Версия 1.0

12.05.2020

Проект: Хотел

Представено от: Димитър колев

Студент Специалност: КН

Курс: 1 Поток: 2 ФН: 6MI0800091

Съдържание

[Кратко описание на класовете 2](#_Toc103343081)

[Непредвидени проблеми 2](#_Toc103343082)

[Кратко описание на класовете 2](#_Toc103343083)

[Структура на класовете за модели (стая, резервация, затворена стая) 2](#_Toc103343084)

[Дата 2](#_Toc103343085)

[Време 3](#_Toc103343086)

[Период 3](#_Toc103343087)

[Стая 3](#_Toc103343088)

[Резервация 3](#_Toc103343089)

[Затворена стая 3](#_Toc103343090)

[Структура на класовете за бизнес логика 3](#_Toc103343091)

[Бизнес логика за Стая 3](#_Toc103343092)

[Бизнес логика за резервация 3](#_Toc103343093)

[Бизнес логика за затвяряне на стая 3](#_Toc103343094)

[Структура на класовете за контролери 3](#_Toc103343095)

[Контолер на Стая 3](#_Toc103343096)

[Контолер на резервация 4](#_Toc103343097)

[Контолер на затворена Стая 4](#_Toc103343098)

[Идеи за бъдещо развитие 4](#_Toc103343099)

[Линк към Github 5](#_Toc103343100)

# Кратко описание на класовете

След като се запознах с изискванията на проекта, реших да го реализирам чрез MVC (**Model-View-Controller**) модела. Затова направих базови класве за стая (room), резервация (**reservation**) и за затворена стая (**closeRoom**), както и класове за дата (**Date**), време (**Time**), и период (**Period**).

След това започнах да изграждам бизнес логиката на приложението и как да обработва множество от информация за стая (**RoomsService**), резервация (**ReservationsService**) и затворяне на стая (**CloseRoomsService**).

Накрая изградих контролери за стаи (**RoomsController**), резервация (**ReservationsController**) и затворяне на стая (**CloseRoomsController**). Те управляват кога кой метод да се извика от бизнес логиката. Също така контролерите имат достъп до файлове.

