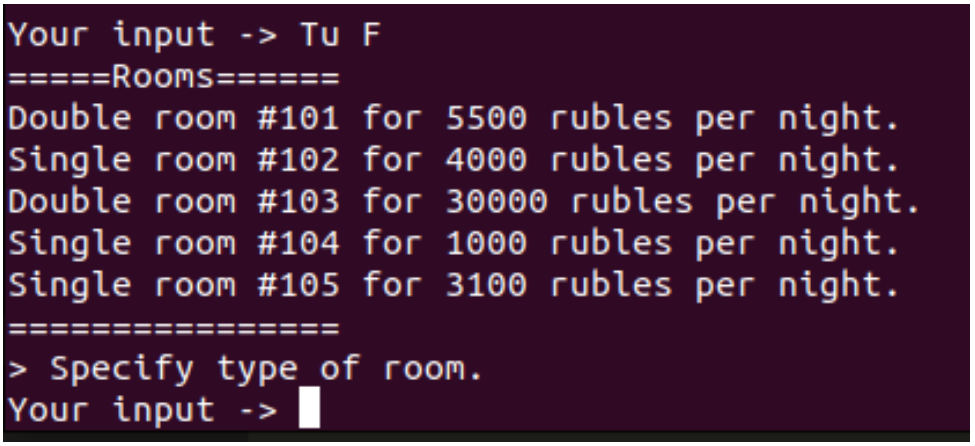
Если с файлом всё в порядке, то будет выведена информация об отеле (его название), а также о доступных командах.

```
> To start booking process type file with hotel info.
hotel.txt
********************************
> Hello! Glad to see you in our hotel Solnishko!
> Commands:
/start - to start booking a room,
/change - to change booking parameteres,
/cancel - to stop booking process,
/exit - to exit from program.
********************************
Your input ->
```

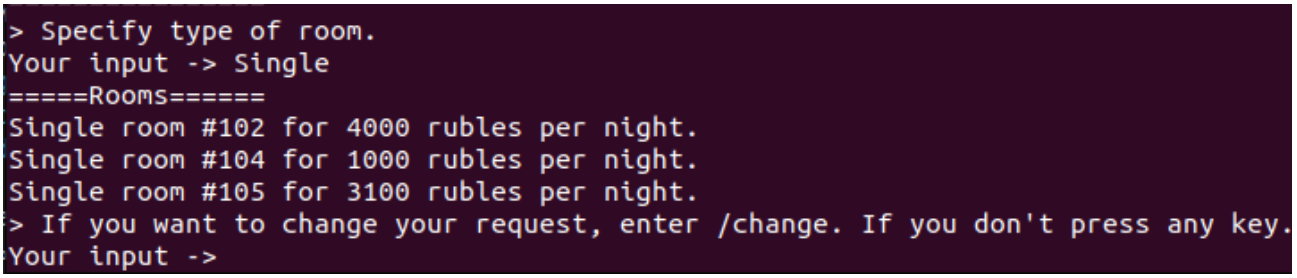
Далее для начала процедуры бронирования нужно ввести `/start`. Пользователю будет предложено ввести запрос в виде "День заезда, день выезда, тип комнаты". Последний параметр необязателен на этом этапе, его можно не вводить.

```
Your input -> /start

> Type your request below in form
1.Check-in date.
2.Check-out date.
3.(optional) Type of room: Single or Double.
For example: "W Su Single" or "M Th".

Your input ->
```

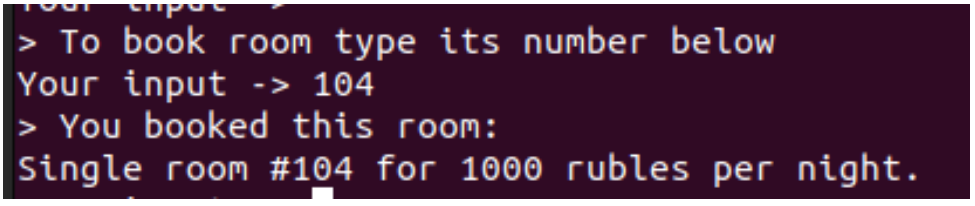
На этом этапе пользователь может отменить бронирование, либо ввести свой запрос. После ввода запроса будут выведены все доступные варианты.



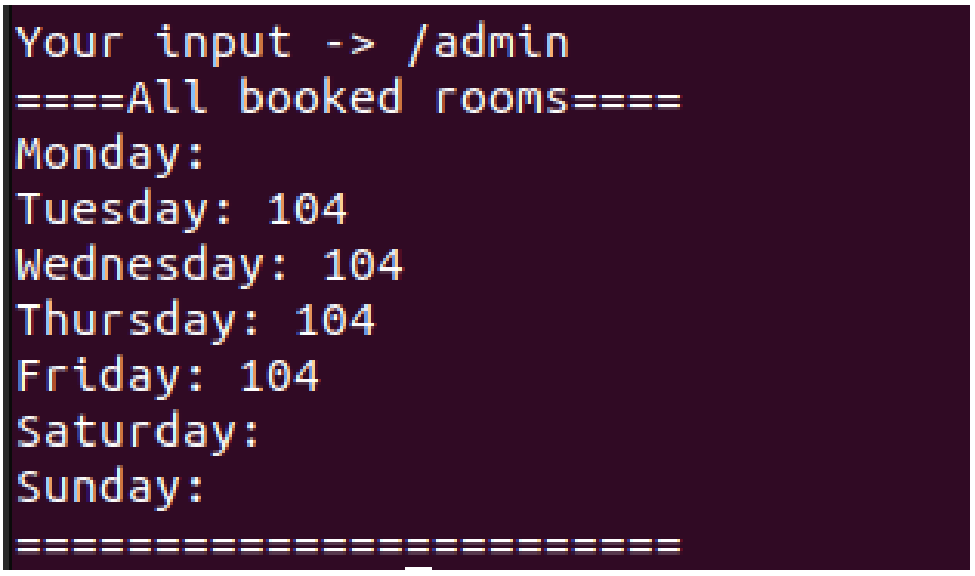
Если запрос был неполным, то после просмотра результатов нужно будет его дополнить необходимым типом комнаты.



После этого можно изменить запрос, введя команду `/change`, тогда просто процесс брони начнётся заново. Если нажать любую клавишу, то пользователю будет предложено выбрать номер комнаты, которую он хочет забронировать. После выбора комнаты будет показана ещё раз информация об этой комнате. На этом процесс бронирования заканчивается.



С помощью специальной команды `/admin` можно посмотреть всю занятость отеля на неделе.



Как видно на скриншоте, здесь отобразился забронированный ранее номер 104.

Данные, которые вводит пользователь, проверяются на корректность и программа ожидает от пользователя подходящего ввода.