TAMK HOTEL PROGRAM

By Tomás Glavina | Group 21I260EA

Welcome to my Hotel Room Reservation project, I aim to the 5 so the program will be according to the requirements stated in moodle. As well, I will add some functionalities. The program works as a Hotel program where the user gets the option to reserve a room, the user can check before hand if the specific room is available and also the user gets the choice to check out, if the user knows the booking number reservation, which is printed in the invoice (it is assumed in this project that the user pays automatically the invoice when checking out) and lastly, the user can print the invoice before the check out, if he/she wishes so.

The copy-paste of the program copied the format of the header file but not from the cpp one, I am sorry about that.

Although it is explained in the code, I decided to use vectors because I wanted to use a different random number of rooms each time the program was opened. The code is divided in many functions, to help the visualization of it when reading it. I put all the prototyping in one header file, because since my project is just for this class, I don't think they will have any functionality outside of here. I would have liked to create more functions that complete one and only one task, but I am happy with the result of this one.

In a nutshell, the code works as follows: it first do the preparations, creating a seed for the random rooms, then creating the vectors and giving these vectors the same amount of elements as the random number, setting up the prices, room types etc. The customer elements at this point are empty and will only get values when an actual customer reserves a room.

After the preparations, the main function calls the start function, taking the hotel and customer vectors as parameters. The start function prints some welcome message and then starts the big loop that keeps the program running. It is an infinite loop that it only stops either if the amount of free rooms is 0 (gets full) or if the user presses 0. For this project it is assumed that this program is an ongoing program and each time the loop repeats it is talking to a new user, although a customer is allowed to reserve again, it will still be treated as a new object and not the same one with two rooms. The start functions first calls the hotel availability and then, if it's available, calls the choose_option() function, which returns an integer of the selected option. This function only takes the input but it doesn't handle it, that goes to the option_handler() which checks which option has been selected and then calls said function, either reserve a room, check a room, print invoice or checkout. I hope this program meets the criteria for a 5, I did my best to get a decent program, that is easy to read, as well commented as I could and the command line user interface is very intuitive and almost graphical(I hope you like the invoice format). Thank you very much for this course, it has been a pleasure to learn in here.

ClassAssigment.h

```
#pragma once
using namespace std;

//I would have liked to create classes for this but I don't know perfectly how they work in C++, so I
choose to go with structs.
struct Customer;
struct Room; //Instead of making a hotel struct, I made a Room, which is also a good struct, and the hotel
can be just a vector of rooms.

//Here are all my functions for this program.
int choose_option(); //This function returns the selected option from the user
void start(vector<Room>&, vector<Customer>&); //Starts the program with a loop, which exits either when
customer enters 0 or all rooms are full.
void option_handler(vector<Room>&, vector<Customer>&, int , int);
void reserve_room(vector<Room>&, vector<Customer>&, int ); //main function that reserves rooms
bool is_room_available(vector<Room> , int); //room checker, returns true if the room is free
bool is_hotel_empty(vector<Room>); //returns true if all the rooms are free
int select_room_type(); //returns the type of room 1(single) or 2(double).
void checkout(vector<Room>&, vector<Customer>&); //checks out the customer, prints invoice and sets the
room free.
int checkHotelAvailability(vector<Room>); //checks at the beginning of the program if there is any room
available
void check_room(vector<Room>); //returns true if the specific room is available.
void print_invoice(vector<Room>, vector<Customer>, int); //prints an invoice with all the information of
the reservation.
float check_discount(Room, Customer); //returns a discount of 0, 10, 15 or 20% if it matches the criteria.
float price_calculator(float, int, float); //returns the net price (I believe there is no need for a gross
price function since it's so basic).
