Hotel

Generated by Doxygen 1.9.3

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 Date Class Reference

Class representing date with day, month and year.

`#include <Date.hpp>`

Public Member Functions

- Date (unsigned short d=1, unsigned short m=1, unsigned short y=1900)

  Construct a new Date object from day, month and year. Default Date is 1/1/1900.
- bool operator< (const Date other) const

  checks if this Date is chronologically before other Date
- bool **operator<=** (const Date other) const

  see operator<
- bool operator> (const Date other) const

  checks if this Date is chronologically after other Date
- bool **operator>=** (const Date other) const

  see operator>
- bool operator== (const Date other) const

  checks if two dates are identical
- const char * operator() (char *buf) const

  records this Date in buffer in format YYYY-MM-DD
- int operator- (Date other) const
- Date & operator++ ()

  overloaded prefix incremention operator for Date

Static Public Member Functions

- static Date getToday ()

  Get the today date.

Friends

- std::istream & operator>> (std::istream &is, Date &d)

  overloaded operator for inputing Date
- std::ostream & operator<< (std::ostream &os, const Date &d)

  overloaded operator for outputing Date

3.1.1 Detailed Description

Class representing date with day, month and year.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Date()

```
Date::Date (
            unsigned short d = 1,
            unsigned short m = 1,
            unsigned short y = 1900 )  [inline]
```

Construct a new Date object from day, month and year. Default Date is 1/1/1900.

Parameters

d	day
m	month
y	year

3.1.3 Member Function Documentation

3.1.3.1 getToday()

```
Date Date::getToday ( )  [static]
```

Get the today date.

Returns

Date

3.1.3.2 operator()()

```
const char * Date::operator() (
            char * buf ) const
```

records this Date in buffer in format YYYY-MM-DD

Parameters

buf	buffer where Date is recorded

Returns

const char* pointer to beginning of buf

3.1.3.3 operator++()

```
Date & Date::operator++ ( )
```

overloaded prefix incremention operator for Date

Returns

Date& reference to this Date

3.1.3.4 operator-()

```
int Date::operator- (
            Date other ) const
```

Parameters

other	Date

Returns

int difference between of this Date and other

3.1.3.5 operator<()

```
bool Date::operator< (
            const Date other ) const
```

checks if this Date is chronologically before other Date

Parameters

other	compared Date

Returns

true this Date is chronologically before other

false this Date is not chronologically before other

3.1.3.6 operator==()

```
bool Date::operator== (
            const Date other ) const
```

checks if two dates are identical

Parameters

other	compared Date

Returns

true the dates are identical

false the dates are not identical

3.1.3.7 operator>()

```
bool Date::operator> (
            const Date other ) const
```

checks if this Date is chronologically after other Date

Parameters

other	compared Date

Returns

true this Date is chronologically after other

false this Date is not chronologically after other

3.1.4 Friends And Related Function Documentation

3.1.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & os,
            const Date & d )  [friend]
```

overloaded operator for outputing Date

Parameters

os	output stream
d	Date to be output

Returns

std::ostream& reference to the output stream

3.1.4.2 operator>>

```
std::istream & operator>> (
            std::istream & is,
            Date & d )  [friend]
```

overloaded operator for inputing Date

Parameters

is	input stream
d	Date to be input

Returns

std::istream& reference to the input stream

The documentation for this class was generated from the following files:

- Date.hpp
- Date.cpp

3.2 DatePeriod Struct Reference

Class containing two dates forming a period of time from Date to Date.

```
#include <Date.hpp>
```

Public Member Functions

- unsigned length () const
  distance in days of the period
- DatePeriod & operator++ ()
  moving period one day froward
- void **readProper** ()
  method to read from stdin a proper period of time (from is before to)

Public Attributes

- Date **from**
  beginning Date of the period
- Date **to**
  end Date of the period

3.2.1 Detailed Description

Class containing two dates forming a period of time from Date to Date.

3.2.2 Member Function Documentation

3.2.2.1 length()

```
unsigned DatePeriod::length ( ) const  [inline]
```

distance in days of the period

Returns

unsigned days between beginning and end

3.2.2.2 operator++()

