**ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ**

«М.Х. ДУЛАТИ АТЫНДАҒЫ ТАРАЗ ӨҢІРЛІК УНИВЕРСИТЕТ»

КОММЕРЦИЯЛЫҚ ЕМЕС АКЦИОНЕРЛІК ҚОҒАМЫ

Технологиялық **факультеті**

Қолданбалы информатика және бағдарламалау **кафедрасы**

**Оқу практика бойынша есебі**

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тобы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/аты-жөні/ /қолы/

Жетекші **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

/қызметі/ /аты-жөні/ /қолы/

Қорғалды «\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_20\_\_ж. бағасы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/жазбаша/

Комиссия мүшелері: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/аты-жөні/ /қолы/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/аты-жөні/ /қолы/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/аты-жөні/ /қолы/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/аты-жөні/ /қолы/

Өндіріс жетекшісі **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

/қызметі, аты-жөні/ /қолы/

Өндіріс М.О.

##### Тараз 2024

**Мазмұны**

Кіріспе………………………………………………………………………….......3

Теориялық бөлім …………...…………………………………………..….4

Бағдарламада қолданылған С++ тілінің технологиялары…......................……………...….......................5

Жеке жұмыс.........................................................................6

**КІРІСПЕ**

***Менің тәжірибелік жұмысымның тақырыбы С++ тіліндегі негізгі технологиялармен танысып оны практика жүзінде қолдану***

*Жеке жұмыс :* ***Hotel-ді немесе Hostel-ді басқару жүйесін жасау***

*Жұмыстың өзектілігі* :

* Қонақүйде тұратын адам санының артуы: қонақүйлерді басқару жүйелері қысқа мерзімді тұруды, ұзақ мерзімді жалға алуды немесе жатақханалар, курорттық қонақүйлер, пәтерлер және т.б. сияқты әртүрлі тұру опцияларын қалайтын қонақтардың талаптарын қанағаттандыру үшін жүйе ауқымды болуы керек.
* Қызмет көрсету сапасын жақсарту және Қызметтерді жекелендіру: қонақүйлерді басқару жүйелері қонақтарға қызмет көрсетуді жекелендіру мүмкіндіктерін қолдауы керек, мысалы, қалаулар мен тұру тарихын сақтау арқылы, бұл тұтынушылардың қонақ үй брендіне деген адалдығын арттырады.
* Ресурстарды басқару және толтыруды оңтайландыру: нөмірлерді толтыруды оңтайландыру және ресурстарды тиімді басқару қонақүйлерге пайданы барынша арттыруға және шығындарды азайтуға көмектеседі, әсіресе төмен жүктеме кезеңдерінде.
* Цифрлық трансформация: онлайн брондау жүйелері, қонақтарға арналған мобильді қосымшалар, автоматтандырылған тіркеу және шығу процестері сияқты заманауи технологияларды енгізу қонақүйлерге операциялардың тиімділігін жақсартуға және өз клиенттеріне ыңғайлы және икемді қызмет көрсетуге көмектеседі.
* Қауіпсіздік : пандемияға және қауіпсіздікке байланысты оқиғаларды ескере отырып, қонақүйлерді басқару жүйелері санитарлық шаралар, кіруді бақылау және т.б. сияқты барлық қажетті қауіпсіздік стандарттарын сақтауы керек.

Осылайша, қонақүйлерді басқару жүйелері қонақжайлылық нарығының өзгеріп отыратын қажеттіліктері мен талаптарына бейімделуді жалғастыруда, қонақ үй кәсіпорындарының табысты жұмысын қамтамасыз етуде және қонақтардың қажеттіліктерін өтеуде шешуші рөл атқарады.

*Жұмыстың мақсаты*:

1. Қонақтарды басқару: қонақ үй ресурстарын оңтайлы пайдалану үшін қонақтар туралы деректерді қосу, жою және жаңарту және оларды көшіру.
2. Нөмірлерді басқару: әр қабаттағы нөмірлердің қол жетімділігі мен толықтығын бақылау.

