РОССИЙСКИЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Г.В. ПЛЕХАНОВА

ФАКУЛЬТЕТ МАТЕМАТИЧЕСКОЙ ЭКОНОМИКИ, СТАТИСТИКИ И ИНФОРМАТИКИ

Кафедра «Управления информационными системами и программирования»

**Лабораторная работа**

**по курсу «Технологии программирования»**

**на тему «Разработка и оценка ПС»**

Вариант №4

**Решение транспортной задачи линейного программирования методами:**

**- северо-западного угла**

**- выбора минимального элемента в строке**

**- Лебедева**

Язык программирования: C++

Выполнили студенты группы ДКТ-141Б

Харченко А.В. Д0082Б/14

Серов П.В. Д0067Б/14

Хандожко А.С. Д0086Б/14

Руководитель

Ковалевская Е.В.

Москва 2016г.

## Оглавление.

1. Разработка требований и внешнее проектирование ПС
   1. Цели создания ПС.
   2. Теоретическая часть.
      1. Постановка задачи.
      2. Алгоритмы решения задачи.
      3. Контрольное выполнение задачи.
   3. Описание спецификации на ПС.
   4. Описание диалога с пользователем.
2. Разработка ПС.
   1. Общие сведения о ПС.
   2. Входная информация.
   3. Выходная информация.
   4. Архитектура ПС
   5. Логическая структура ПС.
   6. Описание модулей.
   7. Вызов и загрузка.
3. Программа и методика испытаний программного средства.
   1. Тестирование и отладка программного средства.
      1. Программа проведения тестирования и отладки.
      2. Набор тестовых данных.
      3. Анализ результатов тестирования.
   2. Оценка ПС.
      1. Оценка качественных показателей.
      2. Оценка сложности.
      3. Оценка надёжности.
      4. Оценка экономических показателей.
   3. Исследование разработанного ПС.
4. Управление технической безопасностью с ПС.
5. Маркетинг и реклама.
6. Список литературы.
7. Приложения.
   1. Тексты модулей.
   2. Результаты тестирования и выполнения программы.
   3. Экранные формы.
   4. Заставка и реклама.

### ****Разработка требований и внешнее проектирование программного средства.****

**Проектирование** – процесс создания описания, необходимого для построения в заданных условиях еще не существующего объекта на основе первичного описания этого объекта

**Проектирование программного средства** — процесс преобразования входной информации об объекте проектирования о методах и об опыте проектирования в соответствии с требованиями проекта программного средства.

### ****1.1 Цели создания программного средства.****

1. **Краткое описание**. Данное программное средство разработано для решения транспортной задачи линейного программирования методами: «северо-западного угла», «двойного предпочтения», «Лебедева».
2. **Определение пользователей**. Пользователями могут быть компании и организации, использующие это программное средство для решения экономических задач например, транспортные перевозки, курьерские службы.
3. **Подробное описание функциональных задач.** Смотри пункт 1.2.
4. **Документация.** Руководство пользователя, предназначенное для сотрудников, работающих с данным программным средством.
5. **Эффективность.** Программное средство накладывает ограничения:
   1. количество поставщиков n, n<=10;
   2. количество потребителей m, m<=10;
   3. задача закрытого типа;
   4. значения А, В, С, Х – целые неотрицательные.

0 < A <=999

0 < B <= 999

0 < C <= 9

Программное средство не накладывает ограничения на:

* + 1. ресурсы;
    2. память.

1. **Совместимость.** При разработке программного средства учитывается международный стандарт ISO/IEC-12207 (1999 г.). Среда программирования – MS Visual Studio 2015, язык программирования C#.
2. **Конфигурация.** Минимальные системные требования технико-технические характеристики компьютера:

* Операционная система – Windows 7 и последующие версии;
* Процессор – частота не менее 1,6 Ггц;
* Устройства: мышь, клавиатура;
* Оперативная память 2 Гбайта и более.

1. **Безопасность.** В целях обеспечения безопасности программа использует систему паролей и контроля несанкционированного доступа
2. **Обслуживание.** Для обеспечения процесса тестирования и отладки по требованию пользователя было выделено 2 месяца.
3. **Установка.** Смотри пункт 2.7.
4. **Надёжность.** Для обеспечения надёжности программного средства используется множество проверок входных данных.

### ****1.2 Теоретическая часть. Предметная область.****

### Предметная область – это транспортная задача линейного программирования.

### 1.2.1 Постановка задач

Некоторый однородный продукт, сосредоточенный у m поставщиков в количестве ( i = 1..m) единиц соответственно, необходимо доставить n потребителям в количестве ( j = 1..n) единиц. Известна стоимость перевозки единицы груза от i-го поставщика к j-му потребителю.

Требуется определить базисный план поставок продукции, позволяющий вывести все грузы и полностью удовлетворить потребности потребителей.

Матрица исходных данных.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | … | Вj | … | Вn |
| A1 | C11 x11 | C12 x12 | C13 x13 | … | C1j x1j | … | C1n x1n |
| A2 | C21 x21 | C22 x22 | C23 x23 | … | C2j x2j | … | C2n x2n |
| A3 | C31 x31 | C32 x32 | C33 x33 | … | C3j x3j | … | C3n x3n |
| … | … | … | … | … | … | … | … |
| Ai | Ci1 xi1 | Ci2 xi2 | Ci3 xi3 | … | Cij xij | … | Cin xin |
| … | … | … | … | … | … | … | … |
| An | Cn1 xn1 | Cn2 xn2 | Cn3 xn3 | … | Cnj xnj | … | Cnn xnn |

Пусть – количество единиц груза, запланированных к перевозке от i-го поставщика к j-му потребителю.

Тогда PL = – стоимость всего плана.

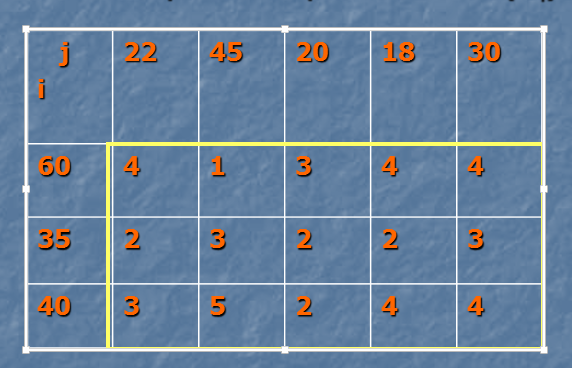
Ограничения:

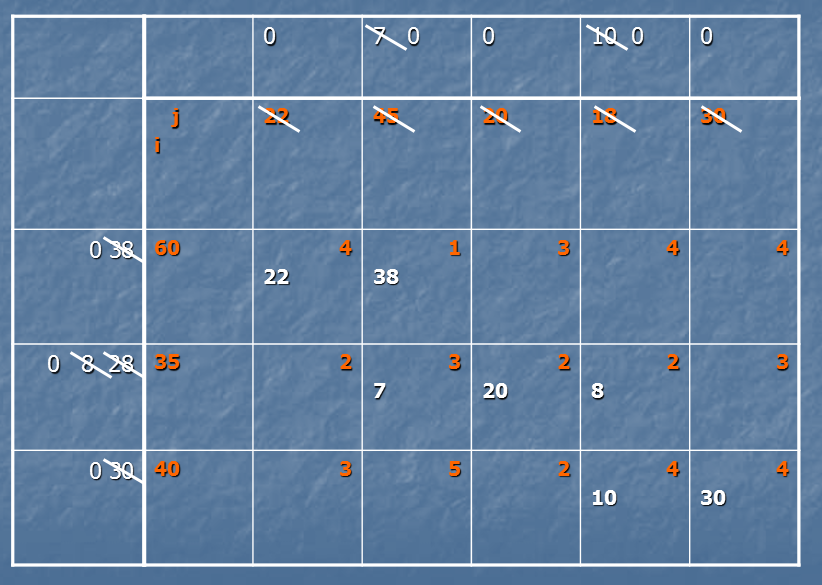
1. Xij>=0 - размер поставок положительный.
2. ΣXij=Аi, где j=1..n - объем поставок равен количеству продаж, т.е. все грузы должны быть перевезены.
3. ΣXij=Вj, где i=1..m - объем поставок равен спросу, т.е. Все потребности должны быть удовлетворены.
4. Количество продукции у поставщиков равняется количеству продукции, необходимого для удовлетворения потребителя.

**1.2.2** **Алгоритмы решения задач**

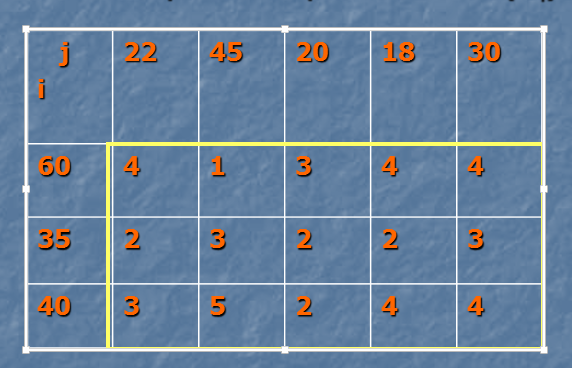
1. Метод северо-западного угла.