```
DatePeriod & DatePeriod::operator++ ( )
```

moving period one day froward

Returns

DatePeriod& this DatePeriod

The documentation for this struct was generated from the following files:

- Date.hpp
- Date.cpp

3.3 Hotel Class Reference

Class representing hotel with name, current Date and a building (list of rooms)

`#include <Hotel.hpp>`

Public Member Functions

- Hotel (std::string hotelDataFile)

  Construct a new Hotel object.
- **Hotel** (const Hotel &)=delete
- Hotel & **operator=** (const Hotel &)=delete
- ~**Hotel** ()

  Destroy the Hotel object.
- std::string getName () const

  get the name of this Hotel
- void **nextDay** ()

  advance to the nextDay
- bool reserveRoom (unsigned number, const DatePeriod &period, std::string name="-", std::string note="None\n")

  makes a new Reservation for particular Room and period with options for name of guest and notes to the Reservation
- Hotel & showAvailableRooms (std::ostream &, Date)

  output to stream all available rooms for a particular Date
- bool freeRoom (unsigned number)

  tries to free Room with particular ID
- Hotel & getReport (DatePeriod &period)

  Creates report for the usage of this Hotel's rooms in the period from-to. Report written in file named "report-YYYY-MM-DD.txt" where YYYY-MM-DD is the beginning of the period.
- void searchRoom (unsigned minBeds, const DatePeriod &period) const

  given minimum number of beds and a desired period to print most suitable rooms for accomodation
- bool serviceRoom (unsigned number, const DatePeriod &period, std::string note)

  plans maintenance for particular Room and period leaving note for the service
- Hotel & showToday ()

  print status of the building rooms
- Hotel & seeRoomForNights (unsigned number, unsigned nights)

  print soonest period of particular number of days when particular room is free

Static Public Member Functions

- static Date today ()

  get today's Date according to all Hotels

3.3.1 Detailed Description

Class representing hotel with name, current Date and a building (list of rooms)

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Hotel()

```
Hotel::Hotel (
            std::string hotelDataFile )
```

Construct a new Hotel object.

Parameters

hotelDataFile	path to file where rooms are recorded

3.3.3 Member Function Documentation

3.3.3.1 freeRoom()

```
bool Hotel::freeRoom (
            unsigned number )
```

tries to free Room with particular ID

Parameters

number	Room's ID

Returns

true room is now free

false room not found

3.3.3.2 getName()

```
std::string Hotel::getName ( ) const  [inline]
```

get the name of this Hotel

Returns

std::string

3.3.3.3 getReport()

```
Hotel & Hotel::getReport (
            DatePeriod & period )
```

Creates report for the usage of this Hotel's rooms in the period from-to. Report written in file named "report-YYYY-MM-DD.txt" where YYYY-MM-DD is the beginning of the period.

Parameters

period	desired period of time

Returns

Hotel& this Hotel

3.3.3.4 reserveRoom()

```
bool Hotel::reserveRoom (
            unsigned number,
            const DatePeriod & period,
            std::string name = "-",
            std::string note = "None\n" )
```

makes a new Reservation for particular Room and period with options for name of guest and notes to the Reservation

Parameters

number	of the desired Room
from	accomodation Date
to	leaving Date
name	guest's name
note	note to the reservation

Returns

true successfull reservation

false failed reservation (not made)

3.3.3.5 searchRoom()

```
void Hotel::searchRoom (
            unsigned minBeds,
            const DatePeriod & period ) const
```

given minimum number of beds and a desired period to print most suitable rooms for accomodation

Parameters

minBeds	minimum number of beds
period	desired time period

3.3.3.6 seeRoomForNights()

```
Hotel & Hotel::seeRoomForNights (
            unsigned number,
            unsigned nights )
```

print soonest period of particular number of days when particular room is free

Parameters

number	ID of a room
nights	number of nights to stay in the Hotel for

Returns

Hotel& this Hotel

3.3.3.7 serviceRoom()

```
bool Hotel::serviceRoom (
            unsigned number,
            const DatePeriod & period,
            std::string note )
```

plans maintenance for particular Room and period leaving note for the service

Parameters

number	Room's ID
period	desired period of time
note	any notes to the service

Returns

true service planned successfully

false service planning failed (room not found or is reserved for the period)

3.3.3.8 showAvailableRooms()

```
Hotel & Hotel::showAvailableRooms (
            std::ostream & os,
            Date d )
```

output to stream all available rooms for a particular Date

Returns

Hotel& this Hotel

3.3.3.9 showToday()

```
Hotel & Hotel::showToday ( )
```

print status of the building rooms

Returns

Hotel& this Hotel

3.3.3.10 today()

```
static Date Hotel::today ( )  [inline], [static]
```

get today's Date according to all Hotels

Returns

Date

The documentation for this class was generated from the following files:

- Hotel.hpp
- Hotel.cpp

3.4 HotelBuilding Class Reference

Class representing list of rooms.

