Чёрный Алексей Владимирович  
БПИ 239-2

Вариант 16

Условие:

Задача о социалистической гостинице. В гостинице 30 одноместных номеров. Клиенты гостиницы снимают номер на одни или

несколько суток (задается при создании клиента). Если в гостинице нет свободных номеров, клиенты не уходят, а устраиваются на

рядом с гостиницей на скамейках и ждут в порядке очереди, пока любой номеров не освободится (других гостиниц в городе нет).

Создать многопоточное приложение, моделирующее работу гостиницы. Каждого клиента и гостиницу(точнее ее администратора) моделировать отдельным потоком.

Сценарий, описывающий одновременное поведение представленных в условии задания сущностей в терминах предметной области:

Представим конец 70-х. Горячее, пыльное лето в захудалом городке, где единственная гостиница гордо носит звание «Социалистическая» и держит все жилищные карты в своей руке. На улице — духота, туристы откуда-то издалека выползают из душных вагонов, облепленные пылью и жаждущие хоть немного комфорта. Но проблема в том, что номеров — всего N, и когда они заняты, все идут на одну и ту же лавку, притулившуюся возле входа.

Давайте заглянем внутрь. За стойкой регистрации стоит Администратор. Этот тип не похож на любезного портье из голливудских фильмов: он холоден, как сталь, но при этом справедлив. У него мрачное настроение и циничная ухмылка. Он знает, что творится в его царстве комнат — все N заняты какими-то проходимцами, затянувшимися на несколько дней. И в эту минуту к дверям подъезжает очередной клиент — зовем его Клиент #1 — выглядит как парень, который уже пережил поездку в плацкарте среди криков младенцев и запаха бутербродов с колбасой, влажных от жаркой духоты.

**Клиент #1 (потягиваясь):**  
«Хозяин, мне бы на денек-другой уголок, переночевать, душ принять, ну ты понял?»  
Администратор, не поднимая глаз от журнала заселения, медленно произносит:  
«Все занято. Присаживайся-ка на скамейку, парень.»

Клиент #1 оборачивается, видит уже целую ораву таких же измученных туристов, уставших путников. Все они сидят рядком, как в приёмной дешёвого врача, и ждут. Это выглядит как сцена из криминальной драмы: напряженные лица, в воздухе пахнет потом и лёгкой агрессией. Все эти клиенты — отдельные личности, со своими историями и мотивами. Клиент #2 — тип в шляпе, недоверчиво оглядывающийся по сторонам. Клиент #3 — усталая женщина, укутанная в цветастый платок, уже полу спит, уронив голову на плечо незнакомца. Клиент #4 пытается выцыганить у кого-то сигарету, хотя и зная, что у него не будет огня.

В это время в коридоре наверху — один из постояльцев, уже заселенный несколько дней назад, собирает вещи, кидает в чемодан мятую одежду. Ему пора валить. Он спускается по лестнице, бросает ключ администратору. Администратор, как прозорливый бармен, сразу вскидывает голову, и брови у него чуть поднимаются. Освободился номер. Он выходит к очереди у скамейки:

«Эй, первый по очереди! Ладно, заходи, есть для тебя уголок. Номер свободен!» — голос его звучит почти как выстрел, отдавая холодом. Первый в очереди клиент вскакивает, будто ему помахали перед носом стодолларовой купюрой, отряхивает пыль со штанов, молча кивает остальным в очереди (мол, без обид, ребята), и ныряет внутрь гостиницы.

Остальные клиенты на скамейке вздыхают, кто-то ворчит: «Скорей бы ещё один освободился, а то я здесь просто прохожу круги ада». Некоторые криво усмехаются, кое-кто вытягивает ноги — все они ждут, пока кто-нибудь еще внутри не покинет свою комнату. И так будет повторяться снова и снова: кто-то наверху заканчивает свой отдых, освобождает номер — внизу счастливчик из очереди вскакивает и идёт на заселение. Так, с холодным профессионализмом администратора и нервным терпением ожидающих, эта социалистическая гостиница крутит свою дьявольскую рулетку: не комнатный сервис, а настоящая система выживания, где всё подчинено простому правилу — у кого терпение и везение, тот получит ключ. А остальным остаётся коротать время на скамье ожидания, мечтая о мягкой подушке за дверью ресепшена.

В конце концов, к вечеру, может, большинство клиентов уже заселятся, а те, кто не дождется — они так и останутся сидеть, надеясь, что ночь принесет освобождение ещё одного номера. А администратор? Он просто будет продолжать свою игру, распределяя номера, будто раздавая последние патроны голодным стрелкам, и с циничной усмешкой смотреть, как эта странная очередь живёт своей жизнью.

Таким образом, в любой момент времени мы можем наблюдать три группы действующих лиц:

1. **Администратор гостиницы:**  
   Распоряжается свободными номерами. Всегда знает, сколько их, и кому предоставить следующий. При освобождении номера оповещает тех, кто сидит в очереди.
2. **Клиенты, которые заселились:**  
   Они находятся в номерах разное количество дней — кто-то ночь, кто-то несколько. После завершения срока пребывания они освобождают номер.
3. **Клиенты, ожидающие очереди на заселение:**  
   Ждут, когда освободится номер. Они сидят в порядке прибытия, не толкаются и не пытаются обойти друг друга: первый, кто приехал и не нашел места, первый и получит освободившийся номер, как только таковой появится.

