**📚 Библиотека и системы хранения**

1. Поиск книги на полке

В библиотеке каждая книга имеет уникальный ISBN. Реализуйте систему, которая по ISBN мгновенно определяет, на какой полке находится книга.

2. Стек возвращённых книг

Читатели возвращают книги в специальный ящик. Реализуйте систему, которая обрабатывает книги в порядке "последняя возвращена — первая взята на проверку".

3. Очередь на выдачу книг

В читальном зале книги выдаются по запросу. Реализуйте систему очереди, чтобы книги выдавались в порядке поступления запросов.

4. Дерево каталогов

Книги в библиотеке классифицированы по разделам (наука, искусство и т. д.). Постройте структуру, позволяющую быстро находить все книги в подразделе "Фантастика → Космическая опера".

**🏥 Больница и медицинские системы**

5. Регистратура

Пациенты записываются к врачу. Реализуйте систему, которая вызывает их в порядке живой очереди.

6. Срочные пациенты

В приёмном покое пациенты с тяжёлыми состояниями должны обслуживаться вне очереди, но обычные пациенты ждут своей очереди. Реализуйте систему с возможностью добавления срочных случаев в начало очереди.

7. Хранение истории болезней

У каждого пациента есть уникальный ID. Реализуйте систему, которая по ID мгновенно находит его медицинскую карту.

8. Аптека

Лекарства поступают партиями с определённым сроком годности. Реализуйте систему, которая выдаёт лекарства по принципу "первым истёк срок — первым выдано".

**🛒 Супермаркет и логистика**

9. Кассы

Покупатели становятся в очередь к кассам. Реализуйте систему, которая обрабатывает их в порядке FIFO (первым пришёл — первым обслужен).

10. Склад

На склад поступают товары, и их нужно отгружать как "первым пришёл — первым ушёл" (FIFO), но иногда требуется срочно отгрузить последнюю партию (LIFO). Реализуйте гибридную систему.

11. Поиск товара по штрих-коду

Каждый товар имеет уникальный штрих-код. Реализуйте систему, которая мгновенно определяет, в каком отделе он находится.

12. Маршруты доставки

Курьеры развозят заказы по городу. Постройте граф улиц и найдите кратчайший путь для доставки.

**🎮 Игры и развлечения**

13. История ходов в шахматах

Игроки могут отменять ходы (Ctrl+Z). Реализуйте систему, которая хранит историю ходов и позволяет откатываться назад.

14. Мультиплеерная очередь

Игроки ждут своей очереди для входа на сервер. Реализуйте систему, которая пускает их в порядке подключения.

15. Генерация лабиринта

Случайным образом создайте лабиринт, в котором есть хотя бы один путь от входа к выходу.

**📡 Сети и телекоммуникации**

16. Маршрутизация пакетов

Пакеты данных поступают на сервер. Реализуйте систему, которая обрабатывает их в порядке поступления.

17. Кэширование DNS-запросов

Часто запрашиваемые домены должны находиться быстрее. Реализуйте кэш, хранящий IP-адреса по доменным именам.

18. Топология сети

Определите, какие серверы соединены между собой, и найдите кратчайший путь для передачи данных.

19. Стек вызовов в программе

При вызове функции её контекст сохраняется. Реализуйте механизм, позволяющий возвращаться к предыдущему состоянию.

**🏦 Банки и финансы**

20. Транзакции

Клиент может отменить последнюю операцию. Реализуйте систему отката транзакций.

21. Очередь в банкомате

Люди ждут своей очереди, чтобы снять деньги. Реализуйте систему FIFO.

22. Быстрый поиск клиента

По номеру счёта мгновенно находите данные клиента.

**🚗 Такси и транспорт**

23. Вызов такси

Заказы обрабатываются в порядке поступления. Реализуйте диспетчерскую систему.

24. Поиск ближайшего водителя

Найдите кратчайший путь от водителя до клиента на карте.

25. Тарифы

В зависимости от типа поездки (эконом, комфорт, бизнес) мгновенно получайте стоимость.

26. История поездок

Пассажир может отменить последний заказ. Реализуйте систему отмены.

**🎵 Музыка и мультимедиа**

27. Плейлист

Песни играют в порядке добавления. Реализуйте систему воспроизведения.

28. Поиск песни по названию

Пользователь вводит название — система мгновенно находит трек.