```
#include <HotelBuilding.hpp>
```

Public Member Functions

- HotelBuilding (std::ifstream &ifs)

  Construct a new HotelBuilding object from a text file containing rooms info Format of the file:
- **HotelBuilding** (const HotelBuilding &other)=delete
- HotelBuilding & **operator=** (HotelBuilding &other)=delete
- ~**HotelBuilding** ()

  Destroy the HotelBuilding object.
- size_t getRoomCount () const

  Get the room count.
- Room * operator[ ] (unsigned roomNumber) const

  seek for a room with particular number
- void newDate (Date d)

  update this HotelBuilding rooms data on a new Date
- void showAvailableRooms (std::ostream &os, Date d) const

  show available rooms on a particular Date
- void createReport (DatePeriod &period) const

  Create a report for the usage of rooms for a particular period of time (ending before the today Date) in folder reports Format of the report:
- void suggestRoom (unsigned beds, const DatePeriod &period)

  show top DISPLAY (or all rooms if less than DISPLAY + 1) sorted by suitabilty for the guest, given minimal number of beds and particular period of time for an eventual reservation
- void showRoomsStatesToday (Date today) const

  prints to stdout all rooms together with up to today Date info about their Availability (now, in future or in the past)
- void showRoomForNights (unsigned number, unsigned nights, Date today) const

  show soonest period of particular nights when particular room is free

Friends

- class RoomAnalyzer

  using RoomAnalyzer to perform algorithms for the room list (database)

3.4.1 Detailed Description

Class representing list of rooms.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 HotelBuilding()

```
HotelBuilding::HotelBuilding (
            std::ifstream & ifs )
```

Construct a new HotelBuilding object from a text file containing rooms info Format of the file:

1. <size> ... n(>1). <room #(n-2) number> <room #(n-2) count of beds>

Parameters

ifs	input file stream to text file, containing rooms data

3.4.3 Member Function Documentation

3.4.3.1 createReport()

```
void HotelBuilding::createReport (
            DatePeriod & period ) const
```

Create a report for the usage of rooms for a particular period of time (ending before the today Date) in folder reports Format of the report:

Report for the usage of the rooms between <beginning of period> and <end of period>: ... Room #" <Room number> between <beginning of period> and <end of period>: <count of nights> nights.

Parameters

period	period of time

3.4.3.2 getRoomCount()

```
size_t HotelBuilding::getRoomCount ( ) const  [inline]
```

Get the room count.

Returns

size_t count of rooms

3.4.3.3 newDate()

```
void HotelBuilding::newDate (
            Date d )
```

update this HotelBuilding rooms data on a new Date

Parameters

d	new Date

3.4.3.4 operator[]()

```
Room * HotelBuilding::operator[] (
            unsigned roomNumber ) const
```

seek for a room with particular number

Parameters

roomNumber	number if the sought room

Returns

Room∗ if found -> pointer to this Room else -> nullptr

3.4.3.5 showAvailableRooms()

```
void HotelBuilding::showAvailableRooms (
            std::ostream & os,
            Date d ) const
```

show available rooms on a particular Date

Parameters

os	output stream where the available rooms will be shown in format: Available rooms for <today>: Number: <room number> Bed count: <count of beds> ...
d	Date

3.4.3.6 showRoomForNights()

```
void HotelBuilding::showRoomForNights (
            unsigned number,
            unsigned nights,
            Date today ) const
```

show soonest period of particular nights when particular room is free

Parameters

number	ID of the room
nights	length of the period
today	today's Date

3.4.3.7 showRoomsStatesToday()

```
void HotelBuilding::showRoomsStatesToday (
            Date today ) const
```

prints to stdout all rooms together with up to today Date info about their Availability (now, in future or in the past)

Parameters

today	Date

3.4.3.8 suggestRoom()