*Міндеттері*:

* Қонақты қосу: жаңа қонақты бос бөлмеге тіркеу.
* Қонақты алып тастау: қонақ шыққаннан кейін бөлмені босату.
* Қонақ деректерін жаңарту: қонақ туралы ақпаратты өзгерту, мысалы, болу мерзімін ұзарту кезінде.
* Қонақты көшіру: қонақты бір бөлмеден екінші бөлмеге ауыстыру.
* Бөлме туралы ақпарат: әр қабаттағы қол жетімді және бос емес бөлмелерді қарау.

Теориялық бөлім:

1. Стандартты C++ кітапханалары: <string>, <iostream>, <vector>, <map> сәйкесінше жолдарды басқару, енгізу/шығару, контейнерлер және экранға шығару үшін қолданылады.
2. Санақтар (enum): бөлменің күйін анықтау үшін қолданылады (бос немесе бос емес).
3. Құрылымдар (struct): Visitor және Room сәйкесінше қонақ пен бөлме туралы ақпаратты ұсыну үшін қолданылады.
4. Сыныптар (class): Hostel - қонақ үй мен оның ресурстарын басқаруға арналған.

Жобаны ары қарай дамыту:

1. Дерекқор интеграциясы: қонақтар мен бөлмелер туралы деректерді тиімдірек сақтау үшін.
2. Басқарудың кеңейтілген функционалдығы: бөлмелерді брондау мүмкіндігі, автоматты шот-фактура және т. б.
3. Пайдаланушы интерфейсі: қызметкерлердің жүйемен ыңғайлы өзара әрекеттесуі үшін графикалық интерфейсті дамыту.

***Жобаның аты : HostelManagmentSystem***

**#include <string>**

**#include <iostream>**

**#include <vector>**

**#include <map>**

**enum RoomState**

**{**

**AVAILABLE,**

**UNAVAILABLE,**

**};**

**std::string roomStateToStr(RoomState r)**

**{**

**switch (r)**

**{**

**case RoomState::AVAILABLE:**

**return "Свободно";**

**case RoomState::UNAVAILABLE:**

**return "Несвободно";**

**}**

**}**

**struct Visitor**

**{**

**std::string name;**

**};**

**struct Room**

**{**

**int roomNum;**

**int placeNum;**

**RoomState state;**

**Visitor visitor;**

**};**

**struct Floor**

**{**

**int floorNum;**

**std::vector<Room> rooms;**

**};**

**class Hostel**

**{**

**private:**

**std::vector<Floor> \_hostel;**

**int getUserInput(std::string msg, std::string errMsg)**

**{**

**int x;**

**while (true)**

**{**

**std::cout << msg;**

**std::cin >> x;**

**std::cin.ignore(32767, '\n');**

**if (std::cin.fail())**

**{**

**std::cin.clear();**

**std::cin.ignore(32767, '\n');**

**std::cout << errMsg;**

**continue;**

**}**

**return x;**

**}**

**}**

**void addVisitor()**

**{**

**int floorNum = -1;**

**while (floorNum > \_hostel.size() or floorNum < 1)**

**floorNum = getUserInput("Номер этажа : ", "Неверный выбор. Повторите занова!\n");**

**floorNum--;**

**int roomNum = -1;**

**while (roomNum > \_hostel[floorNum].rooms.size() or roomNum < 1)**

**roomNum = getUserInput("Номер комнаты : ", "Неверный выбор. Повторите занова!\n");**

**roomNum--;**

**if (\_hostel[floorNum].rooms[roomNum].state == RoomState::UNAVAILABLE)**

**{**

**std::cout << "Там уже живет : " << \_hostel[floorNum].rooms[roomNum].visitor.name << std::endl;**

**return;**

**}**

**std::cout << "Имя гостя : ";**

**std::string visitorName;**

**std::cin >> visitorName;**

**std::cin.ignore(32767, '\n');**

**\_hostel[floorNum].rooms[roomNum].state = RoomState::UNAVAILABLE;**

**\_hostel[floorNum].rooms[roomNum].visitor = Visitor{visitorName};**

**std::cout << "\nДобавлено Гость: " << visitorName << "  " << "[Этаж №" << floorNum + 1 << " Комната №" << roomNum + 1 << "]\n"**

