**Тестовое задание на позицию разработчика в команду разработки логистических алгоритмов**

**Модель**

Рассмотрим работу *распределительного центра (distribution center, DC)* некоторой торговой организации. Распределительный центр – это центральный склад, откуда товары доставляются клиентам.

Распределительный центр имеет четко определенные часы работы (например, с 8:00 до 22:00). Организация имеет собственный ограниченный по размеру *флот* *(fleet)* одинаковых грузовых автомобилей (ресурсов – *resource)*, имеющих ограниченную вместимость (*capacity -* выражается скалярной величиной).

В течение дня клиенты, расположенные в разных точках города, делают *заказы (order)* через оператора колл-центра. При подтверждении заказа оператор договаривается с клиентом об удобном интервале доставки. Этот интервал называется *временным окном доступности (time window)* заказа.

Каждый день вечером диспетчер организации загружает в специализированное приложение все заказы, которые необходимо доставить клиентам на следующий день.

Заказы (один или несколько) должны быть погружены на автомобиль в пределах периода работы распределительного центра, затем автомобиль стартует из распределительного центра и доставляет заказы клиентам. Суммарный вес заказов, перевозимых в одном автомобиле, не должен при этом превышать вместимость автомобиля. Когда все заказы доставлены, автомобиль должен вернуться в распределительный центр до завершения его часов работы.

Приложение, используемое диспетчером, принимает всю информацию о заказах и ресурсах и для каждого ресурса строит расписание, содержащее:

1. Список заказов, которые будет перевозить автомобиль
2. Время начала погрузки заказов в распределительном центре
3. Время выезда автомобиля из распределительного центра
4. Для каждого заказа:
   1. Время прибытия на точку обслуживания клиента
   2. Время отъезда с точки обслуживания клиента (после отгрузки товара)
5. Время возвращения автомобиля в распределительный центр

**Характеристики концептов модели**

*Распределительный центр:*

- Координаты:

- Широта

- Долгота

- Окно доступности (время начала работы – время завершения работы)

- Число ресурсов

*Ресурс:*

- Скорость передвижения (км/ч)

- Вместимость (кг)

*Заказ:*

- Координаты:

- Широта

- Долгота

- Вес (кг)

- Временное окно доступности

- Время, необходимое на погрузку заказа в распределительном центре (мин)

- Время, необходимое на разгрузку заказа у клиента (мин)

**Метрика растояний**

Для вычисления расстояний между конкретными координатами, можно использовать любую вариацию метрики Crowflight (также встречаются варианты *Crowfly distance*, *As the crow flies*).

В качестве старта для поиска информации по метрикам можно использовать Википедию (<https://en.wikipedia.org/wiki/As_the_crow_flies>). Уделять большое внимание реализации конкретной метрики не нужно (допускается копирование любой реализации из интернета с указанием источника).

Плюсом будет реализация в дополнение к crowflight интеграции с любым онлайн сервисом, предоставляющим маршрутизацию (например, OSRM: <http://project-osrm.org/docs/v5.5.1/api/#route-service>).

**Целевая функция**

Оптимальным считается расписание, суммарная *длительность работы* *ресурсов* в котором, минимальна. Длительность работы ресурса – это время, прошедшее с момента погрузки в распределительном центре в начале рейса до момента возвращения в распределительный центр в конце рейса. Времена погрузки и обслуживания заказов известны, времена переездов между координатами можно вычислить, используя скорость передвижения, заданную для ресурса.

**Задание**

Необходимо разработать прототипы модулей приложения, решающих следующие задачи:

***1. Симуляция***

С каждым заказом в системе помимо веса ассоциировано его окно доступности. При этом если, например, депо начинает работать в 8:00, а окно доступности первого заказа в рейсе начинается в 16:00, нет смысла начинать погрузку в 8:00, и потом ждать, пока клиент приедет домой. В этом случае, погрузку можно начать позже, таким образом, чтобы приехать точно к началу окна доступности клиента.

