**1ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ   
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Преподаватель департамента программной инженерии факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.Н. Береснева  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г. | УТВЕРЖДАЮ  Академический руководитель  образовательной программы  «Программная инженерия», канд. техн.  наук, профессор ДПИ ФКН  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г. |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **ПРОГРАММА РЕШЕНИЯ ЗАДАЧИ МАРШРУТИЗАЦИИ С ОГРАНИЧЕНИЕМ ПО ГРУЗОПОДЪЕМНОСТИ НА ОСНОВЕ МЕТОДА ИМИТАЦИИ ОТЖИГА**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ RU.17701729.503200-01 81 01-1-ЛУ**  Исполнитель  Студент группы БПИ163  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Д.В. Строков /  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2018 г. | |  |

**УТВЕРЖДЕН**

**RU.17701729.503200-01 81 01-1 ЛУ**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **ПРОГРАММА РЕШЕНИЯ ЗАДАЧИ МАРШРУТИЗАЦИИ С ОГРАНИЧЕНИЕМ ПО ГРУЗОПОДЪЕМНОСТИ НА ОСНОВЕ МЕТОДА ИМИТАЦИИ ОТЖИГА**  **Пояснительная записка  RU.17701729.503200-01 81 01-1  Листов 38** |  |

**СОДЕРЖАНИЕ**

[1. ВВЕДЕНИЕ 3](#_Toc483276326)

[1.1. Наименование программы 3](#_Toc483276327)

[1.2. Документы, на основании которых ведётся разработка 3](#_Toc483276328)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 4](#_Toc483276329)

[2.1. Функциональное назначение 4](#_Toc483276330)

[2.2. Эксплуатационное назначение 4](#_Toc483276331)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc483276332)

[3.1. Постановка задачи на разработку программы 5](#_Toc483276333)

[3.2. Описание алгоритмов и функционирования программы 5](#_Toc483276334)

[3.2.1 Переходы между сценами 5](#_Toc483276335)

[3.2.2 Генерация игрового поля 5](#_Toc483276336)

[3.2.3 Генерация графа исходов для искусственного интеллекта 6](#_Toc483276337)

[3.2.4 Ход искусственного интеллекта 7](#_Toc483276338)

[3.3 Описание и обоснование выбора метода организации входных и выходных данных 7](#_Toc483276339)

[3.4 Описание и обоснование выбора состава технических и программных средств 7](#_Toc483276340)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 8](#_Toc483276341)

[4.1. Ориентировочная экономическая эффективность 8](#_Toc483276342)

[4.2. Предполагаемая потребность 8](#_Toc483276343)

[4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 8](#_Toc483276344)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 9](#_Toc483276345)

[Приложение 1 10](#_Toc483276346)

[Приложение 2 12](#_Toc483276347)

[Приложение 3 14](#_Toc483276348)

# 1. ВВЕДЕНИЕ

## 1.1. Наименование программы

Наименование программы: «Программа решения задачи маршрутизации с ограничением по грузоподъемности на основе метода имитации отжига» («Program for Solving the Capacitated Vehicle Routing Problem Based on Simulated Annealing Algorithm»).

## 1.2. Документы, на основании которых ведётся разработка

Приказ НИУ ВШЭ об утверждении тем курсовых работ № 2.3-02/1502-01 от 15.02.2017, утверждённый руководителем департамента программной инженерии факультета компьютерных наук.

# 2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## 2.1. Функциональное назначение

Функциональным назначением программы является получение входных данных для задачи маршрутизации с ограничением по грузоподъёмности и решение данной задачи с использованием метода имитации отжига, а также наглядная визуализация работы данного алгоритма и вывод результатов его работы.

## 2.2. Эксплуатационное назначение

Эксплуатационным назначением программы является её использование для решения задач маршрутизации с ограничением по грузоподъёмности, и получения наглядного отображения работы алгоритма имитации отжига. Также программа может пригодиться для практических целей в логистике – для подбора оптимального маршрута доставки грузов.

# 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## 3.1. Постановка задачи на разработку программы

Программа должна посредством алгоритма имитации отжига на заданном наборе вершин с двумерными координатами и заданным в виде числа грузом для каждой вершины, а также числе максимальной нагрузки на транспорт и номеру вершины, являющейся депо решать и визуализировать процесс решения задачи маршрутизации с ограничением грузоподъёмности – строить набор маршрутов, минимизируя их суммарную длину, так чтобы каждая вершина (кроме вершины - депо) была посещена ровно один раз, суммарная нагрузка на каждом маршруте не превышала заданную максимальную нагрузку, а каждый маршрут начинался и заканчивался в вершине - депо. Также должна быть возможно сохранять и загружать готовые решения.