С течением времени часть клиентов будет уезжать, освобождая номера, а другие — дожидаться своей очереди и заселяться. В конце концов, когда все клиенты будут обслужены, все номера будут свободны. Если новых гостей не будет, очередь на скамейках со временем тоже закончится.

Модель параллельных вычислений: **Клиенты и серверы**.

Из условия: «Еще один способ взаимодействия неравноправных потоков. Клиентский поток запрашивает сервер и ждет ответа.

Серверный поток ожидает запроса от клиента, затем действует в соответствии с поступившим запросом».

В контексте задачи клиентом является клиент, а сервером – гостиница (точнее администратор гостиницы).

Код программы на 4-5 баллов:

#include <iostream>  
#include <pthread.h>  
#include <queue>  
#include <random>  
#include <unistd.h>  
  
pthread\_mutex\_t mutex; // Мьютекс для защиты общих ресурсов  
pthread\_cond\_t room\_available; // Условная переменная для ожидания освобождения номеров  
  
int NUM\_ROOMS = 25; // Количество номеров в гостинице (ввод с консоли)  
int NUM\_CLIENTS = 50; // Количество клиентов (ввод с консоли)  
int available\_rooms; // Текущее количество доступных номеров  
std::queue<int> waiting\_queue; // Очередь клиентов, ожидающих номер  
  
int min\_stay = 1; // Минимальное количество дней пребывания  
int max\_stay = 5; // Максимальная длительность  
  
// Функция для потока клиента  
void\* client\_thread(void\* arg) {  
 int client\_id = \*(int\*)arg;  
  
 // Генератор случайных чисел.  
 std::random\_device rd;  
 std::mt19937 gen(rd());  
 std::uniform\_int\_distribution<> distr(min\_stay, max\_stay);  
  
 // Определяем случайную длительность пребывания  
 int stay\_duration = distr(gen);  
  
 // Клиент пытается заселиться  
 pthread\_mutex\_lock(&mutex);  
  
 std::cout << "Клиент " << client\_id << " приехал в гостиницу и хочет остаться на "  
 << stay\_duration << " дн.\n";  
  
 if (available\_rooms == 0) { // Проверка наличия свободных номеров. Если их нет, то клиент встает в очередь  
 std::cout << "Клиент " << client\_id << " не нашел свободных номеров и ждет на кресле.\n";  
 waiting\_queue.push(client\_id);  
 while (waiting\_queue.front() != client\_id || available\_rooms == 0) { // Клиент ждет своей очереди и хотя бы один свободный номер  
 pthread\_cond\_wait(&room\_available, &mutex);  
 }  
 waiting\_queue.pop(); // Клиент получил очередь на заселение, выходим из очереди  
 }  
  
 // Клиент, который может занять номер (изначально или после выхода из очереди)  
 available\_rooms--;  
 std::cout << "Клиент " << client\_id << " заселяется в номер. Свободных номеров осталось: "  
 << available\_rooms << ".\n";  
  
 pthread\_mutex\_unlock(&mutex);  
  
 // Клиент проводит в номере заданное количество "дней" (секунд)  
 sleep(stay\_duration);  
  
 // Клиент освобождает номер  
 pthread\_mutex\_lock(&mutex);  
 available\_rooms++;  
 std::cout << "Клиент " << client\_id << " выехал из номера. Свободных номеров стало: "  
 << available\_rooms << ".\n";  
  
 // Оповещаем всех ожидающих в очереди, что номер освободился  
 pthread\_cond\_broadcast(&room\_available);  
 pthread\_mutex\_unlock(&mutex);  
  
 return nullptr;  
}  
  
int main() {  
 // Ввод начальных данных с консоли  
 std::cout << "Введите количество номеров в гостинице: ";  
 std::cin >> NUM\_ROOMS;  
  
 std::cout << "Введите количество клиентов: ";  
 std::cin >> NUM\_CLIENTS;  
  
 std::cout << "Введите минимальную длительность пребывания (дней): ";  
 std::cin >> min\_stay;  
  
 std::cout << "Введите максимальную длительность пребывания (дней): ";  
 std::cin >> max\_stay;  
  
 if (min\_stay > max\_stay) {  
 std::cerr << "Ошибка: минимальная длительность должна быть больше максимальной!";  
 return 1;  
 }  
 else if (min\_stay < 1 || max\_stay < 1) {  
 std::cerr << "Ошибка: длительность пребывания должна быть положительным числом!";  
 return 1;  
 }  
 else if (NUM\_ROOMS < 1) {  
 std::cerr << "Ошибка: количество номеров в гостинице должно быть положительным числом!";  
 return 1;  
 }  
 else if (NUM\_CLIENTS < 1) {  
 std::cerr << "Ошибка: количество клиентов должно быть положительным числом!";  
 return 1;  
 }  
   
  
 available\_rooms = NUM\_ROOMS; // Количество свободных номеров  
  
 pthread\_mutex\_init(&mutex, nullptr);  
 pthread\_cond\_init(&room\_available, nullptr);  
  