Данный метод легко алгоритмизировать, однако в подавляющем большинстве случаев он приводит к плану поставок, весьма далекому от оптимального.



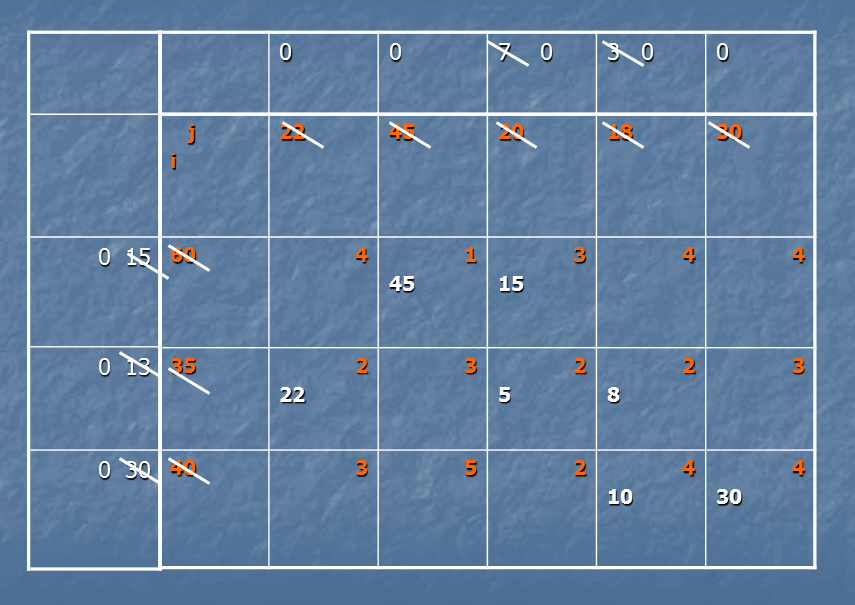
При этом способе «поставки» располагаются, начиная от левого верхнего и кончая нижним правым углом матрицы. На географических картах левый верхний угол соответствует северо-западу, эта аналогия и дала название способу

1. Метод минимального элемента в строке.

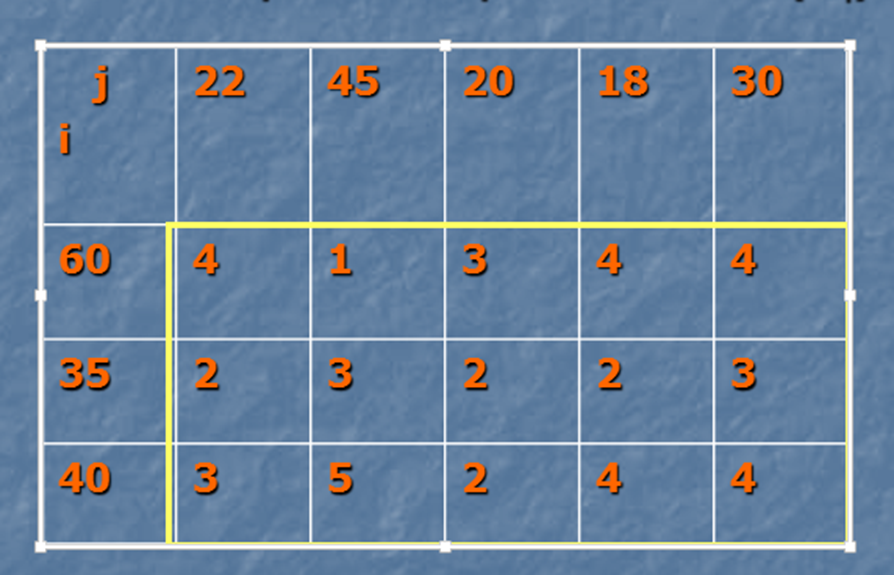


Метод заключается в том, что поочередно в строках матрицы отмечаются минимальные показатели и в соответствующие клетки заносятся поставки в зависимости от спроса и предложения. Если при записи поставок спрос по строке удовлетворен не полностью, ищется следующий по величине элемент в данной строке, отмечается новый элемент и так до полного удовлетворения спроса. Только после этого переходят на следующую строку.

Если в строке два или несколько одинаковых по величине минимальных показателя, то может быть отмечен любой из них.



1. Метод Лебедева.



1.Имеется матрица с спросом потребителей(Bi), с количеством продаж у поставщиков(Ai) и со стоимостью поставок(сij), так же имеются ограничения.

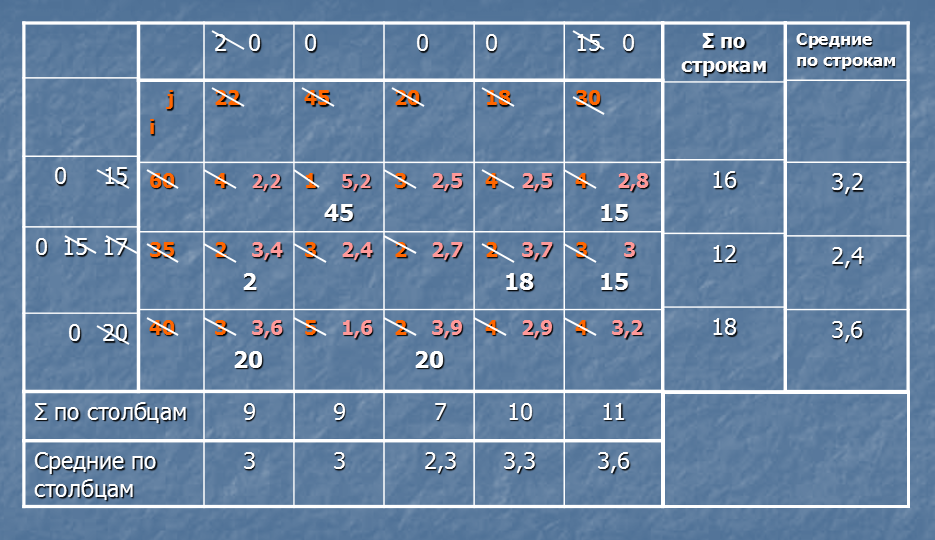
2.Для начала считаем сумму стоимостных показателей по столбцам и сумму стоимостных показателей по строкам.

3.Затем считаем среднее по столбцам, для этого сумму стоимостных показателей по столбцу делим на количество стоимостных показателей в этом столбце. Так считаем среднее по строкам.

4.Теперь для каждого стоимостного показателя складываем его среднее по столбцу и среднее по строке и из этой суммы вычитаем стоимостной показатель. Записываем в клетку стоимостного показателя новое значение, которое будем называть коэффициентом очередности. Проделываем эти действия с каждым стоимостным показателем.

Среднее по столбцу + среднее по строке - xij=коэффициент очередности

5.Разделение поставок мы начинаем с клетки с наибольшим коэффициентом очередности, далее идет следующий за ним по величине.



### 1.2.3 Контрольное выполнение задач

***Задача №1:*** Матрица(5x5). Харченко А. В.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 50 | 38 | 60 | 57 | 72 |
| 63 | 4 | 7 | 5 | 4 | 8 |
| 80 | 6 | 3 | 10 | 2 | 2 |
| 53 | 5 | 3 | 7 | 8 | 5 |
| 36 | 4 | 9 | 8 | 1 | 8 |
| 45 | 9 | 6 | 6 | 3 | 10 |

***Исходные данные:***

1. **Нахождение базисного плана методом северно-западного угла.**

***Ход решения:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0  ~~50~~ | 0  ~~25~~  ~~38~~ | 0  ~~5~~  ~~60~~ | 0  ~~9~~  ~~57~~ | 0  ~~45~~  ~~72~~ |
| 0 ~~13 63~~ | 4  50 | 7  13 | 5 | 4 | 8 |
| 0 ~~55 80~~ | 6 | 3  25 | 10  55 | 2 | 2 |
| 0 ~~48 53~~ | 5 | 3 | 7  5 | 8  48 | 5 |
| 0 ~~27 36~~ | 4 | 9 | 8 | 1  9 | 8  27 |
| 0 ~~45~~ | 9 | 6 | 6 | 3 | 10  45 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 50 | 38 | 60 | 57 | 72 |
| 63 | 4  50 | 7  13 | 5 | 4 | 8 |
| 80 | 6 | 3  25 | 10  55 | 2 | 2 |
| 53 | 5 | 3 | 7  5 | 8  48 | 5 |
| 36 | 4 | 9 | 8 | 1  9 | 8  27 |
| 45 | 9 | 6 | 6 | 3 | 10  45 |

***Выходные данные:***

Plbas = (4 \* 50) + (7 \* 13) + (3 \* 25) + (10 \* 55) + (7 \* 5) + (8 \* 48) + (1 \* 9) + (8 \* 27) + (10 \* 45) = 2010

Время решения: 3 минуты.