## 3.2. Описание алгоритмов и функционирования программы

### 3.2.1 Общая структура программы

Модуль исполнения служит точкой запуска программы. После запуска интерфейса модуль исполнения принимает команды интерфейса на запуск, паузу или остановку алгоритма имитации отжига, а также на сохранение и загрузку файлов и отправляет текущее состояние выполнения задачи. Модуль интерфейса получает от пользователя вышеперечисленные команды и отправляет их в модуль исполнения, а также визуализирует текущее состояние.

### 3.2.2 Алгоритм имитации отжига

Изначально алгоритм генерирует случайное состояние – набор маршрутов который не нарушает наложенные ограничения, а также устанавливает заданную начальную температуру. Затем происходит последовательное исполнение итераций.

На каждой итерации генерируется “соседнее” состояние следующим образом: случайным образом выбираются от 1 до 8 вершин и перемещаются в случайное место таким образом, чтобы не нарушались наложенные ограничения. Пусть delta = разность суммарных расстояний маршрутов в новом и текущем состояниях. Тогда вероятность перехода к новому состоянию равняется exp(-delta / t), где t – текущая температура. Если текущее состояние выгоднее лучшего, то лучшее состояние становится равным новому.

Как только температура становится меньшей, чем заданная минимальная, либо превышен лимит по указанному времени, алгоритм прекращает работу.

### 3.2.3 Сохранение графика результатов

На каждой итерации минимизируемая оценка выгодности состояния равная суммарному расстоянию всех маршрутов состояния сохраняется в специальный контейнер (см. класс ChartData). Т.к. итераций бывает много, хранить оценки всех состояний не оптимально и может вызвать переполнение памяти. Поэтому при достижении лимита количества точек (по умолчанию 1000) график сжимается, последовательно извлекая из каждой тройки соседних точек минимальную и максимальную точку.

### 3.2.4 Визуализация выполнения алгоритма

Модуль интерфейса, получая текущее состояние выполнения отображает график, список загрузок и пройденный путь транспорта, а также граф и анимированный список маршрутов посредством HTML/CSS и JavaScript.

## 3.3 Описание и обоснование выбора метода организации входных и выходных данных

Входной файл должен соответствовать следующему формату:

CAPACITY C

DIMENSION N

NODE\_COORD\_SECTION

N1 X1 Y1

N2 X2 Y2

…

Nn Xn Yn

DEMAND\_SECTION

N1 D1

N2 D2

…

Nn Dn

EOF

Где C – ограничение по грузоподъёмности на единицу транспорта, N – количество вершин графа, Ni – номер вершины, Xi – x-координата вершины, y-координата вершины, Di – величина груза вершины.

Выходной файл должен соответствовать следующему формату:

Route #1: v1 v2 … vk

Route #2: v1 v2 … vk

…

Route #M: v1 v2 … vk

cost T

Где M – количество маршрутов, vi – вершина маршрута, T – сумма дистанций всех маршрутов.

## 3.4 Описание и обоснование выбора состава технических и программных средств

- Операционная система Windows версии 7 и выше

- Наличие экрана с разрешением не менее 640x480

- Размер оперативной памяти не менее 1 гигабайт

- Наличие 300 мегабайт свободного пространства на жестком диске.

# 4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## 4.1. Ориентировочная экономическая эффективность

В рамках данной работы расчёт экономической эффективности не предусмотрен.

## 4.2. Предполагаемая потребность

Данный продукт должен быть востребован в сфере решения задач маршрутизации, а также для наглядной демонстрации работы алгоритма имитации отжига.

## 4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

На момент начала разработки аналогов разрабатываемого программного обеспечения не нашлось в открытом доступе.

# 5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. A Simulated Annealing Algorithm for The Capacitated Vehicle Routing Problem H. Harmanani, D. Azar, N. Helal W. Keirouz Department of Computer Science & Mathematics Department of Computer Science Lebanese American University American University of Beirut Byblos, 1401 2010, Lebanon Beirut, 1107 2020, Lebanon.
9. Статья об алгоритме имитации отжига для задачи CVRP, Networking and Emerging Optimization [Электронный ресурс] // URL: <http://neo.lcc.uma.es/vrp/solution-methods/metaheuristics/simulated-annealing/>
10. Computer technologies department, ITMO University. Задачи маршрутизации транспорта [Электронный ресурс] // URL: http://rain.ifmo.ru/cat/view.php/theory/unsorted/vrp-2006