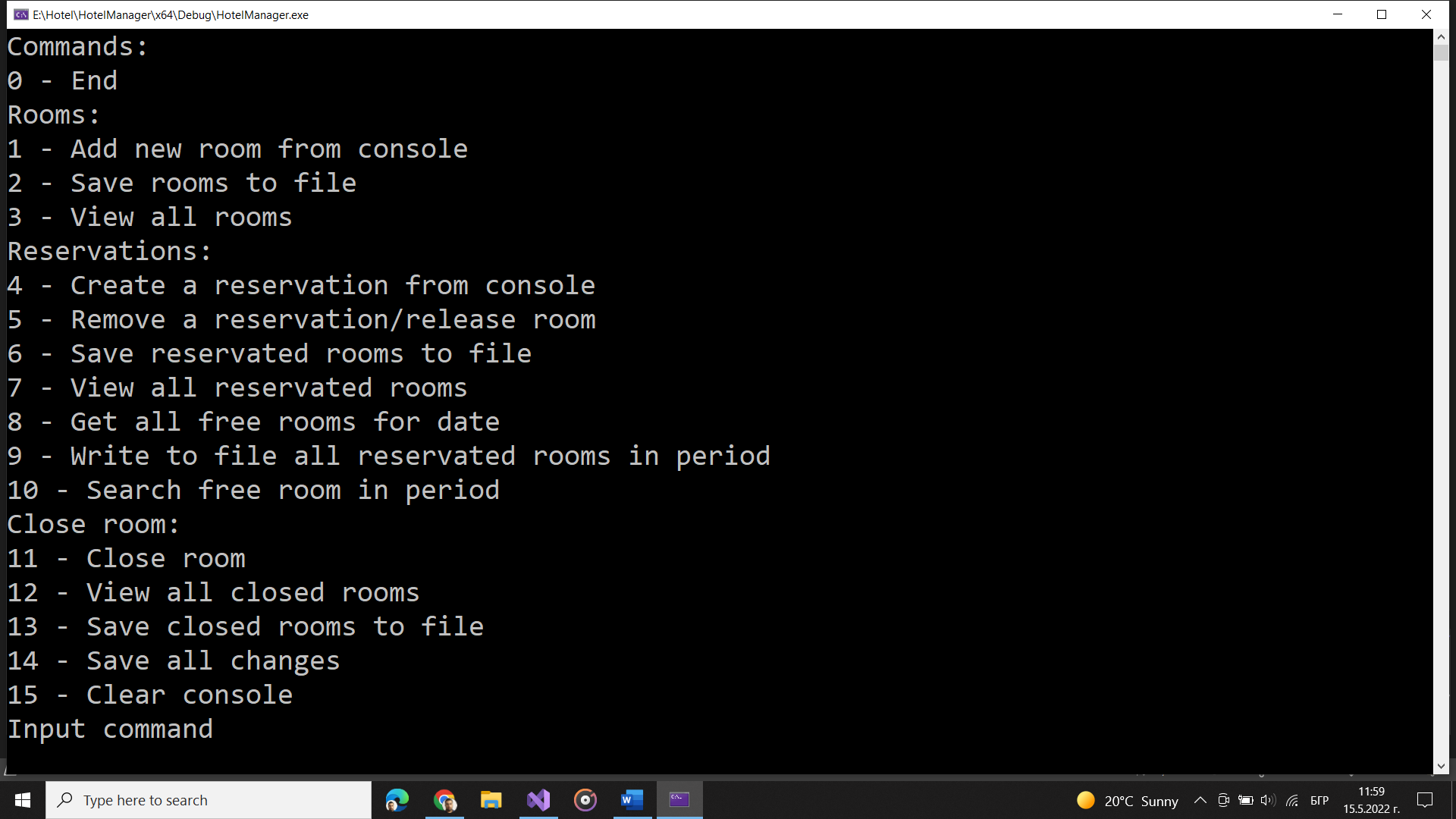
## Непредвидени проблеми

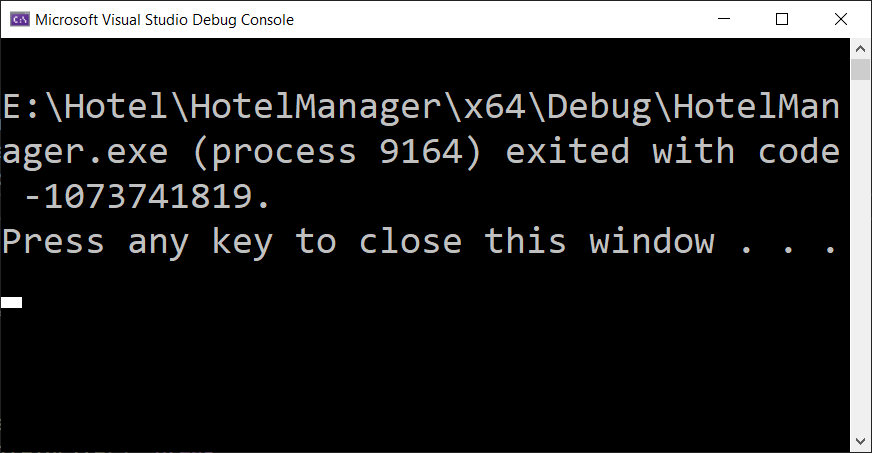
Проектът има лено задание, но реализацията беше много трудна и много дълга (40+ часа). Сблъсках се с много и различни проблеми, като например:

Как да направя справка за всички свободни стаи в даден период? (4+ часа)

Как като не вкарам файловете с дефинициите на класовете (**\*.h**) както трябва и всичко се чупи? (3+ часа)

Как не се пише код?

Стартиране на проекта – в последната си версия проекта има грашка при стартиране (код -1073740940) и понякога не може да зареди както трябва. Но след няколко пускания започва да работи, без да има видима причина затова (може нещо да бърка файловето, но не мога да го хвана от къде се получава така):



# Кратко описание на класовете

## Структура на класовете за модели (стая, резервация, затворена стая)

Картина, която съдържа текст, черен, монитор, екранна снимка

Описанието е генерирано автоматичноЗа спестяване на време и по-лесно ползване на написаното във всеки клас съм написал оператор>> и оператор<<. Това позволява бързо и удобни вземане и изпращане на данни от и към поток, независимо дали това е файл или конзола, защото съм написал както ofsream и ifstream потоците, така и ostream и istream потоците. При реализацията на проекта се наложи да има и написани оператори за сравненив в някои класове, за да може да бъде по-лесна и по-класива тяхната употреба.

### Дата

Класа за дата съдържа ден, месец и година. Ако се въведат данни, които не отговарят на реалността (например: за месец 15), то те биват променени и нормализирани (месец 15 ще стане месец 3 и годината ще се увличи с едно). Това е бизнес-логиката (алгоритъмът), която прилагам.

### Време

Класа за времето съдържа час и минути - бах решил, когато се прави резервация да се оказва от колко часа ще бъде резервирана. Посъветвах се с преподавател (др. доц. Ангелова), като казах, че това не е много нужно, защото в повечето хотели стая се наема от 14 часа и се освобождава до 10 часа. Тя потвърди моята теория, че не е нужно. Реших да не премахвам този клас, защото може някой ден да се иска фунцкионалност за време на престоя.

### Период

Класа за периода съдържа ден от (dayFrom) и ден до (dayTo). Направих този клас, за да мога по-лесно да управлявам логиката свързана с тези данни – например дали ден до е след ден от или не е въведана информацията коректно.

### Стая

Класа за стаята съдържа идентификационен номер и брой легла с нужните допълнителни функции за вземане и подаване на информация.

### Резервация

Класа за резервация съдържа идентификационен номер, дата на настаняване, дата на напускане, идентификационен номер на стаята и име на гост. Тук съм добавил нужните до фунцкии за вземане и подаване на информация. Като съм добавил и други методи за:

* Резервацията е в даден период;
* Запис на резервацията в **текстови** файл.