 // Создаем потоки (клиентов)  
 auto clients = new pthread\_t[NUM\_CLIENTS];  
 int\* client\_ids = new int[NUM\_CLIENTS];  
  
 for (int i = 0; i < NUM\_CLIENTS; i++) {  
 client\_ids[i] = i + 1;  
 pthread\_create(&clients[i], nullptr, client\_thread, &client\_ids[i]);  
 }  
  
 // Ожидаем завершения всех потоков  
 for (int i = 0; i < NUM\_CLIENTS; i++) {  
 pthread\_join(clients[i], nullptr);  
 }  
  
 // Освобождение выделенных ресурсов  
 delete[] clients;  
 delete[] client\_ids;  
 pthread\_mutex\_destroy(&mutex);  
 pthread\_cond\_destroy(&room\_available);  
  
 std::cout << "Все клиенты обслужены. Программа завершается.\n";  
  
 return 0;  
}

Входные данные:

**NUM\_ROOMS** - количество номеров в гостинице;

**NUM\_CLIENTS** - количество клиентов;

**min\_stay** – минимальное количество дней (секунд в рамках работы программы) пребывания;

**max\_stay** – максимальное количество дней (секунд в рамках работы программы) пребывания.

Дни пребывания для каждого из клиентов генерируются случайно в диапазоне от min\_stay до max\_stay. Числа по логике задания обознаяают количество дней, в контексте программы дни – секунды (об этом также в комментариях в программе).

Входные данные изначально заданы в программе (для дальнейшего критерия на 6-7 баллов, который касается ввода с консоли. Например, в случае, когда пользователь запускает в командной строке программу без ввода параметров), однако подразумевается ввод пользователем в консоль во время работы программы. После ввода происходит валидация введенных данных.

Тест 1:

Входные данные:

25

50

2

1

Выходные данные:

Ошибка: минимальная длительность должна быть больше максимальной!

Тест 2:

Входные данные:

25

50

-2

-1

Выходные данные:

Ошибка: длительность пребывания должна быть положительным числом!

Тест 3:

Входные данные:

-30

30

1

5

Выходные данные:

Ошибка: количество номеров должно быть положительным числом!

Тест 4:

Входные данные:

25

-50

1

5

Выходные данные:

Ошибка: количество клиентов должно быть положительным числом!

Тест 5:

Входные данные:

25

50

1

5

Выходные данные:

Введите количество номеров в гостинице: 25

Введите количество клиентов: 50

Введите минимальную длительность пребывания (дней): 1

Введите максимальную длительность пребывания (дней): 5

Клиент 1 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 1 заселяется в номер. Свободных номеров осталось: 24.

Клиент 7 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 7 заселяется в номер. Свободных номеров осталось: 23.

Клиент 6 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 6 заселяется в номер. Свободных номеров осталось: 22.

Клиент 2 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 2 заселяется в номер. Свободных номеров осталось: 21.

Клиент 9 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 9 заселяется в номер. Свободных номеров осталось: 20.

Клиент 3 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 3 заселяется в номер. Свободных номеров осталось: 19.

Клиент 20 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 20 заселяется в номер. Свободных номеров осталось: 18.

Клиент 21 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 21 заселяется в номер. Свободных номеров осталось: 17.

Клиент 11 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 11 заселяется в номер. Свободных номеров осталось: 16.

Клиент 24 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 24 заселяется в номер. Свободных номеров осталось: 15.

Клиент 10 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 10 заселяется в номер. Свободных номеров осталось: 14.

Клиент 5 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 5 заселяется в номер. Свободных номеров осталось: 13.

Клиент 27 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 27 заселяется в номер. Свободных номеров осталось: 12.

Клиент 15 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 15 заселяется в номер. Свободных номеров осталось: 11.

Клиент 8 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 8 заселяется в номер. Свободных номеров осталось: 10.

Клиент 16 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 16 заселяется в номер. Свободных номеров осталось: 9.

Клиент 17 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 17 заселяется в номер. Свободных номеров осталось: 8.

Клиент 18 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 18 заселяется в номер. Свободных номеров осталось: 7.

Клиент 19 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 19 заселяется в номер. Свободных номеров осталось: 6.

Клиент 22 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 22 заселяется в номер. Свободных номеров осталось: 5.

Клиент 23 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 23 заселяется в номер. Свободных номеров осталось: 4.

Клиент 12 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 12 заселяется в номер. Свободных номеров осталось: 3.

Клиент 25 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 25 заселяется в номер. Свободных номеров осталось: 2.

Клиент 13 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 13 заселяется в номер. Свободных номеров осталось: 1.

Клиент 26 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 26 заселяется в номер. Свободных номеров осталось: 0.

