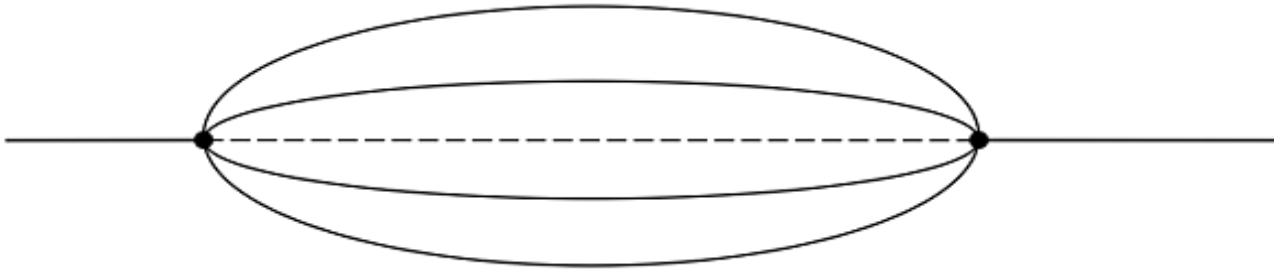
Зарипов Антон Русланович, БПИ-203

Вариант 12

Задача о гостинице-2 (умные клиенты). В гостинице 10 номеров с ценой 200 рублей, 10 номеров с ценой 400 рублей и 5 номеров с ценой 600 руб. Клиент, зашедший в гостиницу, обладает некоторой суммой и получает номер по своим финансовым возможностям, если тот свободен. Если среди доступных клиенту номеров нет свободных, клиент уходит искать ночлег в другое место. Создать многопоточное приложение, моделирующее работу гостиницы.

Описание модели

В моем варианте не указано, какую модель синхронизации потоков использовать, поэтому для написания программы я использовал синхропримитив «управляющий и рабочие», как наиболее подходящий для этой задачи по моему мнению. В модели делегирования выделяется один центральный *поток* (управляющий) и несколько рабочих потоков. Управляющий *поток* запускает рабочие потоки, передает им все необходимые данные, контролирует работу и обрабатывает результаты после их завершения.



В моем отеле главный поток - «консьерж», контролирующий приходящих к нему клиентов и номера для них. Консьерж пускает клиентов, пока есть свободные номера. Если клиент не может себе позволить ни один из доступных номеров, то он уходит. Если же он может позволить самый дорогой, то он занимает его, если же все дорогие номера заняты, то он ищет номер попроще, и тп. Если он не нашел себе номер, то он уходит. Когда все номера заняты, отель закрывается и не пускает клиентов (зачем, если он не сможет их пристроить?). Из описания очевидно, что подконтрольные потоки – это клиенты.

Источники инф-ии о данной модели:
1)Лекции по курсу АВС
2)https://docplayer.com/48706922-Lekciya-5-paradigmy-parallelnogo-programmirovaniya.html
3)https://litresp.ru/chitat/ru/X/hjyuz-kameron/paralleljnoe-i-raspredelennoe-programmirovanie-na-s/6
4)https://rsdn.org/article/baseserv/RUThreadingMethodology.xml

Компиляция и запуск:

Запуск программы я осуществлял с помощью CMake следующими командами из папки проекта:
cmake .
cmake –build ./
./threads <inFile> <outFile>
Файлы с тестами и результатами их прогона также содержатся в архиве.

Вывод:
Программа выводит итоговое состояние отеля: сколько каких номеров занято. Пример вывода:

```
 total:
   200$ rooms: 10
   400$ rooms: 3
   600$ rooms: 5
```