```
void HotelBuilding::suggestRoom (
            unsigned beds,
            const DatePeriod & period )
```

show top DISPLAY (or all rooms if less than DISPLAY + 1) sorted by suitabilty for the guest, given minimal number of beds and particular period of time for an eventual reservation

Parameters

beds	minimal number of beds insisted in the room
period	period of time

The documentation for this class was generated from the following files:

- HotelBuilding.hpp
- HotelBuilding.cpp

3.5 Reservation Class Reference

Class representing information about a reservation.

`#include <Reservation.hpp>`

Public Member Functions

- Reservation (std::string name, const DatePeriod &p, std::string n="None.\n", bool s=false)

  Construct a new Reservation object.
- **Reservation** (const Reservation &)=delete
- Reservation & **operator=** (const Reservation &)=delete

- bool isActive () const

  see if this Reservation is active (today is part of the period)
- bool isPast () const

  see if this Reservation is past (today is after end of period)
- bool isServiced () const

  see if this Reservation is a maintenance
- Date getFrom () const

  get beginning Date of this Reservation
- Date getTo () const

  get end Date of this Reservation
- unsigned getNights () const

  get count of nights of this Reservation
- std::string getNote () const

  get the note to this Reservation
- void onDate (Date d)

  update the state of the reservation based on new today's Date (d)
- ReservationState stateOnDate (Date) const

  see what would the state of this Reservation be on particular Date
- bool LeavingInAdvance (Date)

  try to change end of period for earlier end of this Reservation

3.5.1 Detailed Description

Class representing information about a reservation.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 Reservation()

```
Reservation::Reservation (
            std::string name,
            const DatePeriod & p,
            std::string n = "None.\n",
            bool s = false )
```

Construct a new Reservation object.

Parameters

name	of the reserver
p	
n	note left for the reservation
s	whether it is reservation or maintenance

3.5.3 Member Function Documentation

3.5.3.1 getFrom()

```
Date Reservation::getFrom ( ) const  [inline]
```

get beginning Date of this Reservation

Returns

Date beginning of the Reservation

3.5.3.2 getNights()

```
unsigned Reservation::getNights ( ) const  [inline]
```

get count of nights of this Reservation

Returns

unsigned count of nights of this Reservation

3.5.3.3 getNote()

```
std::string Reservation::getNote ( ) const  [inline]
```

get the note to this Reservation

Returns

std::string

3.5.3.4 getTo()

```
Date Reservation::getTo ( ) const  [inline]
```

get end Date of this Reservation

Returns

Date end of the Reservation

3.5.3.5 isActive()

```
bool Reservation::isActive ( ) const  [inline]
```

see if this Reservation is active (today is part of the period)

Returns

true Reservation is active
false Reservation is not active (past or future)

3.5.3.6 isPast()

```
bool Reservation::isPast ( ) const  [inline]
```

see if this Reservation is past (today is after end of period)

Returns

true Reservation is past
false Reservation is not past (active or future)

3.5.3.7 isServiced()

```
bool Reservation::isServiced ( ) const  [inline]
```

see if this Reservation is a maintenance

Returns

true this Reservation is a maintenace
false this Reservation is for a guest

3.5.3.8 LeavingInAdvance()

```
bool Reservation::LeavingInAdvance (
            Date newTo )
```

try to change end of period for earlier end of this Reservation

Returns

true end date modified for leaving in advance
false new leaving Date not appropriate for earlier leaving

3.5.3.9 onDate()

```
void Reservation::onDate (
            Date d )
```

update the state of the reservation based on new today's Date (d)

Parameters

d	new today's Date

3.5.3.10 stateOnDate()

```
ReservationState Reservation::stateOnDate (
            Date d ) const
```

see what would the state of this Reservation be on particular Date

Returns

ReservationState state on desired Date

The documentation for this class was generated from the following files:

- Reservation.hpp
- Reservation.cpp

3.6 Room Class Reference

Class representing a room in hotel.