Клиент 28 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 28 не нашел свободных номеров и ждет на кресле.

Клиент 4 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 4 не нашел свободных номеров и ждет на кресле.

Клиент 30 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 30 не нашел свободных номеров и ждет на кресле.

Клиент 14 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 14 не нашел свободных номеров и ждет на кресле.

Клиент 29 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 29 не нашел свободных номеров и ждет на кресле.

Клиент 31 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 31 не нашел свободных номеров и ждет на кресле.

Клиент 33 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 33 не нашел свободных номеров и ждет на кресле.

Клиент 32 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 32 не нашел свободных номеров и ждет на кресле.

Клиент 36 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 36 не нашел свободных номеров и ждет на кресле.

Клиент 37 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 37 не нашел свободных номеров и ждет на кресле.

Клиент 34 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 34 не нашел свободных номеров и ждет на кресле.

Клиент 35 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 35 не нашел свободных номеров и ждет на кресле.

Клиент 38 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 38 не нашел свободных номеров и ждет на кресле.

Клиент 41 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 41 не нашел свободных номеров и ждет на кресле.

Клиент 40 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 40 не нашел свободных номеров и ждет на кресле.

Клиент 39 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 39 не нашел свободных номеров и ждет на кресле.

Клиент 42 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 42 не нашел свободных номеров и ждет на кресле.

Клиент 43 приехал в гостиницу и хочет остаться на 4 дн.

Клиент 43 не нашел свободных номеров и ждет на кресле.

Клиент 44 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 44 не нашел свободных номеров и ждет на кресле.

Клиент 45 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 45 не нашел свободных номеров и ждет на кресле.

Клиент 46 приехал в гостиницу и хочет остаться на 1 дн.

Клиент 46 не нашел свободных номеров и ждет на кресле.

Клиент 47 приехал в гостиницу и хочет остаться на 5 дн.

Клиент 47 не нашел свободных номеров и ждет на кресле.

Клиент 48 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 48 не нашел свободных номеров и ждет на кресле.

Клиент 49 приехал в гостиницу и хочет остаться на 2 дн.

Клиент 49 не нашел свободных номеров и ждет на кресле.

Клиент 50 приехал в гостиницу и хочет остаться на 3 дн.

Клиент 50 не нашел свободных номеров и ждет на кресле.

Клиент 11 выехал из номера. Свободных номеров стало: 1.

Клиент 28 заселяется в номер. Свободных номеров осталось: 0.

Клиент 9 выехал из номера. Свободных номеров стало: 1.

Клиент 4 заселяется в номер. Свободных номеров осталось: 0.

Клиент 8 выехал из номера. Свободных номеров стало: 1.

Клиент 30 заселяется в номер. Свободных номеров осталось: 0.

Клиент 12 выехал из номера. Свободных номеров стало: 1.

Клиент 14 заселяется в номер. Свободных номеров осталось: 0.

Клиент 21 выехал из номера. Свободных номеров стало: 1.

Клиент 29 заселяется в номер. Свободных номеров осталось: 0.

Клиент 7 выехал из номера. Свободных номеров стало: 1.

Клиент 31 заселяется в номер. Свободных номеров осталось: 0.

Клиент 28 выехал из номера. Свободных номеров стало: 1.

Клиент 33 заселяется в номер. Свободных номеров осталось: 0.

Клиент 17 выехал из номера. Свободных номеров стало: 1.

Клиент 32 заселяется в номер. Свободных номеров осталось: 0.

Клиент 23 выехал из номера. Свободных номеров стало: 1.

Клиент 36 заселяется в номер. Свободных номеров осталось: 0.

Клиент 25 выехал из номера. Свободных номеров стало: 1.

Клиент 37 заселяется в номер. Свободных номеров осталось: 0.

Клиент 14 выехал из номера. Свободных номеров стало: 1.

Клиент 34 заселяется в номер. Свободных номеров осталось: 0.

Клиент 20 выехал из номера. Свободных номеров стало: 1.

Клиент 35 заселяется в номер. Свободных номеров осталось: 0.

Клиент 16 выехал из номера. Свободных номеров стало: 1.

Клиент 38 заселяется в номер. Свободных номеров осталось: 0.

Клиент 26 выехал из номера. Свободных номеров стало: 1.

Клиент 41 заселяется в номер. Свободных номеров осталось: 0.

Клиент 33 выехал из номера. Свободных номеров стало: 1.

Клиент 40 заселяется в номер. Свободных номеров осталось: 0.

Клиент 29 выехал из номера. Свободных номеров стало: 1.

Клиент 39 заселяется в номер. Свободных номеров осталось: 0.

Клиент 18 выехал из номера. Свободных номеров стало: 1.

Клиент 42 заселяется в номер. Свободных номеров осталось: 0.

Клиент 19 выехал из номера. Свободных номеров стало: 1.

Клиент 43 заселяется в номер. Свободных номеров осталось: 0.

Клиент 10 выехал из номера. Свободных номеров стало: 1.

Клиент 44 заселяется в номер. Свободных номеров осталось: 0.