Допустим, что система уже распределила заказы по машинам и на каждой из них определила порядок их выполнения.

*Необходимо реализовать алгоритм, который выбирает оптимальное время начала погрузки с учетом временных окон доступности заказов и часов работы распределительного центра, а также вычисляет времена начала и завершения обслуживания каждого заказа в рейсе и время возвращения машины в распределительный центр.*

*Время считается с точностью до 1 минуты.*

*Варианты реализации:*

*1.**Методы линейного программирования (LP)*

*2. Перебор с оптимизацией*

*3. Собственный эвристический алгоритм*

*4. Что-то еще (принимается любой вариант, решающий задачу)*

***2. Задача коммивояжера (TSP – Travelling Salesman Problem)***

Допустим, диспетчер вручную распределяет заказы по машинам. Затем, чтобы получить оптимизированное расписание, он нажимает кнопку «оптимизировать порядок заказов», после чего система изменяет последовательность выполнения заказов на маршруте таким образом, чтобы суммарное время работы ресурса было минимальным с учетом работы модуля, выполняющего симуляцию из задачи 1.

*Необходимо разработать алгоритм, который для одного маршрута выполняет эту оптимизацию порядка (решает задачу коммивояжера). Т.к. задача не имеет решения за полиномиальное время, предлагается реализовать один из эвристических алгоритмов (например, 2-opt, муравьиный алгоритм, вариацию LKH или что-то еще).*

*Найти глобально оптимальное решение в рамках данной задачи не требуется.*

Тестовые данные и информацию о возможном решении задачи можно найти в Интернете по запросу “Travelling Salesman Problem with Time Windows “ или “TSPTW” (например, <http://myweb.uiowa.edu/bthoa/TSPTWBenchmarkDataSets.htm>)

***3. Транспортная задача с ограничениями по вместимости и временными окнами (CVRPTW)***

При большом количестве заказов диспетчеру трудно оптимально распределить заказы между машинами за приемлемое время. В этом случае он нажимает кнопку «построить расписание автоматически». При этом система распределяет заказы по машинам, затем для каждой машины решает задачу коммивояжера из задания 2.

При распределении система должна учитывать следующие факторы:

1. Вместимость машины ограничена
2. Число машин ограничено
3. Машина должна вернуться в распределительный центр в период его работы
4. Машина должна успевать обслуживать заказы в их окна доступности
5. Возможна ситуация, в которой не все заказы могут быть распределены без нарушения факторов 1 – 4. В этом случае система должна оставить часть заказов незапланированными.

*Необходимо разработать алгоритм, выполняющий автоматическое распределение заказов по автомобилям. Т.к. задача не имеет решения за полиномиальное время, предлагается реализовать один из известных эвристических алгоритмов (Clarke & Wright Savings, Sweep, Petal, Best Insertion, что-то еще).*

*Приветствуется также реализация дополнительного алгоритма, который, получив на вход решение, с помощью любой вариации локального поиска (Local Search, LS) будет его улучшать.*

*Найти глобально оптимальное решение в рамках данной задачи не требуется.*

Тестовые данные и информацию о возможном решении задачи можно найти в Интернете по запросу “Capacitated Vehicle Routing Problem with Time Windows” или “CVRPTW”.

**Комментарии по выполнению заданий**

Не обязательно самостоятельно придумывать решение всех задач (можно найти готовое решение и разобраться в нем). При использовании готовых решений, необходимо указывать источник информации.

При реализации решения язык программирования не ограничивается, но предпочтительной будет реализация на Java / C++ / C#.

Обязательной к выполнению является только задача 1. Решение остальных задач необязательно для успешного выполнения тестового задания (их можно делать, если остается время и хочется чуть больше хардкора).

Приветствуется максимально эффективная реализация алгоритмов (т.е. лучше сделать 1 задачу, но так, чтобы она решалась быстро, чем 2 задачи, но с тормозами).