2. **Нахождение базисного плана методом минимального элемента в строке.**

***Ход решения:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0  ~~21~~  ~~44~~  ~~50~~ | 0  ~~30~~  ~~38~~ | 0  ~~45~~  ~~60~~ | 0  ~~57~~ | 0  ~~72~~ |
| 0 ~~6 63~~ | 4  6 | 7 | 5 | 4  57 | 8 |
| 0 ~~8 80~~ | 6 | 3  8 | 10 | 2 | 2  72 |
| 0 ~~23 53~~ | 5  23 | 3  30 | 7 | 8 | 5 |
| 0 ~~15 36~~ | 4  21 | 9 | 8  15 | 1 | 8 |
| 0 ~~45~~ | 9 | 6 | 6 | 3 | 10 |

45

***Выходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 50 | 38 | 60 | 57 | 72 |
| 63 | 4  6 | 7 | 5 | 4  57 | 8 |
| 80 | 6 | 3  8 | 10 | 2 | 2  72 |
| 53 | 5  23 | 3  30 | 7 | 8 | 5 |
| 36 | 4  21 | 9 | 8  15 | 1 | 8 |
| 45 | 9 | 6 | 6  45 | 3 | 10 |

Plbas = (4 \* 6) + (4 \* 57) + (3 \* 8) \* (2 \* 72) + (5 \* 23) + (3 \* 30) + (4 \* 21) + (8 \* 15) + (6 \* 45) = 1099

Время решения: 4 минуты.

3. **Нахождение базисного плана методом Лебедева.**

***Ход решения:***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0  ~~48~~  ~~50~~ | 0  ~~38~~ | 0  ~~15~~  ~~60~~ | 0  ~~13~~  ~~21~~  ~~57~~ | 0  ~~72~~ | Сумма в строке | Ср. значение в ячейке |
| 0 ~~48 63~~ | 7.2 4  48 | 4.2 7 | 7.8 5  15 | 5.4 4 | 4.2 8 | 28 | 5.6 |
| 0 ~~8 80~~ | 4.2 6 | 7.2 3 | 1.8 10 | 6.4 2  8 | 9.2 2  72 | 23 | 4.6 |
| 0 ~~13 15 53~~ | 6.2 5  2 | 8.2 3  38 | 5.8 7 | 1.4 8  13 | 7.2 5 | 28 | 5.6 |
| 0 ~~36~~ | 7.6 4 | 2.6 9 | 5.2 8 | 8.8 1  36 | 4.6 8 | 30 | 6 |
| 0 ~~45~~ | 3.4 9 | 6.4 6 | 8 6  45 | 7.6 3 | 3.4 10 | 34 | 6.8 |
| Сумма в столбце | 28 | 28 | 36 | 18 | 33 |  | |
| Ср. значение в ячейке | 5.6 | 5.6 | 7.2 | 3.8 | 6.6 |

***Выходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 50 | 38 | 60 | 57 | 72 |
| 63 | 4  48 | 7 | 5  15 | 4 | 8 |
| 80 | 6 | 3 | 10 | 2  8 | 2  72 |
| 53 | 5  2 | 3  38 | 7 | 8  13 | 5 |
| 36 | 4 | 9 | 8 | 1  36 | 8 |
| 45 | 9 | 6 | 6  45 | 3 | 10 |

Plbas = (4 \* 48) + (5 \* 15) + (2 \* 8) + (2 \* 72) + (5 \* 2) + (3 \* 38) + (8 \* 13) + (1 \* 36) + (6 \* 45) = 961

Время решения: 8 минут.

***Задача №2:*** Матрица (6 x 4). Выполнил Серов П. В.

***Исходные данные:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 200 | 130 | 120 | 100 | 250 | 100 |
| 500 | 2 | 4 | 3 | 8 | 1 | 6 |
| 100 | 4 | 3 | 5 | 6 | 2 | 9 |
| 150 | 8 | 3 | 5 | 4 | 4 | 3 |
| 150 | 1 | 3 | 6 | 7 | 3 | 2 |

1. **Нахождение базисного плана методом северно-западного угла.**

***Ход решения:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0  ~~200~~ | 0  ~~130~~ | 0  ~~120~~ | 0  ~~50~~  ~~100~~ | 0  ~~50~~  ~~200~~  ~~250~~ | 0  ~~100~~ |
| 0 ~~50 170 300 500~~ | 2  200 | 4  130 | 3  120 | 8  50 | 1 | 6 |
| 0 ~~50 100~~ | 4 | 3 | 5 | 6  50 | 2  50 | 9 |
| 0 ~~150~~ | 8 | 3 | 5 | 4 | 4  150 | 3 |
| 0 ~~100 150~~ | 1 | 3 | 6 | 7 | 3  50 | 2  100 |

***Выходные данные:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 200 | 130 | 120 | 100 | 250 | 100 |
| 500 | 2  200 | 4  130 | 3  120 | 8  50 | 1 | 6 |
| 100 | 4 | 3 | 5 | 6  50 | 2  50 | 9 |
| 150 | 8 | 3 | 5 | 4 | 4  150 | 3 |
| 150 | 1 | 3 | 6 | 7 | 3  50 | 2  100 |

Plbas = (2 \* 200) + (4 \* 130) + (3 \* 120) + (8 \* 50) + (6 \* 50) + (2 \* 50) + (4 \* 150) + (3 \* 50) + (2 \* 100) = 3030

Время решения: 3 минуты.

2. **Нахождение базисного плана методом минимального элемента в строке.**

***Ход решения:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0  ~~200~~ | 0  ~~30~~  ~~130~~ | 0  ~~70~~  ~~120~~ | 0  ~~80~~  ~~100~~ | 0  ~~250~~ | 0  ~~100~~ |
| 0 ~~50 250 500~~ | 2  200 | 4 | 3  50 | 8 | 1  250 | 6 |
| 0 ~~100~~ | 4 | 3  100 | 5 | 6 | 2 | 9 |
| 0 ~~20 120 150~~ | 8 | 3  30 | 5 | 4  20 | 4 | 3  100 |
| 0 ~~80 150~~ | 1 | 3 | 6  70 | 7  80 | 3 | 2 |

***Выходные данные:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 200 | 130 | 120 | 100 | 250 | 100 |
| 500 | 2  200 | 4 | 3  50 | 8 | 1  250 | 6 |
| 100 | 4 | 3  100 | 5 | 6 | 2 | 9 |
| 150 | 8 | 3  30 | 5 | 4  20 | 4 | 3  100 |
| 150 | 1 | 3 | 6  70 | 7  80 | 3 | 2 |

Plbas = (2 \* 200) + (3 \* 50) + (1 \* 250) + (3 \* 100) + (6 \* 70) + (7 \* 80) + (3 \* 30) + (4 \* 20) + (3 \* 100) = 2550

Время решения: 5 минут.

3. **Нахождение базисного плана методом Лебедева.**

***Ход решения:***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0  ~~150~~  ~~200~~ | 0  ~~80~~  ~~130~~ | 0  ~~120~~ | 0  ~~100~~ | 0  ~~20~~  ~~250~~ | 0  ~~100~~ | Сумма по строкам | Среднее по строкам |
| 0 ~~230 350 500~~ | 5.8 2  150 | 3.3 4 | 5.8 3  120 | 2.3 8 | 5.5 1  230 | 3 6 | 24 | 4 |
| 0 ~~80 100~~ | 4.6 4 | 4.1 3  80 | 4.6 5 | 5.1 6 | 5.3 2  20 | 0.8 9 | 29 | 4.8 |
| 0 ~~50 150~~ | 0.3 8 | 4.8 3  50 | 4.3 5 | 6.8 4  100 | 3 4 | 6.5 3 | 27 | 4.5 |
| 0 ~~50 150~~ | 6.5 1  50 | 4 3 | 2.5 6 | 3 7 | 3.2 3 | 6.7 2  100 | 22 | 3.7 |
| Сумма по столбцам | 15 | 13 | 19 | 25 | 10 | 20 |  | |
| Среднее по столбцам | 3.8 | 3.3 | 4.8 | 6.3 | 2.5 | 5 |

***Выходные данные:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 200 | 130 | 120 | 100 | 250 | 100 |
| 500 | 2  150 | 4 | 3  120 | 8 | 1  230 | 6 |
| 100 | 4 | 3  80 | 5 | 6 | 2  20 | 9 |
| 150 | 8 | 3  50 | 5 | 4  100 | 4 | 3 |
| 150 | 1  50 | 3 | 6 | 7 | 3 | 2  100 |

Plbas = (2 \* 150) + (3 \* 120) + (1 \* 230) + (2 \* 20) + (3 \* 80) + (3 \* 50) + (4 \* 100) + (1 \* 50) + (2 \* 100) = 1970

Время решения: 7 минут.

***Задача №3:*** Матрица(5x5). Выполнил Хандожко А. С.

***Исходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 45 | 75 | 30 | 24 | 86 |
| 38 | 5 | 6 | 3 | 2 | 4 |
| 55 | 4 | 2 | 8 | 1 | 2 |
| 72 | 3 | 4 | 7 | 6 | 5 |
| 35 | 8 | 2 | 4 | 7 | 1 |
| 60 | 2 | 3 | 12 | 4 | 8 |

1. **Нахождение базисного плана методом северно-западного угла.**

***Ход решения:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0  ~~7~~  ~~45~~ | 0  ~~27~~  ~~75~~ | 0  ~~30~~ | 0  ~~9~~  ~~24~~ | 0  ~~60~~  ~~86~~ |
| 0 ~~38~~ | 5  38 | 6 | 3 | 2 | 4 |
| 0 ~~48 55~~ | 4 | 2  48  7 | 8 | 1 | 2 |
| 0 ~~15 45 72~~ | 3 | 4  27  30  15 | 7 | 6 | 5 |
| 0 ~~26 35~~ | 8 | 2 | 4 | 7  9 | 1  26 |
| 0 ~~60~~ | 2 | 3 | 12 | 4 | 8  60 |

***Выходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 45 | 75 | 30 | 24 | 86 |
| 38 | 5  38 | 6 | 3 | 2 | 4 |
| 55 | 4  7 | 2  48 | 8 | 1 | 2 |
| 72 | 3 | 4  27 | 7  30 | 6  15 | 5 |
| 35 | 8 | 2 | 4 | 7  9 | 1  26 |
| 60 | 2 | 3 | 12 | 4 | 8  60 |

Plbas = (5\* 38) + (4 \* 7) + (2 \* 48) + (4 \* 27) + (7 \* 30) + (6 \* 15) + (7 \* 9) + (1 \* 26) + (8 \* 60) = 1291

Время решения: 3 минуты.

2. **Нахождение базисного плана методом минимального элемента в строке.**

***Ход решения:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0  ~~45~~ | 0  ~~20~~  ~~75~~ | 0  ~~16~~  ~~30~~ | 0  ~~24~~ | 0  ~~44~~  ~~79~~  ~~86~~ |
| 0 ~~14 38~~ | 5 | 6 | 3 | 2  14  24  55  45  20  7  35  16  44 | 4 |
| 0 ~~55~~ | 4 | 2 | 8 | 1 | 2 |
| 0 ~~7 27 72~~ | 3 | 4 | 7 | 6 | 5 |
| 0 ~~35~~ | 8 | 2 | 4 | 7 | 1 |
| 0 ~~16 60~~ | 2 | 3 | 12 | 4 | 8 |

***Выходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 45 | 75 | 30 | 24 | 86 |
| 38 | 5 | 6 | 3  14 | 2  24 | 4 |
| 55 | 4 | 2  55 | 8 | 1 | 2 |
| 72 | 3  45 | 4  20 | 7 | 6 | 5  7 |
| 35 | 8 | 2 | 4 | 7 | 1  35 |
| 60 | 2 | 3 | 12  16 | 4 | 8  44 |

Plbas = (3 \* 14) + (2 \* 24) + (2 \* 55) \* (3 \* 45) + (4 \* 20) + (5 \* 7) + (1 \* 35) + (12 \* 16) + (8 \* 44) = 1029

Время решения: 5 минут.

3. **Нахождение базисного плана методом Лебедева.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0  ~~45~~ | 0  ~~60~~  ~~75~~ | 0  ~~30~~ | 0  ~~24~~ | 0  ~~12~~  ~~20~~  ~~51~~  ~~86~~ | Сумма по строкам | Среднее по строкам |
| 0 ~~8~~ ~~38~~ | 3,4 5 | 1,4 6 | 7,8 3  30 | 6 2 | 4 4  8 | 20 | 4 |
| 0 ~~31~~ ~~55~~ | 3,8 4 | 4,8 2 | 2,2 8 | 6,4 1  24 | 5,4 2  31 | 17 | 3,4 |
| 0 ~~12~~ ~~72~~ | 6,4 3 | 4,4 4  60 | 4,8 7 | 3 6 | 4 5  12 | 25 | 5 |
| 0 ~~35~~ | 0,8 8 | 5,8 2 | 7,2 4 | 1,4 7 | 7,4 1  35 | 22 | 4,4 |
| 0 ~~15~~ ~~60~~ | 8,2 2  45 | 6,2 3  15 | 0,6 12 | 5,8 4 | 1,8 8 | 29 | 5,8 |
| Сумма по столбцам | 22 | 17 | 34 | 20 | 20 |  |  |
| Среднее по столбцам | 4,4 | 3,4 | 6,8 | 4 | 4 |  |  |

***Выходные данные:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 45 | 75 | 30 | 24 | 86 |
| 38 | 5 | 6 | 3  30 | 2 | 4  8 |
| 55 | 4 | 2 | 8 | 1  24 | 2  31 |
| 72 | 3 | 4  60 | 7 | 6 | 5  12 |
| 35 | 8 | 2 | 4 | 7 | 1  35 |
| 60 | 2  45 | 3  15 | 12 | 4 | 8 |

### 1.3 Описание спецификации на программное средство.

**1) Описание входных данных.** Смотри пункт 2.2

**2) Описание выходных данных.** Смотри пункт 2.3.

**3) Преобразование системы**

Северо-западного угла

да

А, В, С

Проверка

Min элемента в строке

Выход

Pαbaz   
базисный план

Лебедева

нет

1. **Характеристика надёжности.** Для обеспечения надёжности программного средства используется множество проверок входных данных.
2. **Эффективность.** Программное средство накладывает ограничения:
   1. количество поставщиков n, n<=10;
   2. количество потребителей m, m<=10;
   3. задача закрытого типа;
   4. значения А, В, С, Х – целые неотрицательные.

0 < A <=999

0 < B <= 999

0 < C <= 9

Программное средство не накладывает ограничения на:

* 1. ресурсы;
  2. память.

## 1.4 Описание диалога с пользователем.

### Разработка программного средства решения транспортной задачи

### 2.1 Общие сведения о программном средстве

1. Количество выполняемых операторов, количество строк: 241 оператор, 662 строки.
2. Количество переменных: 50
3. Количество модулей процедур: 6
4. Среднее время решения задачи: 5 мин.
5. Машинное время для каждого метода: 0.140 c
6. Трудоёмкость решения задачи (чел/час): 0.028
7. Количество специалистов: 3
8. Язык программирования: С++

### 2.2 Входная информация

1) Массив А размерности n – массив поставщиков. Массив, в котором указано количество товара, которое могут предоставить каждый производитель. 0<n<=10, 0<A<=999.

2) Массив B размерности m – массив потребителей. Массив, в котором указано количество товара, которое потребители хотят купить. 0<m<=10, 0<B<=999.

3) Матрица C – матрица n на m стоимости, в которой указано по какой цене за единицу продукции производитель готов предоставить свой товар клиенту. 0<n<=10, 0<m<=10, 0<C<=10.

Ограничения количества товара, имеющего у поставщиков должно быть равно количеству товаров у потребителя.

### 2.3 Выходная информация

1) PL(base) - план поставок продукции, позволяющий вывести все грузы и полностью удовлетворить потребности потребителей.

2) Матрица, состоящая из элементов Хij – количество единиц груза, запланированных к перевозке от i-го поставщика к j-му потребителю.

3) Выходные сообщения об ошибках:

Ошибка в вводе из файла – если выбранный файл не содержит корректно заданной матрицы.

Неверные входные данный – если матрица, над которой осуществляется нахождение базисной матрицы методами Лебедева или Минимального элемента в строке, не соответствует входной информации см 2.2.

Матрица не была сохранена – если матрица, для которой осуществляется сохранение, не соответствует входной информации см 2.2.

1. Справка об авторах, о методах, о данных, о программе.
2. Проверка на закрытый ли тип задачи.

### 2.4 Архитектура программного средства

Методы, основанные на доп. вычислениях

Прямые методы

Метод составления базисного плана

Транспортная задача

Методы оптимизации опорного плана

Потенциалов

Лебедева

Северо-западный угол

Квадратов

Фогеля

Юго-восточный угол

Лебедева-Тихомирова

Распределительный

Минимум в строке

Венгерский

Минимум в столбце

Форда-Фулкерсона

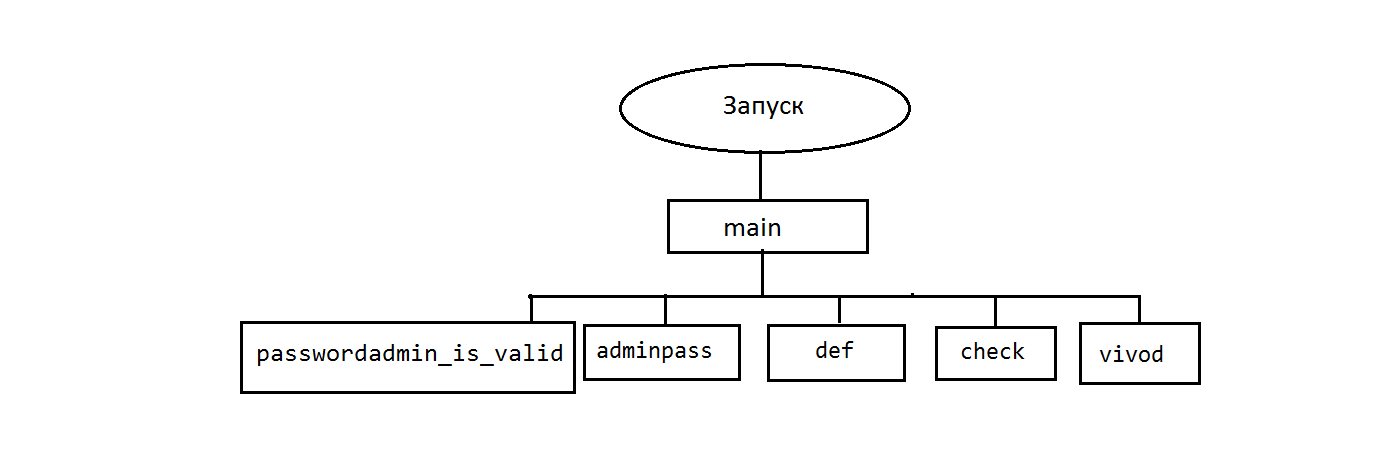
Двойное предпочтение

Разрешающих слагаемых

Минимального элемента в матрице

Дифференциальных рент

### 2.5 Логическая структура



### 2.6 Описание основных модулей функции

main – основной модуль. Вызывает другие модули. Здесь содержится меню программы.