Клиент 30 выехал из номера. Свободных номеров стало: 1.

Клиент 45 заселяется в номер. Свободных номеров осталось: 0.

Клиент 13 выехал из номера. Свободных номеров стало: 1.

Клиент 46 заселяется в номер. Свободных номеров осталось: 0.

Клиент 37 выехал из номера. Свободных номеров стало: 1.

Клиент 47 заселяется в номер. Свободных номеров осталось: 0.

Клиент 38 выехал из номера. Свободных номеров стало: 1.

Клиент 48 заселяется в номер. Свободных номеров осталось: 0.

Клиент 24 выехал из номера. Свободных номеров стало: 1.

Клиент 49 заселяется в номер. Свободных номеров осталось: 0.

Клиент 2 выехал из номера. Свободных номеров стало: 1.

Клиент 39 выехал из номера. Свободных номеров стало: 2.

Клиент 50 заселяется в номер. Свободных номеров осталось: 1.

Клиент 5 выехал из номера. Свободных номеров стало: 2.

Клиент 1 выехал из номера. Свободных номеров стало: 3.

Клиент 3 выехал из номера. Свободных номеров стало: 4.

Клиент 6 выехал из номера. Свободных номеров стало: 5.

Клиент 27 выехал из номера. Свободных номеров стало: 6.

Клиент 22 выехал из номера. Свободных номеров стало: 7.

Клиент 15 выехал из номера. Свободных номеров стало: 8.

Клиент 4 выехал из номера. Свободных номеров стало: 9.

Клиент 46 выехал из номера. Свободных номеров стало: 10.

Клиент 32 выехал из номера. Свободных номеров стало: 11.

Клиент 31 выехал из номера. Свободных номеров стало: 12.

Клиент 41 выехал из номера. Свободных номеров стало: 13.

Клиент 40 выехал из номера. Свободных номеров стало: 14.

Клиент 48 выехал из номера. Свободных номеров стало: 15.

Клиент 49 выехал из номера. Свободных номеров стало: 16.

Клиент 44 выехал из номера. Свободных номеров стало: 17.

Клиент 36 выехал из номера. Свободных номеров стало: 18.

Клиент 45 выехал из номера. Свободных номеров стало: 19.

Клиент 34 выехал из номера. Свободных номеров стало: 20.

Клиент 35 выехал из номера. Свободных номеров стало: 21.

Клиент 43 выехал из номера. Свободных номеров стало: 22.

Клиент 50 выехал из номера. Свободных номеров стало: 23.

Клиент 42 выехал из номера. Свободных номеров стало: 24.

Клиент 47 выехал из номера. Свободных номеров стало: 25.

Все клиенты обслужены. Программа завершается.

Скриншоты с тестами:

Тест 1:

Изображение выглядит как программное обеспечение, текст, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Тест 2:

Изображение выглядит как программное обеспечение, Мультимедийное программное обеспечение, текст, Графическое программное обеспечение

Автоматически созданное описание

Тест 3:

Изображение выглядит как программное обеспечение, Мультимедийное программное обеспечение, текст, Графическое программное обеспечение

Автоматически созданное описание

Тест 4:

Изображение выглядит как программное обеспечение, Мультимедийное программное обеспечение, текст, Графическое программное обеспечение

Автоматически созданное описание

Тест 5 (начало и конец вывода):

Изображение выглядит как программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение, текст

Автоматически созданное описание

Изображение выглядит как Мультимедийное программное обеспечение, программное обеспечение, Графическое программное обеспечение, Редактирование

Автоматически созданное описание

6-7 баллов:

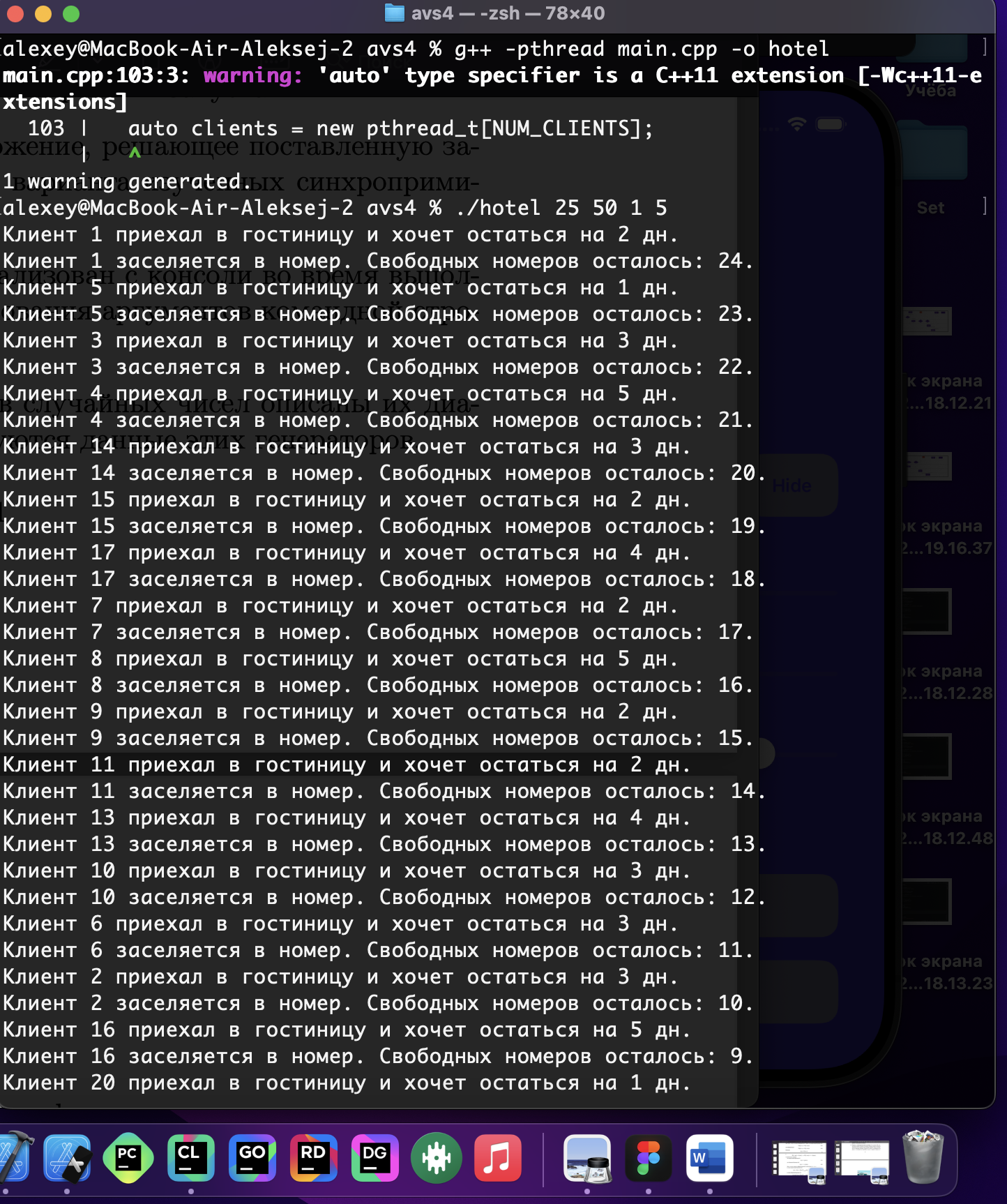
Обновленный код программы на 6-7 баллов:

#include <iostream>  
#include <pthread.h>  
#include <queue>  
#include <random>  
#include <unistd.h>  
#include <cstdlib>   
  
pthread\_mutex\_t mutex; // Мьютекс для защиты общих ресурсов  
pthread\_cond\_t room\_available; // Условная переменная для ожидания освобождения номеров  
  
int NUM\_ROOMS = 25; // Количество номеров в гостинице (ввод с консоли)  
int NUM\_CLIENTS = 50; // Количество клиентов (ввод с консоли)  
int available\_rooms; // Текущее количество доступных номеров  
std::queue<int> waiting\_queue; // Очередь клиентов, ожидающих номер  
  
int min\_stay = 1; // Минимальное количество дней пребывания  
int max\_stay = 5; // Максимальная длительность  
  
// Функция для потока клиента  
void\* client\_thread(void\* arg) {  
 int client\_id = \*(int\*)arg;  
  
 // Генератор случайных чисел.  
 std::random\_device rd;  
 std::mt19937 gen(rd());  
 std::uniform\_int\_distribution<> distr(min\_stay, max\_stay);  
  
 // Определяем случайную длительность пребывания  
 int stay\_duration = distr(gen);  
  
 // Клиент пытается заселиться  
 pthread\_mutex\_lock(&mutex);  
  
 std::cout << "Клиент " << client\_id << " приехал в гостиницу и хочет остаться на "  
 << stay\_duration << " дн.\n";  
  
 if (available\_rooms == 0) { // Проверка наличия свободных номеров. Если их нет, то клиент встает в очередь  
 std::cout << "Клиент " << client\_id << " не нашел свободных номеров и ждет на кресле.\n";  
 waiting\_queue.push(client\_id);  
 while (waiting\_queue.front() != client\_id || available\_rooms == 0) { // Клиент ждет своей очереди и хотя бы один свободный номер  
 pthread\_cond\_wait(&room\_available, &mutex);  
 }  
 waiting\_queue.pop(); // Клиент получил очередь на заселение, выходим из очереди  
 }  
  
 // Клиент, который может занять номер (изначально или после выхода из очереди)  
 available\_rooms--;  
 std::cout << "Клиент " << client\_id << " заселяется в номер. Свободных номеров осталось: "  
 << available\_rooms << ".\n";  
  
 pthread\_mutex\_unlock(&mutex);  
  
 // Клиент проводит в номере заданное количество "дней" (секунд)  
 sleep(stay\_duration);  
  