**<< std::endl;**

**}**

**void removeVisitor()**

**{**

**int floorNum = -1;**

**while (floorNum > \_hostel.size() or floorNum < 1)**

**floorNum = getUserInput("Номер этажа : ", "Неверный выбор. Повторите занова!\n");**

**floorNum--;**

**int roomNum = -1;**

**while (roomNum > \_hostel[floorNum].rooms.size() or roomNum < 1)**

**roomNum = getUserInput("Номер комнаты : ", "Неверный выбор. Повторите занова!\n");**

**roomNum--;**

**if (\_hostel[floorNum].rooms[roomNum].state == RoomState::AVAILABLE)**

**{**

**std::cout << "Там никто не живет!" << std::endl;**

**return;**

**}**

**std::cout << "\nУдалено Гость: " << \_hostel[floorNum].rooms[roomNum].visitor.name << "  " << "[Этаж №" << floorNum + 1 << " Комната №" << roomNum + 1 << "]\n"**

**<< std::endl;**

**\_hostel[floorNum].rooms[roomNum].state = RoomState::AVAILABLE;**

**\_hostel[floorNum].rooms[roomNum].visitor = Visitor{""};**

**}**

**void updateVisitorData()**

**{**

**int floorNum = -1;**

**while (floorNum > \_hostel.size() or floorNum < 1)**

**floorNum = getUserInput("Номер этажа : ", "Неверный выбор. Повторите занова!