### Затворена стая

Класа за затворена стая съдържа идентификационен номер и перид на затвяне (дата от и дата до) и допълнителна информация.

## Структура на класовете за бизнес логика

Тези класове управляват колекции (масиви) от обекти. Тук също за във всеки клас съм написал оператор>> и оператор<<. За да мога да боравя удобно с обек от самата колекция съм написал оператор[].

### Бизнес логика за Стая

Този класа съдържа саморазширяващ се масив от тип стая, както и нужни функции за заделяне и освобождаване на памет.

### Бизнес логика за резервация

Този класа съдържа саморазширяващ се масив от тип резервация, както и нужни функции за заделяне и освобождаване на памет. Тук съм добавил методи, които трябва да:

* кажат дали има резервирана стая в даден ден или в даден период;
* върнат константен динамичен масив на резервираните стаи в даден период;
* провери дали има направена резервация по даден идентификационнен номер на стая;
* премахване на резервация/освобождаване на стая;
* съхрани масива от резервации в **текстови** файл

### Бизнес логика за затвяряне на стая

Този класа съдържа саморазширяващ се масив от тип затворени стаи, както и нужни функции за заделяне и освобождаване на памет. Тук съм добавил методи, които трябва да:

* кажат дали дадена стая е затворена;
* добави затворена стая към колекцията със затворени стаи.

## Структура на класовете за контролери

Всеки от контролерите има достъп до нужната му бизнес логика. При създаване на обект от тип контролер (чрез конструктора) се прочита информацията от двоичния файл.

### Контолер на Стая

Контролера за стая има цел да обработва командите на потребителя чрез бизнес логиката за:

* добавяне на нова стая от конзолата (хотела може да е няколко бунгала и да решат да разшият бизнеса си)
* извеждане на стаите на конзолата
* записване на промените във файл

### Контолер на резервация

Контролера за резервация има цел да обработва командите на потребителя чрез бизнес логиката за:

* добавяне на нова резервация от конзолата
* търсене на свободна стая
* проверка дали дадена стая е свободна за дадена дата или период
* премахване на резервация/освобождаване на стая
* проверка дали дадена стая е затворена
* други допълнителни методи

### Контолер на затворена Стая

Контролера за затворена стая има методи, които има цел да обработва командите на потребителя чрез бизнес логиката за:

* четене на затворени стаи от файл
* затваряне на стая от конзолата
* за показване на затворени стаи
* за записване на
* търсене на свободна стая
* проверка дали дадена стая е свободна за дадена дата или период
* премахване на резервация/освобождаване на стая
* проверка дали дадена стая е затворена
* други допълнителни методи

# Идеи за бъдещо развитие

Докато разработвах проекта ми хрумнаха много допълнителни идеи:

1. Запазване на телефонен номер на госта;
2. Запазване на информацията в база данни вместо във файлове;
3. Интернет приложение – така всеки гост сам може да си резервира стая за даден период. В този вариант може да се помисли за обогатяване на избраната/използваната технология или да се потърси по-добра технология/и с цел по-лесна бъдеща поддръжка. Да се намери вариант за оптимално използване на ресурси
   * Обсъждане на маркетинг частта бизнес-за-бизнес (**B2B**) или бизнес-за-клиент (**B2C**)
   * Време за разработка (човекочасове);
   * Финанси;
   * Сървъри – памет на сървъра, както и процесор на сървъра, защита;
   * Време на обработване на данните от програмата и лесно ли е за ползване от крайните потребители;
   * Други;
4. В интернет приложението да функционалности за потребителите:
   * Резервиране на стая;
   * Преглед кога потребителят е наемал стая (история на резервациите);
   * Промоции за активни гости/потребители на хотела;
   * Интеграция със социални мрежи и влияние в тях;
   * Ако бизнес стратегията е бизнес-за-бизнес (**B2B**) може да се реализира функционалност за много хотели (нещо като **booking.bg**)
   * Табло с информация (**Dashboard**)
   * Интеграция към Excel – експортиране на данни
   * Изпращане на електронни писма (**email**) с информация за заетостта на хотела всеки месец, за потвърждение за наемане на стая и други неща;
   * Други маркетингови планове и идеи за привличане на клиенти за хотели.
5. Предложение към ООП екипа:

Завършил съм успешно курс „ИТ кариера“ и сравнявам изпитите там и изпитите в университета. Там на всеки изпит имаше ясно и пълно условие кое как да бъде реализирано и коя функицоналност колко точки носи, както и скала за превръщане на точките в оценка. Това е доста добре, защото даден студент знае се изисква. Разбирам и Вашият подход – самият студент да реши как да направи дадена фунцкионалност с цел да се сблъска сам с проблема. Но всеки проблем има безбраой варианти за решение (спорихме с един колега кое е по-добре за времето да се слага ден, месец и година по подразбиране или да се нормализират данните до валидни такива). Всеки клиент има различно мислене и на един ще му е удобно по един начин, а друг ще харесва различен подход за работа със ситемата. Затова има различни безнес стратегии и бизнес математики за проектиране на продукити.

# Линк към Github

Може да намерите проекта тук: <https://github.com/dimitarkole/Hotel>