 // Клиент освобождает номер  
 pthread\_mutex\_lock(&mutex);  
 available\_rooms++;  
 std::cout << "Клиент " << client\_id << " выехал из номера. Свободных номеров стало: "  
 << available\_rooms << ".\n";  
  
 // Оповещаем всех ожидающих в очереди, что номер освободился  
 pthread\_cond\_broadcast(&room\_available);  
 pthread\_mutex\_unlock(&mutex);  
  
 return nullptr;  
}  
  
int main(int argc, char\* argv[]) {  
 // Ввод начальных данных в командной строке  
 // Создание исполняемого файла с помощью команды: g++ -pthread main.cpp hotel  
 // (hotel - имя параметр, можно задать другое, но его же использовать в следующих командах))  
 // Формат: ./hotel\_simulator [NUM\_ROOMS] [NUM\_CLIENTS] [MIN\_STAY] [MAX\_STAY]  
 if (argc > 1) NUM\_ROOMS = std::atoi(argv[1]);  
 if (argc > 2) NUM\_CLIENTS = std::atoi(argv[2]);  
 if (argc > 3) min\_stay = std::atoi(argv[3]);  
 if (argc > 4) max\_stay = std::atoi(argv[4]);  
  
 if (min\_stay > max\_stay) {  
 std::cerr << "Ошибка: минимальная длительность должна быть больше максимальной!";  
 return 1;  
 }  
 else if (min\_stay < 1 || max\_stay < 1) {  
 std::cerr << "Ошибка: длительность пребывания должна быть положительным числом!";  
 return 1;  
 }  
 else if (NUM\_ROOMS < 1) {  
 std::cerr << "Ошибка: количество номеров в гостинице должно быть положительным числом!";  
 return 1;  
 }  
 else if (NUM\_CLIENTS < 1) {  
 std::cerr << "Ошибка: количество клиентов должно быть положительным числом!";  
 return 1;  
 }  
  
  
  
 available\_rooms = NUM\_ROOMS; // Количество свободных номеров  
  
 pthread\_mutex\_init(&mutex, nullptr);  
 pthread\_cond\_init(&room\_available, nullptr);  
  
 // Создаем потоки (клиентов)  
 auto clients = new pthread\_t[NUM\_CLIENTS];  
 int\* client\_ids = new int[NUM\_CLIENTS];  
  
 for (int i = 0; i < NUM\_CLIENTS; i++) {  
 client\_ids[i] = i + 1;  
 pthread\_create(&clients[i], nullptr, client\_thread, &client\_ids[i]);  
 }  
  
 // Ожидаем завершения всех потоков  
 for (int i = 0; i < NUM\_CLIENTS; i++) {  
 pthread\_join(clients[i], nullptr);  
 }  
  
 // Освобождение выделенных ресурсов  
 delete[] clients;  
 delete[] client\_ids;  
 pthread\_mutex\_destroy(&mutex);  
 pthread\_cond\_destroy(&room\_available);  
  
 std::cout << "Все клиенты обслужены. Программа завершается.\n";  
  
 return 0;  
}

Добавлено чтение с консоли до запуска программы. Если параметры не вводятся, используются параметры, заданные программой.

Скриншоты с примерами на тесте 5 и тесте без параметров:

Тест 5 (начало вывода):

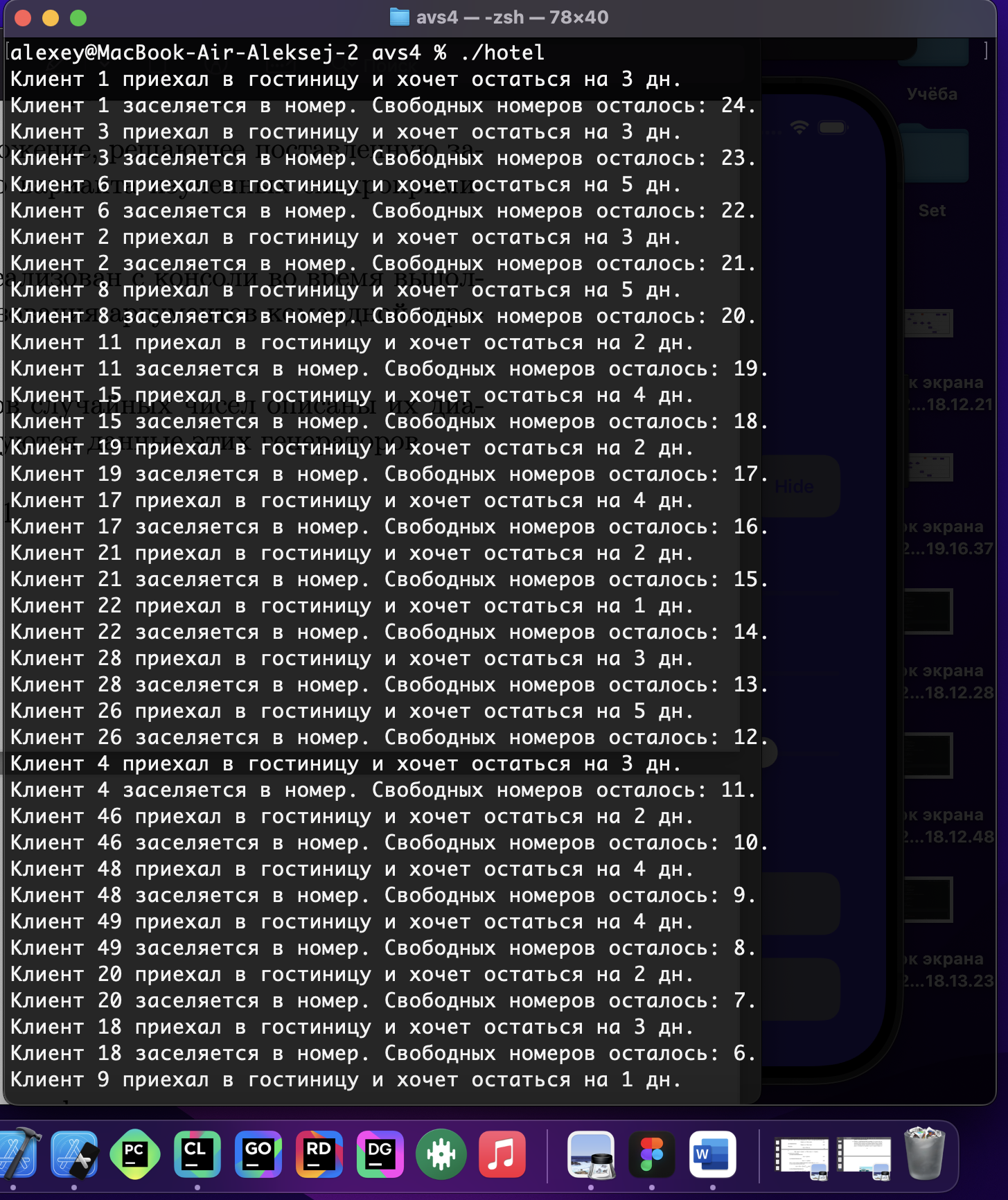


Тест 5 (конец вывода):

Изображение выглядит как электроника, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Тест без параметров:



Обобщенный алгоритм, используемый при реализации программы исходного словесного сценария:

* **Администратор (субъект):** в программе формально не создается отдельный поток администратора, но его роль выполняется совокупностью общих ресурсов и примитивов синхронизации. Можно считать, что «администратор» — это логика, которая реализуется при каждом попытке клиента занять или освободить номер. Это набор функций, работающих с мьютексом, очередью и условной переменной.
* **Клиент (субъект):** клиент отображается на поток (pthread\_t), который при создании получает идентификатор и случайную длительность проживания. Клиент действует по алгоритму:

попытка занять номер → ожидание при отсутствии свободных номеров → заселение на определенный срок → освобождение номера и уведомление очереди.

* **Номера (объект):** количество свободных номеров (available\_rooms) — это общая переменная, отражающая состояние ресурсов. Доступ к ней регулируется мьютексом.
* **Очередь ожидания (объект):** очередь клиентов waiting\_queue отображает «скамейки» перед гостиницей. Доступ имеют все потоки через мьютекс.

### **Пошаговый алгоритм работы программы**

1. **Прибытие клиента:**

Поток клиента создается через pthread\_create и получает свой **идентификатор** (client\_id) и **случайную длительность проживания в заданном диапазоне**.

Клиент обращается к общему ресурсу (available\_rooms) через мьютекс.

1. **Проверка наличия свободных номеров:**

Если есть свободный номер (available\_rooms > 0), клиент сразу занимает его:

* Уменьшает счетчик свободных номеров (available\_rooms--).
* Выводит сообщение о заселении.

Если номеров нет (available\_rooms == 0), клиент:

* Добавляется в очередь ожидания (waiting\_queue.push).
* Ждет освобождения номера с помощью pthread\_cond\_wait.

1. **Ожидание в очереди:**

Клиент остается в ожидании, пока он не окажется первым в очереди (waiting\_queue.front()), **и** номер не станет доступным (available\_rooms > 0).

Как только эти условия выполняются, клиент удаляется из очереди (waiting\_queue.pop), затем заселяется в номер.

1. **Освобождение номера:**

Клиент «спит» (симулирует время пребывания) с помощью sleep(stay\_duration).

По окончании времени клиент:

* Увеличивает счетчик свободных номеров (available\_rooms++).
* Выводит сообщение об освобождении номера.
* Уведомляет всех ожидающих клиентов с помощью pthread\_cond\_broadcast, чтобы те могли проверить доступность номеров.

1. **Цикл ожидания и разбуженные клиенты:**

Когда освобождается номер, условная переменная **разблокирует** всех ожидающих клиентов.

Клиенты по очереди (согласно порядку в waiting\_queue) проверяют, кто сейчас первый, и заселяются в номер.

1. **Завершение работы программы:**

Главный поток (main) вызывает pthread\_join для каждого клиента, дожидаясь завершения их работы.

Завершение выполнения.