# Приложение 1

**Описание и функциональное назначение классов, структур и компонентов**

|  |  |
| --- | --- |
| Класс\Структура\Компонент | Назначение |
| AdvancedTimer | Служит для измерения времени |
| AnnealingSolver | Реализует исполнение алгоритма имитации отжига |
| ChartData | Содержит данные графика |
| GraphData | Содержит данные графа |
| InterfaceGate | Служит для связи с модулем интерфейса |
| RouteSolution | Содержит решение в виде набора маршрутов |
| App.vue | Является точкой запуска интерфейса |
| CarsInfo.vue | Является частью интерфейса со списком единиц транспорта |
| DisplayMode.vue | Является частью интерфейса с кнопкой переключения режима отображения |
| Files.vue | Является частью интерфейса с кнопками загрузки и сохранения файлов |
| Graph.vue | Является частью интерфейса с визуализацией графа |
| ProcessControl.vue | Является частью интерфейса с кнопками запуска, паузы и остановки алгоритма решения |
| Progress.vue | Является частью интерфейса с полосой выполнения алгоритма |
| ResultChart.vue | Является частью интерфейса с визуализацией графика результата |
| Routes.vue | Является частью интерфейса с визуализацией маршрутов |
| Settings.vue | Является частью интерфейса с настройками запуска алгоритма |
| Stats.vue | Является частью интерфейса с отображением параметров статистики |

# Приложение 2

**Описание и функциональное назначение полей, свойств и методов классов, структур и компонентов**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс AdvancedTimer** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| duration | private | long long | Значение продолжительности в миллисекундах | |
| active | bool | Флаг работы таймера | |
| last\_point | time\_point | Значение последнего временного момента | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| AdvancedTimer | public | конструктор | - | Конструктор таймера |
| run | void | - | Запускает таймер |
| pause | void | - | Останавливает таймер |
| value | long long | - | Возвращает длительность временного промежутка в миллисекундах |
| clear | void | - | Сбрасывает таймер |
| now | private | time\_point | - | Возвращает текущий момент времени |
| dist | long long | time\_point a, time\_point b | Находит длительность промежутка времени между двумя моментами времени в миллисекундах |
| **Класс AnnealingSolver** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| bestSolution | private | RouteSolution\* | Ссылка на объект лучшего решения | |
| currentSolution | RouteSolution\* | Ссылка на объект текущего решения | |
| chart | ChartData\* | Ссылка на объект данных графика | |
| timer | AdvancedTimer | Таймер для замера времени решения | |
| t | long double | Текущая температура | |
| factor | long long | Коэффициент охлаждения температуры | |
| tEnd | long long | Конечная температура | |
| timeLimit | long long | Лимит по времени в миллисекундах | |
| iterationsTotal | long long | Общее количество итераций | |
| iterationsCnt | long long | Текущее количество итераций | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| AnnealingSolver | public | конструктор | - | Инициализирует решение |
| ~AnnealingSolver | деструктор | - | Очищает память решения |
| isFinished | bool | - | Проверка не завершено ли решение |
| runIteration | void | - | Запустить итерацию |
| pause | public | void | - | Поставить решение на паузу |
| unpause | void | - | Продолжить решение |
| makeUpdate | PyObject\* | - | Упаковать текущее состояние для передачи интерфейсу |
| makeStats | private | PyObject\* | - | Упаковать параметры статистики текущего состояния для передачи интерфейсу |
| makeCars | PyObject\* | RouteSolution\* | Упаковать информацию о транспортных средствах текущего состояния для передачи интерфейсу |
| makeRoutes | PyObject\* | RouteSolution\* | Упаковать информацию о маршрутах текущего состояния для передачи интерфейсу |
| makeChart | PyObject\* | - | Упаковать точки графика результата текущего состояния для передачи интерфейсу |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс ChartData** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| MAX\_LEVEL | private | int | Константа – максимальное количество уровней сжатия | |
| DEFAULT\_MAX\_SIZE | int | Константа – максимальное количество точек графика по умолчанию | |
| max\_size | int | Максимальное количество точек графика | |
| size | int | Текущее количество точек графика | |
| levels | deque <pair <long long, double> >[] | Уровни сжатия, содержащие точки графика | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| ChartData | public | конструктор | int max\_size | Инициализирует контейнер точек графика |
| add\_node | void | long long x, double y | Добавляет точку в контейнер |
| get\_points | vector <pair <long long, double> > | - | Возвращает массив точек графика |
| compress\_data | private | void | - | Сжимает контейнер |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс GraphData** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| nodes | private | Node\* | Массив вершин | |
| n | public | int | Количество вершин | |
| c | int | Ограничение по грузоподъёмности | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| GraphData | public | конструктор | PyObject \*problem | Инициализирует граф |
| ~GraphData | деструктор | - | Удаляет граф из памяти |
| dist | double | int a, int b | Возвращает расстояние между вершинами |
| demand | int | int a | Возвращает нагрузку вершины |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс InterfaceGate** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| bridge\_dict | private | PyObject\* | Словарь функций доступа к мосту интерфейса | |
| state | PyGILState\_STATE | Состояние интерпретатора моста интерфейса | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| pylock | public | void | - | Занять мост интерфейса для обращения |
| pyunlock | void | - | Разблокировать мост интерфейса |
| init | void | - | Инициализировать интерфейс |
| extract\_event | PyObject\* | - | Получить событие интерфейса |
| sendAppEvent | void | string newState, PyObject \*update | Отправить событие в интерфейс |
| call | private | PyObject\* | string func, PyObject \*args | Вызвать функцию моста интерфейса |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс RouteSolution** | | | | |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Описание | |
| graph | public | GraphData\* | Ссылка на объект графа | |
| routes | vector <Route\*> | Массив маршрутов | |
| cost | double | Оценка решения | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Описание |
| RouteSolution | public | конструктор | GraphData \*graph | Инициализирует первое случайное решение |
| ~RouteSolution | public | конструктор | RouteSolution \*prev | Инициализирует новое решение путём случайного изменения предыдущего |
| count\_cost | private | void | – | Пересчитывает оценку решения |