```
#include <Room.hpp>
```

Public Member Functions

- Room (unsigned n, unsigned bC)
  Construct a new Room object.
- **Room** (const Room &)=delete
  forbidden copying of rooms
- Room & **operator=** (const Room &)=delete
  forbidden copying of rooms
- ∼**Room** ()
  Destroy the Room object.
- unsigned **getNumber** () const
- unsigned **getBedCount** () const
- bool **isFreeNow** () const
- bool freeRoom (Reservation ∗&currentRes)
  try to free this room
- void **changeLeaving** (Reservation ∗, Date newDate)
- void **newDate** (Date)
  apply new Date to state of all reservations and respectively of the room availability
- bool isFreeOnDate (Date) const
  see if this room is free in certain date

- bool isFreeInPeriod (const DatePeriod &period) const

  see if this Room is free in particular period of time (it is free in all days of the period)
- bool showReservationsInPeriod (std::ostream &os, const DatePeriod &period) const

  print to output stream info about the number of nights (if positive) in a period this Room has been taken
- bool addReservation (std::string name, std::string note, const DatePeriod &period)

  try to add Reservation to this Room
- bool closeForService (std::string note, const DatePeriod &period)

  try to add Reservation (about a maintenance) to this Room
- void **showActivity** () const

  print to stdout information about this Room latest busyness

3.6.1 Detailed Description

Class representing a room in hotel.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 Room()

```
Room::Room (
            unsigned n,
            unsigned bC )
```

Construct a new Room object.

Parameters

n	number of constructed Room
bC	number of beds in constructed Room

3.6.3 Member Function Documentation

3.6.3.1 addReservation()

```
bool Room::addReservation (
            std::string name,
            std::string note,
            const DatePeriod & period )
```

try to add Reservation to this Room

Parameters

name	name of the guest
note	note to this Reservation
period	period of time

Returns

true successfully added Reservation

false adding a Reservation failed (the room is not free in this DatePeriod)

3.6.3.2 closeForService()

```
bool Room::closeForService (
            std::string note,
            const DatePeriod & period )
```

try to add Reservation (about a maintenance) to this Room

Parameters

note	note to this maintenance
period	period of time

Returns

true successfully added maintenance

false adding a maintenance failed (the room is not free in this DatePeriod)

3.6.3.3 freeRoom()

```
bool Room::freeRoom (
            Reservation *& currentRes )
```

try to free this room

Returns

true sucesfully freed room

false room is already free

3.6.3.4 isFreeInPeriod()

```
bool Room::isFreeInPeriod (
            const DatePeriod & period ) const
```

see if this Room is free in particular period of time (it is free in all days of the period)

Parameters

period	period of time

Returns

true the room is free (in all days of the period)

false the room is not free (there is a day in period when the room is taken)

3.6.3.5 isFreeOnDate()

```
bool Room::isFreeOnDate (
            Date d ) const
```

see if this room is free in certain date

Returns

true the room is free

false the room is taken

3.6.3.6 showReservationsInPeriod()

```
bool Room::showReservationsInPeriod (
            std::ostream & os,
            const DatePeriod & period ) const
```

print to output stream info about the number of nights (if positive) in a period this Room has been taken

Parameters

os	output stream
period	period of time

Returns

true there has been taken for at least one night and info has been printed

false the room has been free during this period and no info has been printed

The documentation for this class was generated from the following files:

- Room.hpp
- Room.cpp

3.7 RoomAnalyzer Class Reference

supporting class to perform algorythms on the rooms in a building

`#include <RoomAnalyzer.hpp>`

Static Public Member Functions

- static void suggest (HotelBuilding &hB, unsigned beds, DatePeriod period)

  print top DISPLAY rooms info based on suitability of a Room (desired number of beds and period of time)
- static void soonestFreePeriod (const HotelBuilding &hB, unsigned number, unsigned nights, Date today)

  print soonest period when a particular room is free for particular number of nights

3.7.1 Detailed Description

supporting class to perform algorythms on the rooms in a building

3.7.2 Member Function Documentation

3.7.2.1 soonestFreePeriod()

```
void RoomAnalyzer::soonestFreePeriod (
            const HotelBuilding & hB,
            unsigned number,
            unsigned nights,
            Date today )  [static]
```

print soonest period when a particular room is free for particular number of nights

Parameters

hB	HotelBuilding
number	ID of the Room
nights	length of a period
today	today's Date

3.7.2.2 suggest()

```
void RoomAnalyzer::suggest (
            HotelBuilding & hB,
```

```
unsigned beds,
DatePeriod period )  [static]
```

print top DISPLAY rooms info based on suitability of a Room (desired number of beds and period of time)

Parameters

hB	HotelBuilding
beds	desired number of beds
period	period of time

The documentation for this class was generated from the following files:

- RoomAnalyzer.hpp
- RoomAnalyzer.cpp

Chapter 4

File Documentation

4.1 Date.hpp