```

ClassAssigment.cpp

```
#include <string>
#include <iostream>
#include <vector>
#include <cstdlib>
#include "ClassAssigment.h"
using namespace std;

//In the header file there is explanations about the structs, although the names are very self-explanatory.
struct Customer {
    string name;
    int booking_number;
    int room_number;
    int nights_staying;
};

struct Room {
    int number;
    int room_type;
    float price;
    bool free;
    Customer customer;
};

bool is_hotel_empty(vector<Room> hotel) {
    int rooms_taken = 0;    //starts with the assumption that hotel is empty
    for (int i = 0; i < hotel.size(); i++) {    //loop, checking every room if they are taken or free
        if (hotel[i].free == false) {
            rooms_taken++;
        }
    }
    if (rooms_taken == 0) return true;     // if rooms_taken is still zero, hotel is empty
    return false;
}

int checkHotelAvailability(vector<Room> hotel) {        // this one returns the number of rooms available. If its 0, then the hotel is full, of course.
    int rooms_available = 0;
    for (int i = 0; i < hotel.size(); i++) {
        if (hotel[i].free == true) {
```

```
            rooms_available++;
        }
    }
    return rooms_available;
}

float check_discount(Room room, Customer customer) {
    /*The discounts go in this order --> 0-4 nights = 0%, 5-9 nighs = 5% and 10+ nights = 10%
    If the room is double, then the discount is double.
    Returns the discount in float type (10% = 0.1; 20% = 0.2)
    */
    float discount;

    if (customer.nights_staying >= 10) {
        discount = 0.1;
    }
    else if (customer.nights_staying >= 5)
    {
        discount = 0.05;
    }
    else if (customer.nights_staying < 5) {
        return 0;
    }

    if (room.room_type == 2) {
        return (discount * 2);
    }

    return discount;
}

float price_calculator(float price, int nights, float discount) {     //Self explanatory, only coment is
that I casted the nights as float to get a float
    return (price * (static_cast<float>(nights)) * (1.0 - discount));
}

int select_room_type() {  // returns the type of room 1(single), 2 (double)
    int input;
    do {
        cout << "Would you like a single or a double room?" << endl
            << "Please select 1 for single room or 2 for double room." << endl
            << "Select 0 to exit." << endl;
```

```
        cin >> input;
        while (cin.fail()) {   //Again, cin.fail to check if its integer
            cout << "That's not a real number! \nPlease select 1 or 2." << endl;
            cin.clear();
            cin.ignore(256, '\n');
            cin >> input;
        }

        if (input == 0) {
            return 0;
        }

    } while (input != 1 && input != 2); //Checking that it is the correct output

    return input;

}

void reserve_room(vector<Room>& hotel, vector<Customer>& customer, int customer_number) {
/* To reserve a room, first I ask what type of room they want to take, then check if its available,
 then the user choses the room number and checks again if that room is available.
After selecting the room and checking that the type and number are free, it gives the customer a name,
room number and booking number.*/

    string name;
    int input;
    int room_number;
    int nights;
    int continue_or_not;
    int rooms_available = 0;



    //This algorithm selects the room type, as long as there is room available of said type.
    //It is a loop which continues going until process is completed, either by exiting (0) or selecting a
room type which is available



    //First we ask to input 1 or 2, whether its a single or double room.
    input = select_room_type();

    if (input == 0) { //how to exit the program
        return;
```

```
    }

    if (input == 1) {
        for (int x = 0; x < hotel.size(); x += 2) {
            if (hotel[x].free) {
                rooms_available++;
            }
        }
    }

    if (input == 2) {
        for (int x = 1; x < hotel.size(); x += 2) {
            if (hotel[x].free) {
                rooms_available++;
            }
        }
    }

    // here, after checking all rooms available of that type, if there is 0 available, the user selects
wether to continue or not.
    if (rooms_available == 0) {
        do {
            cout << "There is no room available of that type." << endl
                << "Would you like to select another type or stop?" << endl
                << "1 to continue, 0 to stop.";
            cin >> continue_or_not;
            while (cin.fail()) {
                cout << "That's not a real number! \nPlease select chose a valid room number." << endl;
                cin.clear();
                cin.ignore(256, '\n');
                cin >> continue_or_not;
            }
        } while (continue_or_not != 0 && continue_or_not != 1);
        if (continue_or_not == 0) {
            return;
        }
    }



    if (input == 0) { //if they decide not to continue
        return;
```

```
    }

    /*Then we ask for the room number, according with which room type they chose.

    In this input checking, I made an algorithm that checks if the input was indeed an integer.

    The cin.fail() algorithm is explained here and it will be repeted in every input part of this program,
so I won't repeat what it is each time.*/

    do {

        if (input == 1) {

            do {

                cout << "For convinience of the customer, single rooms are odd numbered (1, 3, 5... etc)"
<< endl

                    << "Which room would you like to reserve?" << endl;

                cin >> room_number;

                while (cin.fail()) {  //I use cin.fail() to check if the input was indeed an integer, if
not, I clear the input and ask again

                    cout << "That's not a real number! \nPlease select chose a valid room number." <<
endl;

                    cin.clear();

                    cin.ignore(256, '\n');

                    cin >> room_number;

                }

                //Since it's type 1(single), the room must be positive integer and odd.

            } while (room_number <= 0 && printf("Wrong room number, please select again.\n") ||
room_number % 2 == 0 && printf("Wrong room number, please select again.\n") || room_number > (hotel.size()
+ 1) && printf("Wrong room number, please select again.\n"));

        }

        else if (input == 2) {

            do {

                cout << "For convinience of the customer, double rooms are even nuembered (2, 4, 6...
etc)" << endl

                    << "Which room would you like to reserve?" << endl;

                cin >> room_number;

                while (cin.fail()) {

                    cout << "That's not a real number! \nPlease select chose a valid room number." <<
endl;

                    cin.clear();

                    cin.ignore(256, '\n');

                    cin >> room_number;

                }

                //Here I check that the room is actually an even number

            } while (room_number <= 0 && printf("Wrong room number, please select again.\n") ||
room_number % 2 != 0 && printf("Wrong room number, please select again.\n") || room_number > (hotel.size()
+ 1) && printf("Wrong room number, please select again.\n"));

        }

        //Here I check that the room selected is actually free or not.

    } while (!(is_room_available(hotel, room_number)) && printf("The selected room is not available,
please select another one.\n"));
```

```
    do {
        cout << "How many nights are you staying in the hotel?" << endl;
        cin >> nights;
        while (cin.fail()) {
            cout << "That's not a real number! \nPlease select chose a valid number: " << endl;
            cin.clear();
            cin.ignore(256, '\n');
            cin >> nights;
        }
    } while (nights < 1 && printf("You can't stay less than 1 night, please try again.\n"));



    //Lastly we ask for their full name
    cout << "How is your full name?" << endl;
    cin.ignore();
    getline(cin, name);


    //After all the input, we generate a random booking number and store all the information in the
customer array.
    srand((unsigned int)time(NULL));    //creating a new seed for the random booking number
    customer[customer_number].booking_number = rand() % 89999 + 10000;
    customer[customer_number].name = name;
    customer[customer_number].room_number = room_number;
    customer[customer_number].nights_staying = nights;


    hotel[room_number - 1].customer = customer[customer_number]; //Also, giving the hotel the Customer
information, for the room that he/she is staying.
    hotel[room_number - 1].free = false;


    //Here I explain that the booking number is important to checkout.
    cout << endl << "Thank you, " << hotel[room_number - 1].customer.name << ", your booking number is "
<< hotel[room_number - 1].customer.booking_number << ". Don't lose it! You need it to checkout." << endl
<< endl;
}


void check_room(vector<Room> hotel) { //checks if the room is free or not and prints it to the user. (The
hotel is not legally entitled to give any personal information, so it only prints if its free or not)
    int room_nro;


    do {
        cout << "Please enter the number of the room: (Enter 0 to exit)" << endl;
        cin >> room_nro;
```

```
        while (cin.fail()) {
            cout << "Thats not a number! \nPlease enter an integer number from 1 to " << hotel.size() <<
": ";
            cin.clear();
            cin.ignore(256, '\n');
            cin >> room_nro;
        }

        if (room_nro == 0) return;

    } while (room_nro < 1 || room_nro > hotel.size());

    if (is_room_available(hotel, room_nro)) cout << endl << "The room " << room_nro << " is free." << endl
<< endl;
    else if (!is_room_available(hotel, room_nro)) cout << endl << "The room " << room_nro << " is not
available." << endl << endl;
}

bool is_room_available(vector<Room> hotel, int room_nro) { //Returns true or false if the room is free
    if (hotel[room_nro - 1].free) return true;
    return false;
}

void checkout(vector<Room>& hotel, vector<Customer>& customer) { //checks out the customer, by setting the
room free and deleting the customer.
    int room_number;
    int booking_number;

    /*The checkout procedure is basicand straight foward, you give your booking numberand room number
     and if the customer has both the room and booking number (meaning if they match with same customer)
     the checkout starts, printing the invoice(is assumed they pay automatically), deleting the customer,
     and setting the room as free.
    */
    cout << "To check out you need to enter your booking number and room number. (0 to return)" << endl;
    do {
        cout << "Now enter your booking number:" << endl;
        cin >> booking_number;
        while (cin.fail()) {
            cout << "Thats not a number! \nPlease enter an integer number: ";
            cin.clear();
            cin.ignore(256, '\n');
            cin >> booking_number;
        }
```

```
        if (booking_number == 0) return;


    } while (booking_number < 1);


    do {
        cout << "Now enter your room number:" << endl;
        cin >> room_number;
        while (cin.fail()) {
            cout << "Thats not a number! \nPlease enter an integer number from 1 to " << hotel.size() <<
": ";
            cin.clear();
            cin.ignore(256, '\n');
            cin >> room_number;
        }


        if (room_number == 0) return;


    } while (room_number < 1 || room_number > hotel.size());


    //checking that the chosen room is actually taken.
    if (hotel[room_number - 1].free) {
        cout << endl << "There is no customer in that room." << endl << endl;
    }
    else if(hotel[room_number-1].customer.booking_number == booking_number) //checking that the booking
number of the room number entered,
    {                                                                       // matches with the booking
number entered.


        print_invoice(hotel, customer, room_number);


        customer.erase(     //deleting the customer from the vector
            remove_if(customer.begin(), customer.end(), [&](Customer const& x) {
                return x.room_number == room_number;
            }),
        customer.end());
        customer.push_back(Customer()); //adding an empty customer to have still the same amount of
possible customers as rooms.
        hotel[room_number - 1].free = true;


        cout << endl << "Thank you for staying, we hope you had a great time.\n" << endl;
    }
    else {
        cout << endl << "ERROR: booking number doesn't match with the bookoing number of the room
entered.\n" <<
```

```
            "Returning to the option selection." << endl << endl;
    }

}

void print_invoice(vector<Room> hotel, vector<Customer> customer, int room) {
    //Calculating discount and gross price first for easier cout
    //Printing prices in $ because € doesnt print correctly(at least in my computer)
    float discount = check_discount(hotel[room - 1], hotel[room - 1].customer);
    float gross_price = hotel[room - 1].price * (static_cast<float>(hotel[room -
1].customer.nights_staying));


    if (hotel[room - 1].free) {     //first I check if the room is actually taken or not
        cout << endl << "No one is staying in that room." << endl;
        return;
    }
    //if it is taken, I proceed to print the invoice for the person staying there.
    cout << "\n-------------------------------------------------" << endl
        << "INVOICE \nTAMK Hotel OY\nKuntokatu 3, \n33520 Tampere" << endl
        << "\nBill to: " << hotel[room - 1].customer.name << "     Booking number: " << hotel[room -
1].customer.booking_number << endl
        << "Room: " << room << "        Room type: "<< hotel[room-1].room_type << "      Nights: " <<
hotel[room-1].customer.nights_staying
        << endl << endl <<"SUBTOTAL: $"<< gross_price
        << endl << "DISCOUNT: " << discount*100 <<"%"
        << endl << "TOTAL: $" << price_calculator(hotel[room-1].price, hotel[room -
1].customer.nights_staying, discount)<< endl
        << "-------------------------------------------------" << endl;


}

int choose_option() {
    int option;
    do {
        cout << "What would you like to do? (Enter the number)" << endl
            << "1. Reserve a room." << endl
            << "2. Search a specific room." << endl
            << "3. Checkout." << endl
            << "4. Print invoice." << endl
            << "0. Exit." << endl;
        cin >> option;
        while (cin.fail()) {
            cout << "That's not a real number! \nPlease select chose a valid room number." << endl;
```

```
            cin.clear();
            cin.ignore(256, '\n');
            cin >> option;
        }
    } while (option < 0 || option > 4);

    return option;
}

void option_handler(vector<Room>& hotel, vector<Customer>& customer, int selected_option, int
customer_number) {
    int room_number;

    switch (selected_option) {
    case 1:
        reserve_room(hotel, customer, customer_number);
        break;
    case 2:
        check_room(hotel);
        break;
    case 3:
        checkout(hotel, customer);
        break;
    case 4:

        //for this option, I decided to have some code here, because the invoice print function is also
used for checkout
        //so I left print_invoice very basic and functional and put some code here.
        if (is_hotel_empty(hotel)) {
            cout << "No customers int the hotel to make invoice at the moment.\n";
            break;
        }

        do {
            cout << "Enter the room number, please: (0 to return)\n";
            cin >> room_number;
            while (cin.fail()) {
                cout << "That's not a real number! \nPlease select chose a valid room number." << endl;
                cin.clear();
                cin.ignore(256, '\n');
                cin >> room_number;
            }
```

```
        } while (room_number < 0 || room_number > hotel.size());

        if (room_number == 0) {
            break;
        }

        print_invoice(hotel, customer, room_number);
        break;
    }
}

void start(vector<Room>& hotel, vector<Customer>& customer) {
    //Starts the program and handles all the function callings.
    int selected_option;
    int customer_number = 0;

    cout  << "Welcome to TAMK's hotel online service!" << endl
        << "Our hotel consists of " << hotel.size() << " rooms, we currently have single and double
rooms." << endl
        << "Our prices are: $100 per night for the single room and for the double room is 150$ per night."
<< endl;

    //An infinite loop that only breaks if the hotel is full or the user selects 0.
    while (true) {

        //While there is rooms available, you can continue reserving
        // by calling the chose_option()
        if (checkHotelAvailability(hotel) != 0) selected_option = choose_option();

        //but if hotel is full(the function returns the number of free rooms), then the program ends.
        if (checkHotelAvailability(hotel) == 0) {
            cout << "We are very sorry, the hotel is full at the moment. Come back another day. Thank
you." << endl;
            break;
        }
        if (selected_option == 0) break;

        //after the option is selected, its handled in this function
        option_handler(hotel, customer, selected_option, customer_number);

        //this is just a method I made to handle customers, each time the loop goes through
        customer_number++;
```

```
    }
}


int main()
{
    //Initialize hotel and customer lists as a vector and use push_back to get more rooms, instead of array.
    // To get the amount of rooms even I multiply the random number given between 20 - 40 by two.
    // The rooms go as it follows --> if index is the Room Number is even, is a double room, if its odd its a single room.

    srand((unsigned int)time(NULL));                    //setting the seed so I get as many random options as possible
    int amount_of_rooms = 2 * (rand() % 20 + 20);       //random amount of rooms

    vector<Customer> customer;
    vector<Room> hotel;


    for (int i = 0; i < amount_of_rooms; i++) {         // using the random number as the top limit of the vector
        hotel.push_back(Room());                        // with a vector I have to push back to add a Room
        customer.push_back(Customer());                 //Customer list gets as big as the hotel rooms, because eventually there will be as many customers as rooms

        hotel[i].number = i + 1;
        if (hotel[i].number % 2 == 0) {                 // if room number is even it gets the double type and 150€
            hotel[i].room_type = 2;
            hotel[i].price = 150;
            hotel[i].free = true;
        }
        else {                                          // if its odd, it gets the single type room and 100€ price
            hotel[i].room_type = 1;
            hotel[i].price = 100;
            hotel[i].free = true;
        }
    }


    //Now we start the user interface.
    //Passing the hotel vector by reference to be able to modify it.
    start(hotel, customer);
```

```
    return 0;

}
```