|  |  |  |
| --- | --- | --- |
| **Компонент App.vue** | | |
| **Поля** | | |
| Имя | Описание | |
| store | Глобальное хранилище интерфейса | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| created | - | Инициализирует систему интерфейса |
| handleAppEvent | event | Обрабатывает полученное событие |

|  |  |  |
| --- | --- | --- |
| **Компонент CarsInfo.vue** | | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| cars | - | Извлекает из хранилища информацию о транспортных средствах |

|  |  |  |
| --- | --- | --- |
| **Компонент DisplayMode.vue** | | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| toggle | - | Сменить режим отображения |
| onlyBest | - | Получить текущий режим отображения |

|  |  |  |
| --- | --- | --- |
| **Компонент Files.vue** | | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| loadProblem | - | Загрузить задачу |
| loadSolution | - | Загрузить решение |
| resetSolution | - | Сбросить решение |
| file | - | Возвращает название текущего файла |
| status | - | Возвращает текущий статус в текстовом виде |
| problemLoadAbility | - | Возвращает флаг можно ли в данный момент загружать задачу |
| solutionLoadAbility | - | Возвращает флаг можно ли в данный момент загружать решение |
| solutionSaveAbility | - | Возвращает флаг можно ли в данный момент сохранять решение |
| solutionResetAbility | - | Возвращает флаг можно ли в данный момент сбросить решение |

|  |  |  |
| --- | --- | --- |
| **Компонент Graph.vue** | | |
| **Поля** | | |
| Имя | Описание | |
| vertices | Вершины графа | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| mounted | - | Инициализирует область для отрисовки |
| updateVertices | v | Обновляет вершины графа |
| redrawGraph | - | Перерисовывает граф |
| drawRoutes | ctx | Рисует маршруты |
| drawVertices | ctx | Рисует Вершины |

|  |  |  |
| --- | --- | --- |
| **Компонент ProcessControl.vue** | | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| launchSolve | - | Запускает решение |
| pauseSolve | - | Ставит решение на паузу |
| stopSolve | - | Останавливает решение |
| launchAbility | - | Возвращает флаг можно ли в данный момент запускать решение |
| pauseAbility | - | Возвращает флаг можно ли в данный момент ставить решение на паузу |
| stoppingAbility | - | Возвращает флаг можно ли в данный момент останавливать решение |

|  |  |  |
| --- | --- | --- |
| **Компонент Progress.vue** | | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| beforeProcess | - | Возвращает флаг, является ли текущее состояние предшествующим запуску решения |
| afterProcess | - | Возвращает флаг, является ли текущее состояние следующим после запуска решения |
| progress | - | Возвращает текущий прогресс решения в процентах |
| styleObj | - | Возвращает стиль полосы прогресса с установленной шириной в процентах |
| textStatus | - | Возвращает текущий статус прогресса выполнения |

|  |  |  |
| --- | --- | --- |
| **Компонент ResultChart.vue** | | |
| **Поля** | | |
| Имя | Описание | |
| currentPoints | Текущие точки графика | |
| bestPoints | Лучшие точки графика | |
| chart | График | |
| **Методы** | | |
| Имя | Аргументы | Описание |
| mounted | - | Инициализирует график |
| updateChart | v | Обновляет график |

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| изменённых | заменённых | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |