Как запускать программу – думаю, там ничего сложного. Как и набрать матрицу и нажать «Решить».  
  
Есть хорошее описание, как работает епсилон-метод(он служит для предотвращения вырожденности плана).  
  
<http://abc.vvsu.ru/Books/ebooks_iskt/%DD%EB%E5%EA%F2%F0%EE%ED%ED%FB%E5%F3%F7%E5%E1%ED%E8%EA%E8/%C8%F1%F1%EB%E5%E4%EE%E2%E0%ED%E8%E5%20%EE%EF%E5%F0%E0%F6%E8%E9/fmi.asf.ru/vavilov/tz61t.htm>  
  
Проект насыщен комментариями, но мой рецепт – полчаса минимум повтыкать в него.

Разберем конструктор класса транспортной задачи:  
// Конструктор

public Transport(ref RichTextBox rTB, double[,] стоимости, double[] потребители, double[] поставщики)

{

окно = rTB;

// Очистим окно

окно.Clear();

// Инициализируем переменный

Стоимости = стоимости;

Потребители = потребители;

Поставщики = поставщики;

КонечнаяСумма = 0;

КоличествоПоставщиков = Поставщики.Length;

КоличествоПотребителей = Потребители.Length;

NVeryLargeNumber = 99999999999;

nomer = 0;

// Т.к. это эпсилон метод - добавим небольшие величины к поставщикам и потребителям

var эпсилон = 0.001;

for (int i = 0; i < КоличествоПотребителей; i++)

{

Потребители[i] += (double)эпсилон / Потребители.Length;

}

окно.Text += '\n' + "Эпсилон: " + эпсилон;

Поставщики[1] += (double)эпсилон;

МатрицаПоискаОптимальногоПлана = new double[КоличествоПоставщиков, КоличествоПотребителей];

for (int i = 0; i < КоличествоПоставщиков; i++)

{

for (int j = 0; j < КоличествоПотребителей; j++)

{

МатрицаПоискаОптимальногоПлана[i, j] = 0;

}

}

ДвойственнаяМатрица = new double[КоличествоПоставщиков, КоличествоПотребителей];

for (int i = 0; i < КоличествоПоставщиков; i++)

{

for (int j = 0; j < КоличествоПотребителей; j++)

{

ДвойственнаяМатрица[i, j] = -1;

}

}

// Начинаем решение с запуска метода сев-западного угла

МетодСевероЗападногоУгла();

X\_КоординатаОпорнойТочки = -1;

Y\_КоординатаОпорнойТочки = -1;

ПечатьШага();

}

Сначала мы инициализируем наши переменные – присваиваем переданные массив матрицам стоимостей, потребителей, поставщиков и т.д. И проводим метод северо-западного угла с печатью полученного результат.  
Забиваем МатрицуПоискаОптимальногоПлана и ДвойственнуюМатрицу шаблонными стандартными значениями, чтобы потом их заполнить при обработке в методе Сев-Западного угла.

Непосредственно решение задачи начинается при вызове функции:

public void НачатьРешение()

{

// Обработка

while (ПроверкаНаОптимальность())

{

окно.Text+='\n'+("Опорная точка: [" + X\_КоординатаОпорнойТочки.ToString() +";"+ Y\_КоординатаОпорнойТочки.ToString()+"]");

СделатьОптимальным();

ПечатьШага();

}

окно.Text+='\n'+"КОНЕЦ" +'\n';

}

Пока проходит проверка на оптимальность полученного опорного плана – мы стараемся получить более оптимальный план и каждый раз печатаем результат проведенных вычислений. Какой же план оптимальный? По условию задачи – в опорных точках которого нет положительных элементов(кажется ☺). Но википедия говорит, что так и задача решается верно, википедия не врёт.

Какие же функции обрабатываются в проверке оптимальности: (по названиям и комментариям можно определить их):

ПолучитьДвойственную();

НайтиПутьЦикл();

ИскатьПоГоризонтали();

ИскатьПоВертикали();

СделатьОптимальным();