**ТЗ на решение задачи коммивояжера**

|  |  |
| --- | --- |
| **Задачи** | Создание ТЗ для функционала «Решение задачи коммивояжера» (части программной системы для решения задач дискретной оптимизации) |
| **Связанные задачи** | Функционал «Решение задач дискретной оптимизации» |
| **Автор** | Семенников Андрей |
| **Версия документа** | 1.0 |

**Оглавление**

* 1. Глоссарий
* 2. Общие положения
* 3. Описание процессов
* 4. Описание объектов и справочников
* 5. Описание интерфейсов
* 6. Алгоритмы

1. **Глоссарий**

|  |  |  |
| --- | --- | --- |
| **№** | **Термин** | **Определение** |
| **1** | Город | Сущность, представляющая собой номер в коллекции городов |
| **2** | Дорога | Сущность, представляющая собой числовую запись в коллекции городов для двух данных городов |
| **3** | Коллекция городов | Класс, хранящий матричное представление множества всех городов (в виде матрице смежности), существующих на данный момент в системе |
| **4** | Маршрут | Класс, представляющий собой последовательность посещенных городов и суммарную длину дорог, включенных в маршрут |
| **5** | Итоговое решение задачи (кратчайший маршрут) | Объект класса Маршрут, представляющий собой маршрут наименьшей протяженностью, начинающийся с города №1 и содержащий все имеющие дороги города хотя бы один раз |

1. **Общие положения**

Данное техническое задание описывает:

* 1. Процесс ввода городов
  2. Процесс ручного или автоматического ввода дорог между городами
  3. Процесс решения задачи и вывода результата в текстовом виде
  4. Процесс удаления всех введенных данных

# **Описание процессов**

* 1. **Ввод городов**

Процесс ввода городов – нажатие мышью в пустое место отведенной панели.

* 1. **Ручной и автоматический ввод дорог между городами**

В случае ручного ввода - последовательные нажатия мышью (всего 2) в окрестности заданных городов на отведенной панели и последующее нажатие мышью на одну из номерных кнопок, обозначающих вводимое число (по аналогии с калькулятором). Автоматическая генерация дорог между всеми городами (создание полного графа) происходит при нажатии пользователем мышью на кнопку «Generate».

* 1. **Решение задачи коммивояжера и вывод результата**

По нажатию пользователем мышью на кнопку «Start» запускается жадный алгоритм, формирующий решение задачи. Затем матрица смежности и решение задачи выводятся в виде текста в консоль среды исполнения.

* 1. **Удаление введенных данных**

По нажатию пользователем мышью на кнопку «Reset» происходит удаление всех введенных данных и очистка панели, отведенной для ввода городов и отображения дорог.

# **Описание объектов**

**Объект 1. Коллекция городов**

1. Функциональное описание
2. Хранит матрицу смежности городов, размер матрицы и количество существующих дорог.
3. Матрица может быть заполнена как вручную, так и автоматически:

* В случае ручного заполнения значение ячейки матрицы смежности на пересечении соответствующей строки и соответствующего столбца, а также симметричной ей относительно главной диагонали ячейки изменяется на номер нажатой пользователем номерной кнопки (например, нажатие на кнопку «7» меняет значение matrix[i][j] на 7).
* В случае автоматического заполнения соответствующая ячейка матрицы и симметричная ей относительно главной диагонали заполняется случайно выбранным числом от 1 до 9 включительно.

1. Структура данных объекта Коллекция городов:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Поле** | **Описание** | **Связанный объект** | **Тип** | **Обязательность** |
| 1 | Матрица | Поле для хранения информации о существующих городах, наличии и длине дорог между ними | нет | int[][] | Да |
| 2 | Размер | Поле для хранения информации о размере матрицы смежности | нет | int | Да |
| 3 | Количество дорог | Поле для хранения информации о количестве существующих дорог | нет | int | Да |

**Объект 2. Маршрут**

1. Функциональное описание
2. Хранит последовательность посещенных городов, их количество (включая повторные посещения) и суммарную длину дорог, входящих в этот маршрут.
3. Создается системой и заполняется во время работы алгоритма.
4. Структура данных объекта Маршрут:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Поле** | **Описание** | **Связанный объект** | **Тип** | **Обязательность** |
| 1 | Города | Поле для хранения информации о входящих в данный маршрут городах (включая повторные вхождения) | нет | ArrayList<Integer> | Да |
| 2 | Количество городов | Поле для хранения информации количестве входящих в данный маршрут городах (включая повторные вхождения) | нет | int | Да |
| 3 | Длина маршрута | Поле для хранения информации о сумме входящих в данный маршрут дорогах (включая повторные вхождения) | нет | int | Да |

1. **Описание интерфейсов**
   1. Интерфейс программы для решения задачи коммивояжера:

Общее описание:

|  |  |  |
| --- | --- | --- |
| **№** | **Свойства формы** | **Значения свойства** |
| **1** | Вид | Страница desktop приложения. На ней расположены: панель-холст для ввода и отображения городов и дорог между ними, управляющие кнопки «Start», «Reset» и «Generate», а также номерные кнопки «1», «2», «3», «4», «5», «6», «7», «8», «9» для ввода расстояния между городами |

Подробное описание интерфейса:

|  |  |  |
| --- | --- | --- |
| **№** | **Элемент формы** | **Описание элемента формы** |
| **1** | Панель-холст | AnchorPane, предназначенная для ввода городов нажатиями мыши, а также для отображения введенных городов и дорог между ними |
| **2** | Элемент управления «Reset» | Кнопка  При нажатии происходит удаление введенных данных и очистка панели-холста |
| **3** | Элемент управления «Start» | Кнопка  При нажатии запускается алгоритм решения задачи, выводится матрица смежности и итоговое решение |
| **4** | Элемент управления «Generate» | Кнопка  При нажатии происходит автоматическое заполнение матрицы смежности (создание дорог между всеми городами) и отображение созданных дорог в виде отрезков на панели-холсте |
| **5** | Элемент управления «1» | Кнопка  При нажатии происходит создание дороги длиной 1 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **6** | Элемент управления «2» | Кнопка  При нажатии происходит создание дороги длиной 2 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **7** | Элемент управления «3» | Кнопка  При нажатии происходит создание дороги длиной 3 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **8** | Элемент управления «4» | Кнопка  При нажатии происходит создание дороги длиной 4 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **9** | Элемент управления «5» | Кнопка  При нажатии происходит создание дороги длиной 5 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **10** | Элемент управления «6» | Кнопка  При нажатии происходит создание дороги длиной 6 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **11** | Элемент управления «7» | Кнопка  При нажатии происходит создание дороги длиной 7 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **12** | Элемент управления «8» | Кнопка  При нажатии происходит создание дороги длиной 8 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |
| **13** | Элемент управления «9» | Кнопка  При нажатии происходит создание дороги длиной 9 между выбранными городами и отображение созданной дороги в виде отрезка на панели-холсте |

# **Алгоритмы**

* 1. **Алгоритм ввода городов**

1. После нажатия на панель-холст проверяем, есть ли в указанном месте город
2. Если в месте клика уже есть город – ничего не происходит. Иначе – создаем город и отображаем его на панели-холсте.
   1. **Алгоритм ручного ввода дорог между городами**
3. После 2 последовательных кликов (по 1 на город) по 2 разным городам разблокируем номерные кнопки
4. После нажатия на одну из номерных кнопок меняем значения соответствующих ячейки матрицы смежности на число, равное номеру кнопки
5. Отображаем на панели-холсте дорогу в виде отрезка, соединяющего центры выбранных ранее кругов, обозначающих города

# **Алгоритм автоматического ввода дорог между городами**

1. В случае наличия более 1 города и нажатия на кнопку «Generate» генерируем полный граф (создаем дороги между каждой парой городов)
2. Отображаем дороги на панели-холсте в виде отрезков, соединяющих центры кругов
   1. **Алгоритм решения задачи и вывода результата**
3. Проверяем корректность ввода (есть ли дороги, ведущие в город №1 и нет ли кластеров городов, не соединенных между собой, например, «1»-«2» и «3»-«4»)
4. Если ввод некорректный – выводим сообщение об ошибке
5. Иначе – запускается жадный алгоритм, заполняющий объект shortestRoute класса Route
6. В виде текста в консоль среды исполнения выводится матрица смежности городов и итоговое решение задачи.
   1. **Алгоритм удаления введенных данных**
7. После нажатия на кнопку «Reset» происходит очистка панели-холста и инициализация использующихся структур данных