```
1 #ifndef __DATE_HPP
2 #define __DATE_HPP
3 #include <iostream>
4 #include <ctime>
5
10 const unsigned daysFromBeginning[] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334};
11
12
17 class Date
18 {
23     unsigned short day, month, year;
30     bool isVaid() const;
38     bool isLeap(unsigned y) const;
39
40 public:
48     Date(unsigned short d = 1, unsigned short m = 1, unsigned short y = 1900) : day(d), month(m), year(y)
       {}
56     bool operator<(const Date other) const;
60     bool operator<=(const Date other) const;
68     bool operator>(const Date other) const;
72     bool operator>=(const Date other) const;
80     bool operator==(const Date other) const;
87     const char *operator()(char *buf) const;
94     int operator-(Date other) const;
100     Date &operator++();
101
107     static Date getToday();
108
116     friend std::istream &operator»(std::istream &is, Date &d);
124     friend std::ostream &operator«(std::ostream &os, const Date &d);
125 };
126
131 struct DatePeriod
132 {
137     Date from;
138
143     Date to;
144
150     unsigned length() const { return to - from; }
151
157     DatePeriod &operator++();
158
163     void readProper();
164 };
165
173 std::istream &operator»(std::istream &is, DatePeriod &dP);
174
175 #endif
```

4.2 Hotel.hpp

```
1 #ifndef __HOTEL_HPP
2 #define __HOTEL_HPP
3 #include "Types.hpp"
```

```
4 #include "Date.hpp"
5 #include "Room.hpp"
6 #include "Reservation.hpp"
7 #include "HotelBuilding.hpp"
8 #include <string>
9
17 std::string readFromIfstream(std::ifstream &ifs, size_t len);
18
23 class Hotel
24 {
29     std::string name;
30
35     static Date now;
36
41     HotelBuilding *building;
42
43 public:
44     Hotel() = delete;
45
51     Hotel(std::string hotelDataFile);
52     Hotel(const Hotel &) = delete;
53     Hotel &operator=(const Hotel &) = delete;
54
59     ~Hotel();
60
66     static Date today() { return now; }
67
73     std::string getName() const { return name; }
74
79     void nextDay();
80
92     bool reserveRoom(unsigned number, const DatePeriod &period, std::string name = "-", std::string note
       = "None\n");
93
99     Hotel &showAvailableRooms(std::ostream &, Date);
100
108     bool freeRoom(unsigned number);
109
116     Hotel &getReport(DatePeriod &period);
117
124     void searchRoom(unsigned minBeds, const DatePeriod &period) const;
125
135     bool serviceRoom(unsigned number, const DatePeriod &period, std::string note);
136
142     Hotel &showToday();
143
151     Hotel &seeRoomForNights(unsigned number, unsigned nights);
152 };
153
154 #endif
```

4.3 HotelBuilding.hpp

```
1 #ifndef __HOTELBUILDING_HPP
2 #define __HOTELBUILDING_HPP
3 #include "Types.hpp"
4 #include "Room.hpp"
5 #include "RoomAnalyzer.hpp"
6 #include <fstream>
7
12 class HotelBuilding
13 {
18     Room **rooms;
23     size_t size;
24
25 public:
35     HotelBuilding(std::ifstream &ifs);
36     HotelBuilding(const HotelBuilding &other) = delete;
37     HotelBuilding &operator=(HotelBuilding &other) = delete;
42     ~HotelBuilding();
43
49     size_t getRoomCount() const { return size; }
50
58     Room *operator[](unsigned roomNumber) const;
59
65     void newDate(Date d);
66
77     void showAvailableRooms(std::ostream &os, Date d) const;
78
89     void createReport(DatePeriod &period) const;
90
97     void suggestRoom(unsigned beds, const DatePeriod &period);
98
```

```
104     void showRoomsStatesToday(Date today) const;
105
113     void showRoomForNights(unsigned number, unsigned nights, Date today) const;
114
119     friend class RoomAnalyzer;
120 };
121
122 #endif
```

4.4 Reservation.hpp