\n");**

**floorNum--;**

**int roomNum = -1;**

**while (roomNum > \_hostel[floorNum].rooms.size() or roomNum < 1)**

**roomNum = getUserInput("Номер комнаты : ", "Неверный выбор. Повторите занова!\n");**

**roomNum--;**

**if (\_hostel[floorNum].rooms[roomNum].state == RoomState::AVAILABLE)**

**{**

**std::cout << "Там никто не живет!" << std::endl;**

**return;**

**}**

**std::cout << "Имя гостя : ";**

**std::string visitorName;**

**std::cin >> visitorName;**

**std::cin.ignore(32767, '\n');**

**std::cout << "\n"**

**<< \_hostel[floorNum].rooms[roomNum].visitor.name << " ---> " << visitorName << "\n"**

**<< std::endl;**

**\_hostel[floorNum].rooms[roomNum].visitor = Visitor{visitorName};**

**}**

**void moveVisitor()**

**{**

**int floorNum1 = -1;**

**while (floorNum1 > \_hostel.size() or floorNum1 < 1)**

**floorNum1 = getUserInput("Номер актуального этажа : ", "Неверный выбор. Повторите занова!\n");**

**floorNum1--;**

**int roomNum1 = -1;**

**while (roomNum1 > \_hostel[floorNum1].rooms.size() or roomNum1 < 1)**

**roomNum1 = getUserInput("Номер актуальной комнаты : ", "Неверный выбор. Повторите занова!