```
1 #ifndef __RESERVATION_HPP
2 #define __RESERVATION_HPP
3 #include "Types.hpp"
4 #include "Room.hpp"
5 #include "Date.hpp"
6 #include "Hotel.hpp"
7 #include <cstring>
8 #include <cassert>
9 #include <fstream>
10 #include <string>
11
16 enum ReservationState
17 {
18     UNKNOWN = 0,
19     PAST,
20     ACTIVE,
21     FUTURE
22 };
23
28 class Reservation
29 {
34     std::string guestName;
39     std::string note;
44     DatePeriod period;
49     ReservationState state;
54     bool service;
55
56 public:
65     Reservation(std::string name, const DatePeriod &p, std::string n = "None.\n", bool s = false);
66     Reservation(const Reservation &) = delete;
67     Reservation &operator=(const Reservation &) = delete;
68
75     bool isActive() const { return state == ACTIVE; }
76
83     bool isPast() const { return state == PAST; }
84
91     bool isServiced() const { return service; }
92
98     Date getFrom() const { return period.from; }
99
105     Date getTo() const { return period.to; }
106
112     unsigned getNights() const { return period.length(); }
113
119     std::string getNote() const { return note; }
120
126     void onDate(Date d);
127
133     ReservationState stateOnDate(Date) const;
134
141     bool LeavingInAdvance(Date);
142 };
143
149 std::ostream &operator«(std::ostream &, const Reservation &);
150
151 #endif
```

4.5 Room.hpp

```
1 #ifndef __ROOM_HPP
2 #define __ROOM_HPP
3 #include <iostream>
4 #include <string>
5 #include "Types.hpp"
6 #include "Reservation.hpp"
7 #include "Hotel.hpp"
8
13 const size_t INIT_CAPACITY = 2;
```

```
14
15
20 class Room
21 {
26     unsigned number;
31     unsigned bedCount;
36     Reservation **reservations;
41     size_t resCount, resCapacity;
42
47     Reservation **pastReservations;
52     size_t pastCount, pastCapacity;
53
58     void expand(Reservation **&arr, size_t &size, size_t &capacity);
63     void shrink(Reservation **&arr, size_t &size, size_t &capacity);
64
65     unsigned daysTakenInPeriod(const DatePeriod &period) const;
66
67     bool newReservation(std::string name, std::string note, const DatePeriod &period, bool service);
68
69     void moveToPast();
70
71 public:
78     Room(unsigned n, unsigned bC);
83     Room(const Room &) = delete;
88     Room &operator=(const Room &) = delete;
93     ~Room();
94
95     unsigned getNumber() const { return number; }
96     unsigned getBedCount() const { return bedCount; }
97     bool isFreeNow() const;
98
105     bool freeRoom(Reservation *&currentRes);
106
107     void changeLeaving(Reservation *, Date newDate); // todo must be private
108
113     void newDate(Date);
114
121     bool isFreeOnDate(Date) const;
122
130     bool isFreeInPeriod(const DatePeriod &period) const;
131
140     bool showReservationsInPeriod(std::ostream &os, const DatePeriod &period) const;
141
151     bool addReservation(std::string name, std::string note, const DatePeriod &period);
152
161     bool closeForService(std::string note, const DatePeriod &period);
162
167     void showActivity() const;
168 };
169
177 std::ostream &operator«(std::ostream &os, const Room &R);
178
179 #endif
```

4.6 RoomAnalyzer.hpp

```
1 #ifndef __ROOMANALYZER_HPP
2 #define __ROOMANALYZER_HPP
3 #include "Types.hpp"
4 #include "HotelBuilding.hpp"
5 #include "Date.hpp"
6
11 const size_t DISPLAY = 5;
12
13
18 class RoomAnalyzer
19 {
27     static void sortRooms(HotelBuilding &hB, unsigned *score, size_t size);
28
29 public:
37     static void suggest(HotelBuilding &hB, unsigned beds, DatePeriod period);
38
47     static void soonestFreePeriod(const HotelBuilding &hB, unsigned number, unsigned nights, Date today);
48 };
49
50 #endif
```

4.7 Types.hpp

```
1 #ifndef __TYPES_HPP
```

```
2 #define __TYPES_HPP
3
4 class Date;
5 class Room;
6 class HotelBuilding;
7 class Reservation;
8 class RoomAnalyzer;
9 class Hotel;
10
11 #endif
```

Index