\n");**

**roomNum1--;**

**if (\_hostel[floorNum1].rooms[roomNum1].state == RoomState::AVAILABLE)**

**{**

**std::cout << "Там никто не живет!" << std::endl;**

**return;**

**}**

**int floorNum2 = -1;**

**while (floorNum2 > \_hostel.size() or floorNum2 < 1)**

**floorNum2 = getUserInput("Номер этажа для переселение : ", "Неверный выбор. Повторите занова!\n");**

**floorNum2--;**

**int roomNum2 = -1;**

**while (roomNum2 > \_hostel[floorNum2].rooms.size() or roomNum2 < 1)**

**roomNum2 = getUserInput("Номер комнаты для переселение : ", "Неверный выбор. Повторите занова!\n");**

**roomNum2--;**

**if (\_hostel[floorNum2].rooms[roomNum2].state == RoomState::UNAVAILABLE)**

**{**

**std::cout << "Там уже живет : " << \_hostel[floorNum2].rooms[roomNum2].visitor.name << std::endl;**

**return;**

**}**

**std::cout << "\nПереселяем Гость: " << \_hostel[floorNum1].rooms[roomNum1].visitor.name << "  "**

**<< "[Этаж №" << floorNum1 + 1 << " Комната №" << roomNum1 + 1 << "] ----> "**

**<< "[Этаж №" << floorNum2 + 1 << " Комната №" << roomNum2 + 1 << "]\n"**

**<< std::endl;**

**Visitor vis = \_hostel[floorNum1].rooms[roomNum1].visitor;**

**\_hostel[floorNum1].rooms[roomNum1].state = RoomState::AVAILABLE;**

**\_hostel[floorNum1].rooms[roomNum1].visitor = Visitor{""};**

**\_hostel[floorNum2].rooms[roomNum2].state = RoomState::UNAVAILABLE;**

**\_hostel[floorNum2].rooms[roomNum2].visitor = vis;**

**}**

**public:**

**Hostel(std::vector<Floor> hostel)**

**{**

**\_hostel = hostel;**

**}**

**void printHostel()**

**{**

**for (const auto &floor : \_hostel)**

**{**

**std::cout << "Этаж №" << floor.floorNum << ":\n";**

**for (const auto &room : floor.rooms)**

**{**

**std::cout << "\tКомната №" << room.roomNum << "  : "**

**<< roomStateToStr(room.state)**

**<< " [Мест в комнате - " << room.placeNum << "] "**

**<< ((room.state == RoomState::UNAVAILABLE) ? "Гость : " + room.visitor.name : "")**

**<< std::endl;**

**}**

**}**

**}**

**void managment()**

**{**

**while (true)**

**{**

**std::cout << std::endl;**

**std::cout << "1) Добавить гостя\n";**

**std::cout << "2) Удалить гостя\n";**

**std::cout << "3) Обнавить данные гостя\n";**

**std::cout << "4) Переселить гостя\n";**

**std::cout << "5) Выводить инфо о комнатах\n";**

**std::cout << std::endl;**

**int managerChoice = -1;**

**while (managerChoice > 5 or managerChoice < 0)**

**managerChoice = getUserInput("Что вы хотите сделать : ", "Неверный выбор. Повторите занова!\n");**

**switch (managerChoice)**

**{**

**case 1:**

**addVisitor();**

**break;**

**case 2:**

**removeVisitor();**

**break;**

**case 3:**

**updateVisitorData();**

**break;**

**case 4:**

**moveVisitor();**

**break;**

**case 5:**

**printHostel();**

**break;**

**}**

**}**

**}**

**};**

**int main()**

**{**

**Hostel hostel({**

**Floor{1,**

**std::vector<Room>{**

**Room{1, 1, RoomState::AVAILABLE, Visitor{""}},**

**Room{2, 2, RoomState::AVAILABLE, Visitor{""}},**

**Room{3, 2, RoomState::AVAILABLE, Visitor{""}},**

**Room{4, 3, RoomState::AVAILABLE, Visitor{""}},**

**}},**

**Floor{2,**

**std::vector<Room>{**

**Room{1, 1, RoomState::AVAILABLE, Visitor{""}},**

**Room{2, 3, RoomState::AVAILABLE, Visitor{""}},**

**}},**

**Floor{3,**

**std::vector<Room>{**

**Room{1, 2, RoomState::AVAILABLE, Visitor{""}},**

**Room{2, 2, RoomState::AVAILABLE, Visitor{""}},**

**Room{3, 2, RoomState::AVAILABLE, Visitor{""}},**

**}},**

**});**

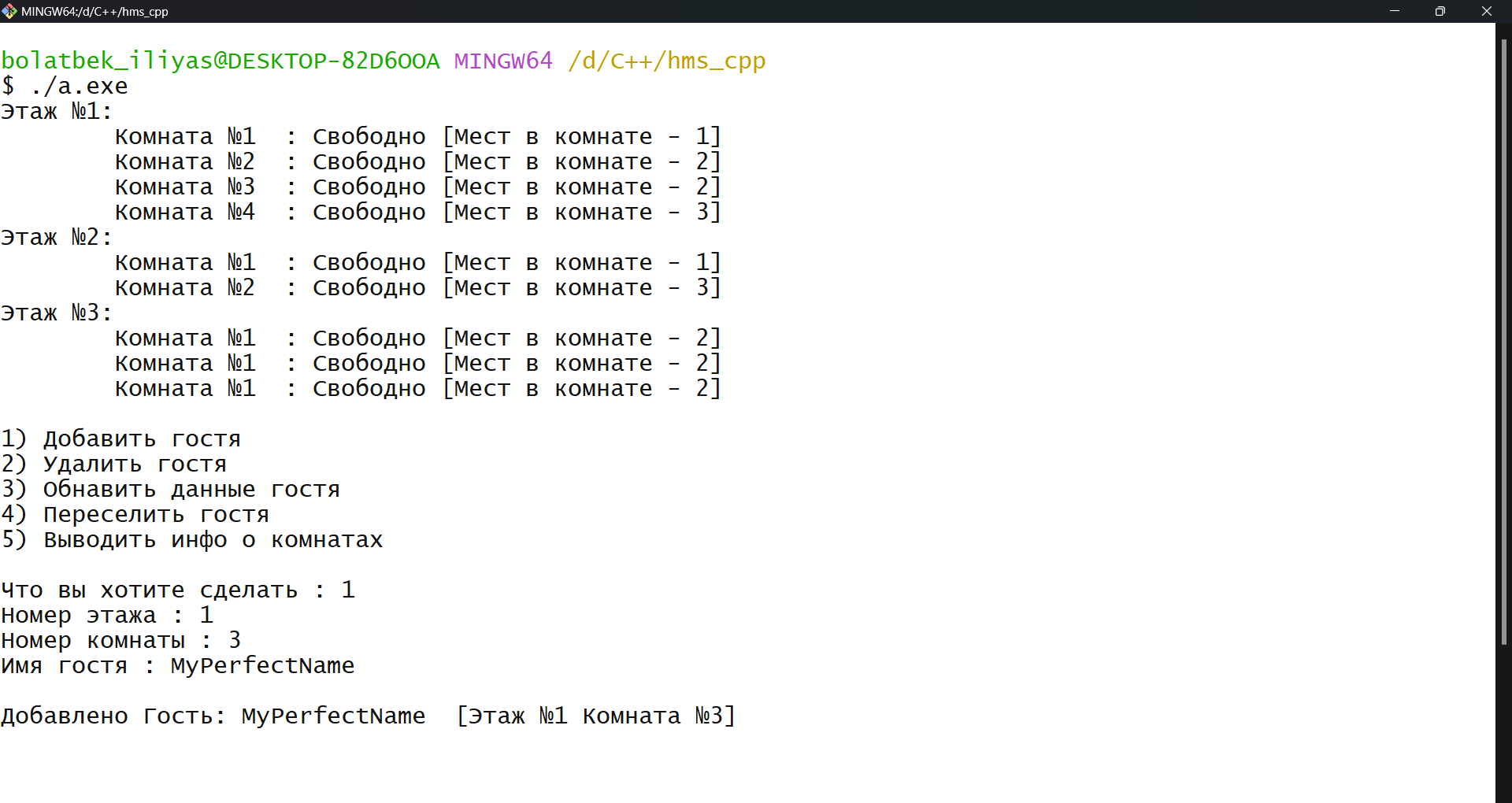
**hostel.printHostel();**

**hostel.managment();**

**}**

*Нәтижесі:*

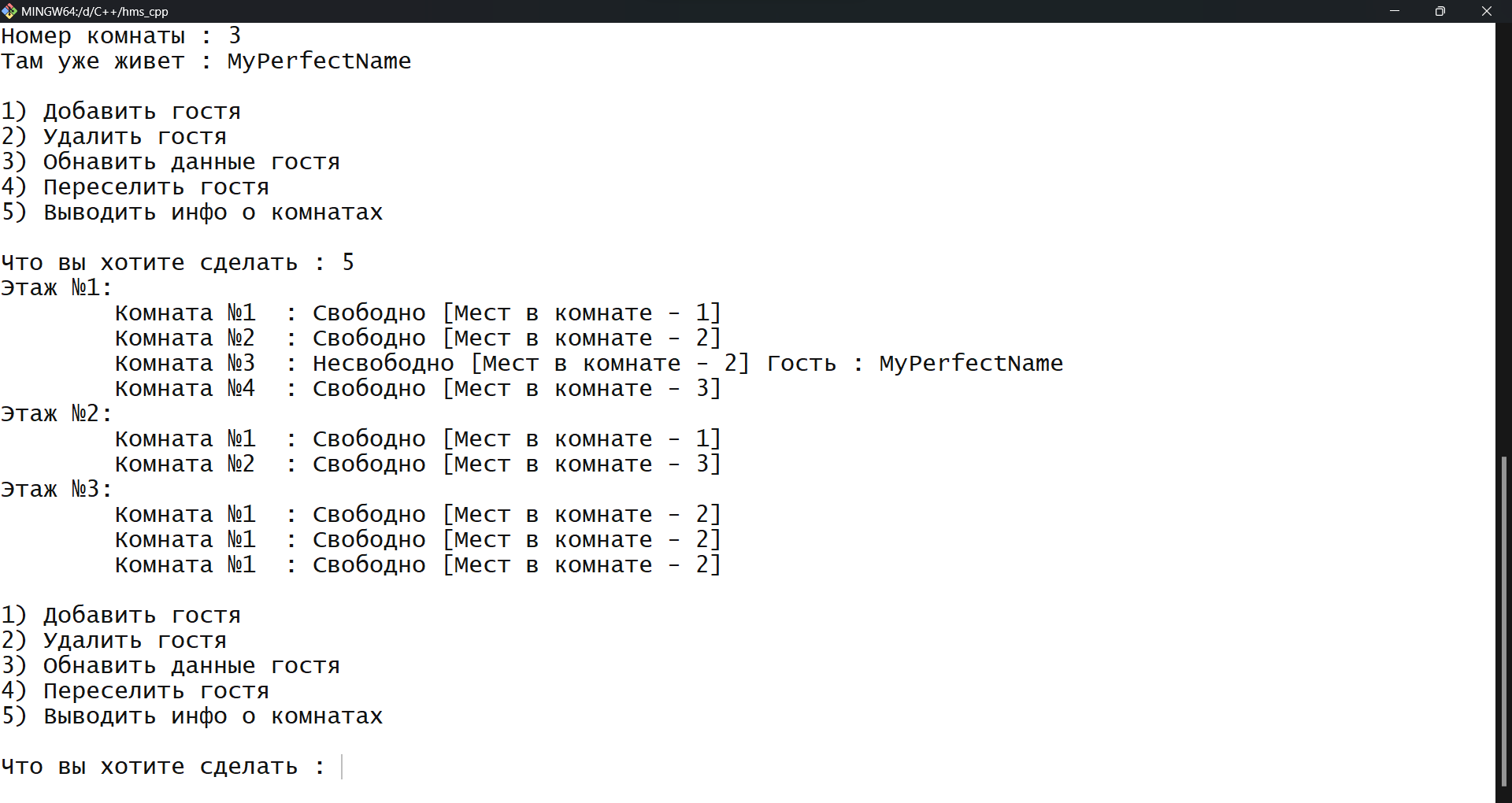
*Screenshot-1*



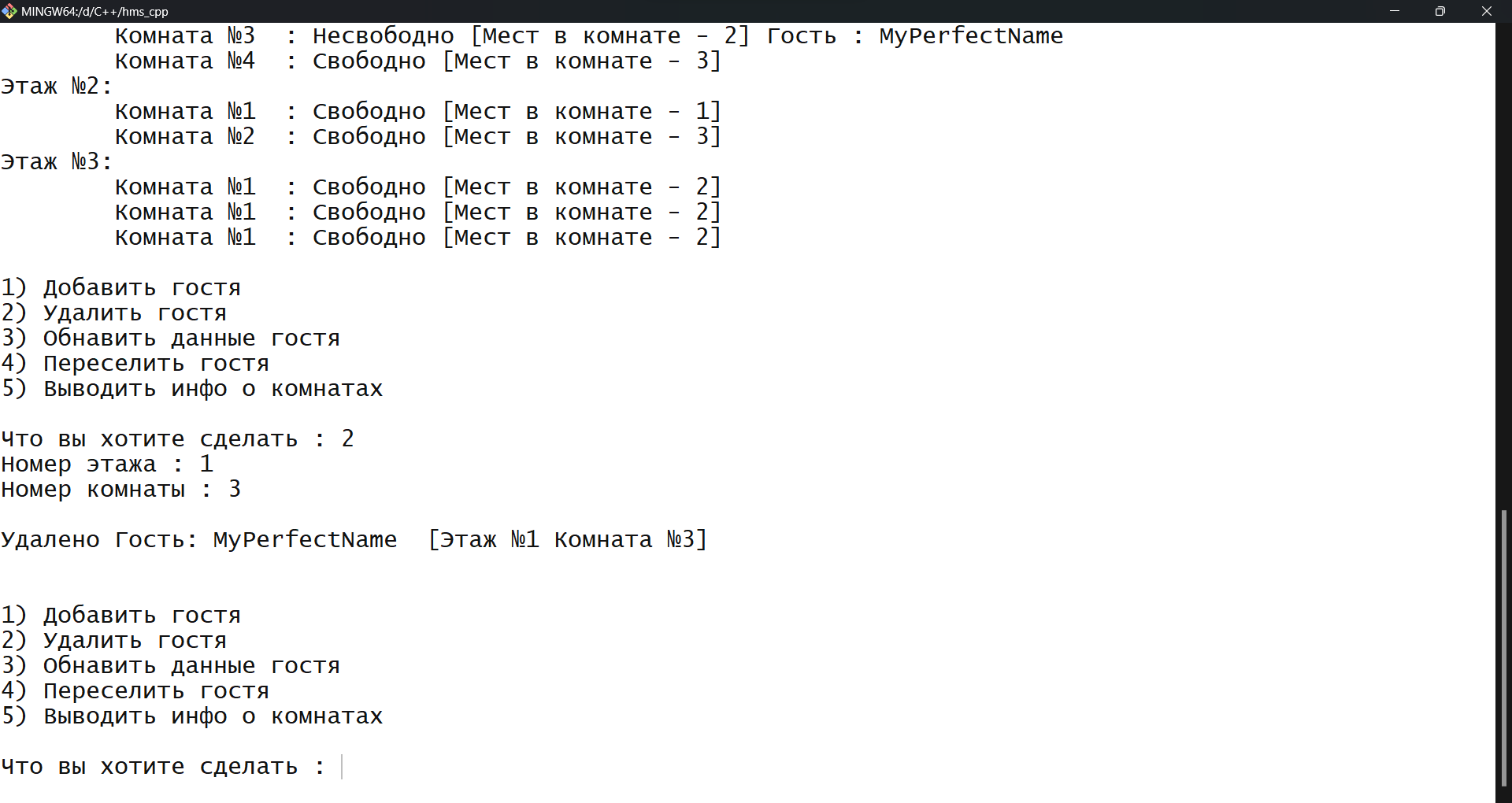
*Screenshot-2*



*Screenshot-3*



*Screenshot-4*



*Screenshot-5*