vivod – модуль, выводящий конечную матрицу на экран пользователя.

check- модуль, проверяющий введенные значения на соответствия.

def- модуль вызывается, когда введено неверное значение.

adminpass- модуль введения пароля пользователя.

passwordadmin\_is\_valid – модуль проверки пароля.

### 2.7 Вызов и загрузка программы

Для запуска программы через «Проводник» операционной системы Windows проделайте следующие шаги:

1. Разархивируйте архив с помощью программы 7-Zip или WinRAR
2. Запустите файл «Start.exe»

Для вызова решения программы через «Visual Studio Community 2015» проделайте следующие шаги:

1. Откройте программу Visual Studio Community 2015
2. Выберите «Открыть -> Проект»
3. Найдите в скачанном ранее архиве файл «лаб\_раб\_ТП.sln» и запустите его

### 3. Программа и методика испытаний программного средства

### 3.1 Тестирования и отладка программного средства.

Тестирование – один из этапов ЖЦ ПО, процесс многократного выполнения ПО с целью обнаружения ошибок.

Результат тестирования – исходные данные для отладки.

Каждый тест определяет свой набор исходных данных и условий для запуска программы. При детерминированном тестировании контролируется каждая комбинация исходных данных и результатов.

Эффективный набор тестовых данных – набор, который может выявить проблемы.

Тестирование обеспечивает: обнаружение ошибок, демонстрация функции и ее назначение, демонстрацию реализацию требований программы.

Методы:

1. Структурное тестирование – как белый ящик
2. Функциональное – как черный ящик

Черный ящик – не видно ничего в ПО, изучение по входу и выходу.

Белый ящик – известно все, включая логику программы и структуру.

### 3.1.1 Программа проведения тестирования и отладки

1. Проверка входа (логина и пароля)
2. Проверка справки об авторах
3. Проверка справки о методах
4. Проверка справки о данных
5. Проверка справки о программе
6. Проверка меню (доступность навигации)
7. Проверка построения базового плана методом северо-западного угла
8. Проверка построения базового плана методом минимального элемента cтроки
9. Проверка построения базового плана методом Лебедева
10. Проверка выхода с подтверждением
11. Проверка на ввод достоверных данных

### 3.1.2 Набор тестовых данных

|  |  |  |
| --- | --- | --- |
| **№** | **Название теста и входные данные** | **Ожидаемые входные данные** |
| 1 | Проверка построения базового плана методом северо-западного угла  10 20 30 40 50 10 20 30 40 50  1 3 4 2 3 4 5 6 7 8 5 6 7 5 4 3 4 5 6 4 3 4 5 | См. пункт 7.2.1 |
| 2 | Проверка построения базового плана методом минимального элемента в строке  10 20 30 40 50 10 20 30 40 50  1 3 4 2 3 4 5 6 7 8 5 6 7 5 4 3 4 5 6 4 3 4 5 | См. пункт 7.2.2 |
| 3 | Проверка построения базового плана методом Лебедева  10 20 30 40 50 10 20 30 40 50  1 3 4 2 3 4 5 6 7 8 5 6 7 5 4 3 4 5 6 4 3 4 5 | См. пункт 7.2.3 |
| 4 | Проверка справки об авторах, Проверка справки о методах, Проверка справки о данных, Проверка справки о программе, Проверка меню (доступность навигации) | См. пункт 7.2.4 |
| 5 | Проверка входа (логина и пароля) | См. пункт 7.2.6 |
| 6 | Проверка выхода с подтверждением | См. пункт 7.2.7 |
| 7 | Проверка на ввод достоверных данных | См. пункт 7.2.8 |

### 3.1.3 Анализ результатов тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № | Определение ошибки | Ошибочное/исправленное написание | Пояснение |
| 1 | Ошибка вычисления | ((t2 - t1) / CLOCKS\_PER\_SEC)/ ((double)(t2 - t1) / CLOCKS\_PER\_SEC) | Деление различных типов данных друг на друга |
| 2 | Логическая ошибка | int pos\_nw = new int[atoi(d)]/ int \*pos\_nw = new int[atoi(d)]; | Ошибка при неправильной инициализации динамического массива |
| 3 | Логическая ошибка | if (sum1 = sum2)/ if (sum1 != sum2) | Присвоение вместо сравнения |
| 4 | Ошибка ввода-вывода | printf (“| |”)/printf (“\n| |”) | Ошибка при формировании вывода в виде матрицы |
| 5 | Ошибка вычисления | s = ((res[i][j]) \* (sp[i][j]))/  s = s + ((res[i][j]) \* (sp[i][j])); | Неправильный подсчет суммы |
| 6 | Логическая ошибка | if (pos[i] == 0) continue; / if (pos[i] == 0) break; | Вместо выхода из цикла происходит просто переход к следующей итерации цикла |

### 3.2 Оценка программного средства

### 3.2.1 Оценка качественных показателей

|  |  |  |  |
| --- | --- | --- | --- |
| **Показатель** | **Сущность показателя** | **Экспертная оценка(вес)**  **Wi** | **Оценка, установленная экспериментом**  **Ri** |
| Понятность | Степень понятности – степень, в которой ПС позволяет оценивающему лицу понять назначение ПС. | 0,3 | 0.5 |
| Удобство эксплуатации | Есть возможность обновления ПС в соответствии с новыми требованиями. Это свойство предполагает понятность, оцениваемость и простоту внесения изменений, которая необходима для устранения обнаруженных дефектов, для добавления новых возможностей или приспособления программы для работы на другой ЭВМ | 0,15 | 0.4 |
| Надежность | Можно ожидать, что ПС будет удовлетворительно выполнять необходимые функции. | 0.3 | 0.3 |
| Структурированность | Взаимосвязанные части ПС организованы в единое целое определенным образом. | 0,15 | 0.5 |
| Информативность | ПС содержит информацию, необходимую и достаточную для понимания читающим лицом назначения ПС, принятых допущений, существующих ограничений, исходных данных, результатов ПС при его функционировании | 0,1 | 0.7 |

Итого: 0.3\*0.5 + 0.15\*0.4 + 0.3\*0.3 + 0.15\*0.5 + 0.1\*0.7 = 0.15+0.06+0.09+0.075+0.07=0.445

**Программное средство обладает свойством:**

* **Понятности** в той степени, в которой оно позволяет оценивающему лицу понять назначение программного средства
* **Завершенность**. ПС обладает свойством завершенности, если в нем присутствует все необходимые компоненты, каждый из которых разработан всесторонне. Завершенность предполагает замкнутость описания и живучести программы.
* **Осмысленность.** Документация программного средства не содержит избыточной информации.
* **Мобильность**. Программное средство может легко и эффективно использоваться для работы на ЭВМ иного типа, чем та, для которой оно предназначено
* **Согласованность.** 2 разновидности: внутренняя (всюду содержит данную нотацию, терминологию.
* **Удобство эксплуатации.**
* **Оцениваемость.**
* **Полезность.**
* **Надежность.**
* **Структурированность.**

### 3.2.2 Оценка сложности

1. Берем данные из п. 2.1 (общие сведения) используя рисунок 1 определяем сложность программы.
2. Оцениваем вычислительную сложность:

* Временная (время решения задачи машинное),
* Программная (количество операторов объём памяти, занятой данное программы оперативная память, которая она занимает место)
* Информационная (место, которое предусмотрено для данных сколько памяти займет информационная сложность).

1. Метод Холстеда.

* Мера длины (формулы в презентации)
* Объем модуля

**Мера длины**: 

где n1 = 241 – число операторов;

n2 = 263– число операндов

**Расчет меры длины**: N=241\*log2241 + 263\*log2263 =241\*7,913+ 263\*8,039=1907,033 + 2114,257 = 4021,29

**Объем модуля**: , где n1 – число операторов; n2 – число операндов

**Расчет объема модуля**: V= 4021,29\* =4021,29\*8,977 = 36099,1203

### Оценка надёжности

### Надёжность – свойство программного средства сохранять работоспособность в течение определенного периода времени, в определенных условиях эксплуатации с учетом последствий для использования каждого отказа.

|  |  |  |
| --- | --- | --- |
| Тип ошибки | Вероятность появления (ai) | Количество выявленных ошибок |
| 1.Ошибки вычисления (A) | 0.07 | 4 |
| 2.Логические ошибки (B) | 0.31 | 8 |
| 3.Ошибки ввода-вывода (C,E) | 0.13 | 3 |
| 4.Ошибки манипулирования данными (D) | 0.15 | 2 |
| 5.Ошибки сопряжения (F) | 0.2 | - |
| 6.Ошибки определения данных (G) | 0.1 | - |

**Алгоритм решения:**



**Решение:**

R = 83/100 + (0,07\*(4-1))/100 + (0,31\*(8-1))/100 + (0,13\*(3-1))/100 + (0,15\*(2-1))/100 = 0,9 + 0,0021 + 0,0217 + 0,0026 + 0,0015 = 0,9279

* + 1. **Оценка экономических показателей**

1. Потрачено часов – 36
2. Стоимость часа – 810 рублей
3. Стоимость разработки ПС – 29 160р
4. Суммарное время решения задач в ручную – 45 минут
5. Суммарное время решения задач машинным путем – 0,005 минуты
6. Экономия времени – 44,995 минут
7. Задача решается всеми методами 5 раз в день
8. В месяце 25 рабочих дней
9. Стоимость решения до внедрения – 1500р
10. После внедрения - 750р
11. Окупаемость – 1,5месяца

Среднее = (1500 - 750) \* 5 \* 25 = 93750

Окупаемость = 29160 / 93750 = 0,31 месяца

### Исследование характеристик программного средства решения транспортной задачи

1. Зависимость времени решения ТЗ от размерности матрицы.
2. Сравнительная характеристика зависимости времени решения ТЗ по программе и вручную;
3. Сравнительная характеристика базисных планов.

### 4. Управление технологической безопасностью

Безопасность данных – защита данных от случайного или преднамеренного разрушения, раскрытия или модификации.

Алгоритм:

bool passwordadmin\_is\_valid(string password)

{

string valid\_pass = "admin";

if (valid\_pass == password)

return true;

else

return false;

}

void adminpass()

{

string user\_pass;

cout << "Введите пароль: ";

cin >> user\_pass;

system("cls");

if (passwordadmin\_is\_valid(user\_pass))

{

cout << "Доступ разрешен." << endl;

}

else

{

cout << "Неверный пароль!" << endl;

adminpass();

}

}

Блок-схема:

Неправильно

Проверка логина

Проверка пароля

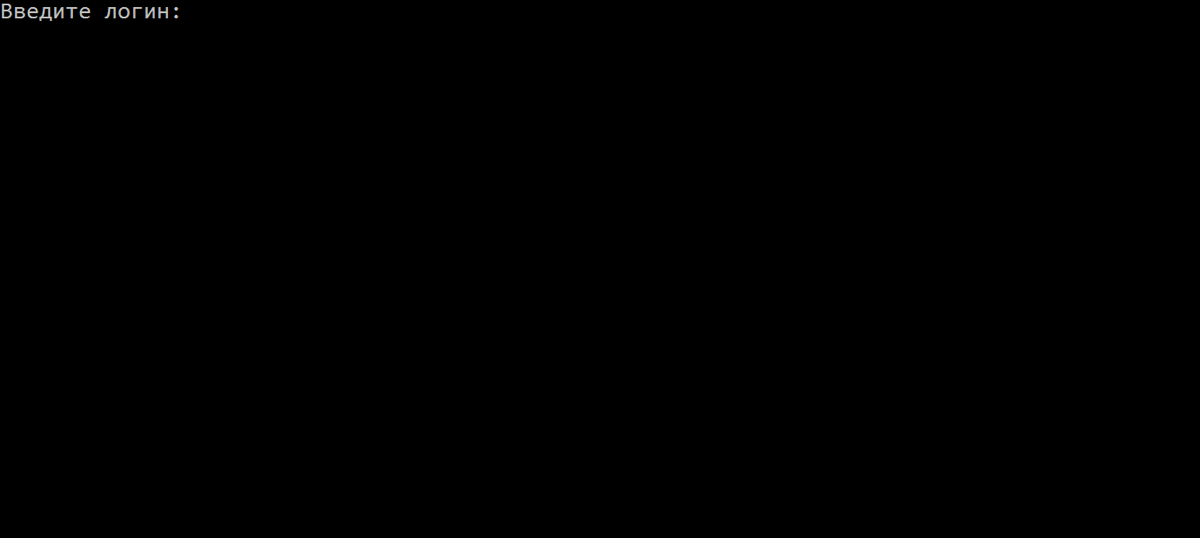
Неправильно

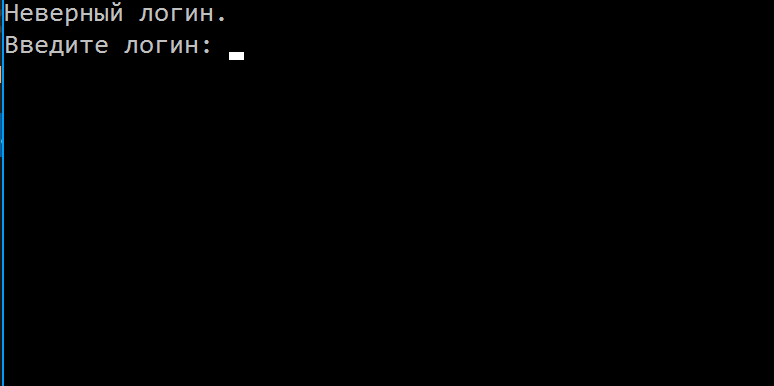
Правильно

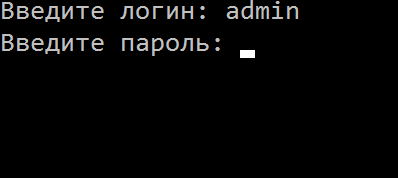
Правильно

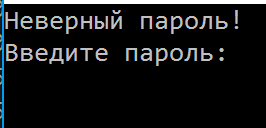
Вход в программу

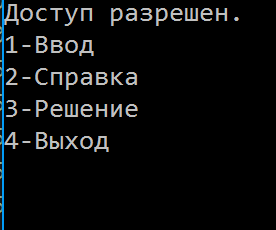
Скриншоты:











### 5. Маркетинг и реклама

**Маркетинг** - это процесс управления, включающий в себя определение, прогнозирование и удовлетворение запросов потребителей с получением прибыли. Наиболее верным является последнее, так как отражает интересы потребителей и продавцов. В целом маркетинг рассматривается как концепция и философия бизнеса, как ориентация организации, и как социальный и управленческий процесс, оптимально реализующий себя в условиях рынка.

Маркетинг как научно-практическая теория управления производственно-сбытовой деятельностью предполагает выполнение ряда функций:

— аналитические: изучение рынка, потребителей, товарной структуры;

— производственные: организация производства новых товаров, внедрение новых технологий, управление качеством и конкурентоспособностью продукции;

— сбытовые: организация системы товародвижения, сервиса, формирование спроса и стимулирование сбыта, проведение целенаправленной товарной и ценовой политики;

— управления и контроля: организация стратегического и оперативного планирования на предприятии, информационное обеспечение, организация системы коммуникаций и контроля маркетинга.

**Реклама** – платное или бесплатное, однонаправленное и неличное обращение, осуществляемое через СМИ и др. формы коммуникации, имеющее своей целью заставить потребителей выполнить желаемое рекламодателем действие.

Основная цель рекламы: заставить потребителя выполнить желаемое рекламодателем действие (убедить потребителя купить товар, воспользоваться услугой, придать известность фирме, содействовать популярности политического деятеля, внушить аудитории социально значимые мысли и т.д.).

**Функции рекламы:**

* Экономическая
* Социальная
* Идеологическая
* Маркетинговая
* Коммуникативная

# 6. Список литературы

1. Гост РИСО/МЭК 122079: интернет ресурс / <http://docs.cntd.ru/document/gost-r-iso-mek-12207-2010> - 2010г.
2. ГОСТ РИСО/МЭК 9126: интернет ресурс / <http://docs.cntd.ru/document/gost-r-iso-mek-9126-93> - 1993г.
3. Методы программирования: учебно-методическое пособие / Е.В. Ковалевская, Н.В. Комлева. — Москва: Евразийский открытый институт, 2011. — 320 с
4. Мастяева И.Н., Методы оптимизации / Мастяева И.Н., Семенихина О.Н. - М.: МЭСИ, 2003. – 135с.
5. Шилдт, Герберт. Ш57 C# 4.0: полное руководство.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2011 1056 с.: ил. — Парал. тит. англ.
6. Ковалевская Е.В., Комлева Н.В. «Методы программирования ПО», М., МЭСИ, 2011. -294с
7. Ковалевская Е.В. «Метрология, качество и сертификация программного обеспечения», М., МЭСИ, 2008. -94с.
8. Юдин Д.Б., Гольштейн Е.Г. «Задачи и методы линейного программирования: Задачи транспортного типа.» Изд.3, 2010 – 184с

# 7. Приложения

**7.1. Тексты модулей.**

#include <iostream>

#include <stdio.h>

#include <algorithm>

#include <string>

#include <time.h>

using namespace std;

int sp[50][50];

int \*pok = new int[50];

int \*pos = new int[50];

bool passwordadmin\_is\_valid(string password)

{

string valid\_pass = "admin";

if (valid\_pass == password)

return true;

else

return false;

}

void adminpass()

{

string user\_pass;

cout << "Введите пароль: ";

cin >> user\_pass;

system("cls");

if (passwordadmin\_is\_valid(user\_pass))

{

cout << "Доступ разрешен." << endl;

}

else

{

cout << "Неверный пароль!" << endl;

adminpass();

}

}

void def()

{

system("cls");

cout << "Вы ввели неверное значение" << endl << "Нажмите любую клавишу" << endl;

system("pause");

}

int check(char \*simb) {

int dlin;

int schet;

dlin = strlen(simb);

schet = 0;

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

if (isdigit(simb[i])) {

schet++;

}

if (schet == dlin)

return 1;

else

return 0;

}

void vivod(char \*c, char \*d, int shipment[50][50]) {

cout << "Решение транспортной задачи: " << endl;

int i, j;

printf ("\n\n");

for (j=0; j<(atoi(c)+1); j++)

printf (" \_\_\_\_\_\_\_");

printf ("\n");

printf ("|       |");

for (j=0; j<atoi(c); j++)

printf ("       |");

printf ("\n");

printf ("|       |");

for (j=0; j<atoi(c); j++)

printf (" %4d  |", pok[j]);

printf ("\n");

printf ("|\_\_\_\_\_\_\_|");

for (j=0; j<atoi(c); j++)

printf ("\_\_\_\_\_\_\_|");

for (i = 0; i<atoi(d); i++)

{

printf("\n");

printf("|       |");

for (j = 0; j<atoi(c); j++)

printf("%3d|   |", sp[i][j]);

printf("\n");

printf("|       |");

for (j = 0; j<atoi(c); j++)

printf("\_\_\_|   |");

printf("\n");

printf("| %4d  |", pos[i]);

for (j = 0; j<atoi(c); j++)

printf("       |");

printf("\n");

printf("|       |");

for (j = 0; j<atoi(c); j++)

printf("  %4d |", shipment[i][j]);

printf("\n");

printf("|\_\_\_\_\_\_\_|");

for (j = 0; j<atoi(c); j++)

printf("\_\_\_\_\_\_\_|");

}

cout << endl;

}

int main()

{

char test[256];

char input\_log[21];

string admin = "admin";

clock\_t t2, t1;

char str\_array[20][6];

int s = 0;

int dlin;

int schet;

int min\_val;

int ksk;

int  posm[50];

int pokm[50];

int minim;

int y;

int k = 0;

int res[50][50];

int l;

int g = 0;

int r;

int u;

int o;

int p;

int h = 0;

int f = 0;

int mini = 0;

int minj = 0;

float possr[50];

float poksr[50];

float sr[50][50];

int v;

int posl[50];

int pokl[50];

char a [256];

char c [256];

char d [256];

char e [256];

char x [256];

int sum1 = 0;

int sum2 = 0;

int shipment [50][50];

setlocale(LC\_ALL,"Rus");

f:

cout << "Введите логин: ";

cin >> input\_log;

if (input\_log == admin)

{

adminpass();

while (1)

{

b:

cout << "1-Ввод" << endl;

cout << "2-Справка" << endl;

cout << "3-Решение" << endl;

cout << "4-Выход" << endl;

cin >> a;

if (check(a)) {

switch (atoi(a))

{

case 1:

{

system("cls");

cout << "Введите количество потребителей(целое число от 1 до 50): ";

cin >> c;

if (check(c)) {

if (atoi(c) > 50 || atoi(c) <= 0)

{

system("cls");

cout << "Введите целое число от 1 до 50"<< endl;

goto b;

}

cout << "Введите количество товаров, необходимых каждому потребителю(целое число от 1 до 1000): " << endl;

for (int i = 0; i < atoi(c); i++)

{

cin >> e;

if (atoi(e) > 1000 || atoi(e) <= 0)

{

system("cls");

cout << "Введите целое число от 1 до 1000" << endl;

goto b;

}

if (check(e))

{

pok[i] = atoi(e);

}

else

{

def();

system("cls");

goto b;

}

}

cout << "Введите количество поставщиков(целое число от 1 до 50): ";

cin >> d;

if (check(d)) {

if (atoi(d) > 50 || atoi(d) <= 0)

{

system("cls");

cout << "Введите целое число от 1 до 50" << endl;

goto b;

}

cout << "Введите количество товаров, которые есть у поставщиков(целое число от 1 до 1000): " << endl;

for (int j = 0; j < atoi(d); j++)

{

cin >> e;

if (atoi(e) > 1000 || atoi(e) <= 0)

{

system("cls");

cout << "Введите целое число от 1 до 1000" << endl;

goto b;

}

if (check(e))

{

pos[j] = atoi(e);

}

else

{

def();

system("cls");

goto b;

}

}

for (int i = 0; i < atoi(c); i++)

{

sum1 = sum1 + pok[i];

}

for (int j = 0; j < atoi(d); j++)

{

sum2 = sum2 + pos[j];

}

if (sum1 != sum2)

{

system("cls");

cout << "Задача открытого типа, введите другие данные" << endl;

system("pause");

system("cls");

sum1 = 0;

sum2 = 0;

goto b;

}

cout << "Введите матрицу стоимостных показателей(целое число от 1 до 100): " << endl;

for (int i = 0; i < atoi(d); i++)

for (int j = 0; j < atoi(c); j++)

{

cin >> test;

if (check(test)) {

sp[i][j] = atoi(test);

if (sp[i][j] > 100 || sp[i][j] < 1)

{

system("cls");

cout << "Введите целое число от 1 до 100" << endl;

goto b;

}

}

else

{

system("cls");

cout << "Введите число"<<endl;

goto b;

}

}

system("pause");

}

else

{

def();

}

}

else

{

def();

}

break;

}

case 2:

{

z:

system("cls");

cout << "1-Об авторах" << endl;

cout << "2-О программе" << endl;

cout << "3-О данных" << endl;

cout << "4-О методах" << endl;

cout << "5-Назад" << endl;

cin >> a;

if (check(a))

{

while (1)

{

switch (atoi(a))

{

case 1:

{

system("cls");

cout << "Программа была выполнена студентами 3 курса РЭУ им. Плеханова, факультета ФМЭСИ," << endl;

cout << "специальности \"Информатика и вычислительная техника\", группы ДКТ-141б: " << endl;

cout << "Харченко Антон" << endl << "Серов Павел" << endl << "Хандожко Алексей" << endl;

cout << "Компания - \"KSK\"" << endl;

cout << "По всем интересующим вопросам можете обратиться на почту: mr.kharchenkoav@gmail.com" << endl;

cout << "Нажмите любую клавишу, чтобы продолжить" << endl;

system("pause");

goto z;

}

case 2:

{

system("cls");

cout << "Данная программа предназначена для решения транспортных задач закрытого типа одним из трёх методов." << endl;

cout << "После введения каждого числа необходимо нажать Enter." << endl;

cout << "Необходимо вводить целые числа. При введении символа, в том числе \".\" и \",\", программа зафиксирует введение символа." << endl;

cout << "Просмотреть введённую информацию Вы сможете в виде матрицы." << endl;

cout << "Стоимостные показатели перевозок будут вынесены в ячейки в левом верхнем углу каждой клетки матрицы." << endl;

cout << "Количество продукции у поставщиков Вы сможете найти в крайнем левом столбце." << endl;

cout << "Количество необходимой потребителям продукции - в верхней строке." << endl;

cout << "После решения задачи одним из методов Вы получите похожую матрицу." << endl;

cout << "В основной части ячеек матрицы будут записаны значения перевозимых продуктов от i - го поставщика j - ому покупателю." << endl;

cout << "Полученные планы не будут являться оптимальными." << endl;

cout << "Нажмите любую клавишу, чтобы продолжить" << endl;

system("pause");

goto z;

}

case 3:

{

system("cls");

cout << "Максимальное количество поставщиков - 50." << endl;

cout << "Максимальное количество потребителей - 50." << endl;

cout << "Максимальное количество продукции у поставщика - 1000." << endl;

cout << "Максимальное количество продукции необходимой потребителю - 1000." << endl;

cout << "Максимальная стоимость перевозки - 100." << endl;

cout << "Нажмите любую клавишу, чтобы продолжить" << endl;

system("pause");

goto z;

}

case 4:

{

system("cls");

cout << "Метод северо - западного угла." << endl;

cout << "При этом способе \"поставки\" располагаются, начиная от левого верхнего и кончая нижним правым углом матрицы." << endl;

cout << "На географических картах левый верхний угол сооветствует северо - западу, эта аналогия и дала название способу." << endl;

cout << "1. Первая ячейка — с которой начинается распределение — будет \"северо-западная\" ячейка в левом верхнем углу таблицы (1-й поставщик, 1-й потребитель)." << endl;

cout << "Вписываем в эту ячейку максимальный объем, который позволяет количество продаж у поставщика и спрос потребителя." << endl;

cout << "2. Переходим в следующую \"северо-западную\" ячейку, не считая уже распределенной области." << endl;

cout << "Вписываем в эту ячейку максимальный объем, который позволяет количество продаж у поставщика и спрос потребителя." << endl;

cout << "Соответственно, уменьшаем оставшиеся не распределенными объемы поставки и потребления в строке и столбце." << endl;

cout << "Продолжаем действовать таким же образом, до исчерпания всего запаса и спроса." << endl;

cout << "Нажмите любую клавишу, чтобы продолжить" << endl;

cout << endl << "Метод минимального элемента в строке." << endl;

cout << "Суть этого метода в том, что в транспортной таблице сначала заполняются ячейки с наименьшими стоимостями поставок, а потом уже ячейки с большими стоимостями поставок." << endl;

cout << "1. Ищем ячейку в исходной таблице с наименьшей стоимостью (если окажется, что есть несколько ячеек с одинаковыми и минимальными стоимостями – выбираем любую из них)." << endl;

cout << "2. В найденную ячейку выписываем объем поставки (Xij), который позволяет количество продаж у поставщика и спрос у потребителя." << endl;

cout << "3. Продажа у поставщика и спрос у потребителя уменьшается на объем поставки. Если продажа у поставщика стала равна нулю, то полностью вычеркиваем эту строку таблицы. Если спрос у потребителя равен нулю, то полностью вычеркиваем этот столбец таблицы." << endl;

cout << "4. Продолжаем этот процесс до тех пор, пока все продажи у поставщиков и спрос у потребителей не будет равен нулю." << endl;

cout << endl << endl;

cout << "Метод Лебедева" << endl;

cout << "1. Имеется матрица с спросом потребителей(Bi), с количеством продаж у поставщиков(Ai) и со стоимостью поставок(сij), так же имеются ограничения." << endl;

cout << "2.Для начала считаем сумму стоимостных показателей по столбцам и сумму стоимостных показателей по строкам." << endl;

cout << "3.Затем считаем среднее по столбцам, для этого сумму стоимостных показателей по столбцу делим на количество стоимостных показателей в этом столбце. Так считаем среднее по строкам." << endl;

cout << "4.Теперь для каждого стоимостного показателя складываем его среднее по столбцу и среднее по строке и из этой суммы вычитаем стоимостной показатель." << endl;

cout << "Записываем в клетку стоимостного показателя новое значение, которое будем называть коэффициентом очередности.Проделываем эти действия с каждым стоимостным показателем." << endl;

cout << "Среднее по столбцу + среднее по строке - xij = коэффициент очередности." << endl;

cout << "5.Разделение поставок мы начинаем с клетки с наибольшим коэффициентом очередности, далее идет следующий за ним по величине." << endl;

system("pause");

goto z;

}

case 5:

{

system("cls");

goto b;

}

default:

{

def();

goto z;

}

}

break;

}

}

else

{

def();

goto z;

}

}

case 3:

{

system("cls");

cout << "1-Метод Северо-Западного угла" << endl;

cout << "2-Метод минимального элемента в строке" << endl;

cout << "3-Методом Лебедева" << endl;

cout << "4-Назад" << endl;

cin >> x;

if (check(x)) {

switch (atoi(x))

{

case 1: {

t1 = clock();

int \*pos\_nw = new int[atoi(d)];

int \*pok\_nw = new int[atoi(c)];

for (int i = 0; i < atoi(d); i++)

for (int j = 0; j < atoi(c); j++)

shipment[i][j] = 0;

for (int i = 0; i < atoi(d); i++) {

pos\_nw[i] = pos[i];

}

for (int i = 0; i < atoi(c); i++) {

pok\_nw[i] = pok[i];

}

for (int i = 0; i < atoi(d); i++)

{

for (int k = 0; k < atoi(c); k++)

{

if (pok\_nw[k] == 0) continue;

min\_val = min(pos\_nw[i], pok\_nw[k]);

shipment[i][k] = min\_val;

pos\_nw[i] -= min\_val;

pok\_nw[k] -= min\_val;

if (pos[i] == 0) break;

}

}

vivod(c, d, shipment);

ksk = (atoi(d))\*(atoi(c)) - 1;

s = 0;

cout << "Pl= ";

for (int i = 0; i < atoi(d); i++)

for (int j = 0; j < atoi(c); j++)

{

if (ksk--)

{

s = s + ((shipment[i][j]) \* (sp[i][j]));

cout << (shipment[i][j]) << " \* " << (sp[i][j]) << " + ";

}

}

s = s + (shipment[atoi(d) - 1][atoi(c) - 1]) \* (sp[atoi(d) - 1][atoi(c) - 1]);

cout << (shipment[atoi(d) - 1][atoi(c) - 1]) << " \* " << (sp[atoi(d) - 1][atoi(c) - 1]);

cout << " = " << s<<endl;

t2 = clock();

cout << "Время, потраченное на решение метода: " << ((double)(t2 - t1) / CLOCKS\_PER\_SEC) << "с." << endl;

system("pause");

break; }

case 2: {

t1 = clock();

for (int i = 0; i < atoi(d); i++)

{

posm[i] = pos[i];

}

for (int i = 0; i < atoi(c); i++)

{

pokm[i] = pok[i];

}

for (int i = 0; i < atoi(d); i++)

{

for (int j = 0; j < atoi(c); j++)

{

res[i][j] = 1001;

}

}

for (u = 0; u < atoi(d); u++)

{

g = u;

minim = 1001;

y = 0;

for (int r = 0; r < atoi(c); r++)

{

if (minim > sp[u][r])

{

minim = sp[u][r];

y = r;

}

}

while (posm[g] != 0)

{

if (posm[g] <= pokm[y])

{

res[g][y] = posm[g];

pokm[y] = pokm[y] - posm[g];

posm[g] = posm[g] - posm[g];

for (p = 0; p < atoi(c); p++)

if (res[g][p] == 1001)

res[g][p] = 0;

}

else

{

res[g][y] = pokm[y];

posm[g] = posm[g] - pokm[y];

pokm[y] = pokm[y] - pokm[y];

{

for (o = g; o < atoi(d); o++)

if (res[o][y] == 1001)

res[o][y] = 0;

}

}

k = sp[u][y];

minim = 101;

for (l = 0; l < atoi(c); l++)

{

if ((minim > sp[g][l]) && (res[g][l] == 1001))

{

minim = sp[g][l];

y = l;

}

}

}

}

vivod(c, d, res);

int ksk;

ksk = (atoi(d))\*(atoi(c)) - 1;

s = 0;

cout << "Pl= ";

for (int i = 0; i < atoi(d); i++)

for (int j = 0; j < atoi(c); j++)

{

if (ksk--)

{

s = s + ((res[i][j]) \* (sp[i][j]));

cout << (res[i][j]) << " \* " << (sp[i][j]) << " + ";

}

}

s = s + (res[atoi(d) - 1][atoi(c) - 1]) \* (sp[atoi(d) - 1][atoi(c) - 1]);

cout << (res[atoi(d) - 1][atoi(c) - 1]) << " \* " << (sp[atoi(d) - 1][atoi(c) - 1]);

cout << " = " << s<<endl;

t2 = clock();

cout << "Время, потраченное на решение метода: " << ((double)(t2 - t1) / CLOCKS\_PER\_SEC) << "с." << endl;

system("pause");

break; }

case 3: {

t1 = clock();

f = 0;

for (int i = 0; i<atoi(d); i++)

{

possr[i] = 0;

for (int j = 0; j<atoi(c); j++)

{

possr[i] = possr[i] + sp[i][j];

}

possr[i] = (float)possr[i] / atoi(c);

}

for (int j = 0; j<atoi(c); j++)

{

poksr[j] = 0;

for (int i = 0; i<atoi(d); i++)

{

poksr[j] = poksr[j] + sp[i][j];

}

poksr[j] = (float)poksr[j] / atoi(d);

}

for (int i = 0; i<atoi(d); i++)

{

for (int j = 0; j<atoi(c); j++)

{

sr[i][j] = (float)possr[i] + poksr[j] - sp[i][j];

}

}

for (int i = 0; i<atoi(d); i++)

posl[i] = pos[i];

for (int i = 0; i<atoi(c); i++)

pokl[i] = pok[i];

for (int i = 0; i<atoi(d); i++)

{

for (int j = 0; j<atoi(c); j++)

{

res[i][j] = 0;

}

}

while (f != 1)

{

float min;

min = (float)sr[0][0];

for (int i = 0; i<atoi(d); i++)

{

for (int j = 0; j<atoi(c); j++)

{

if (min>sr[i][j])

{

min = sr[i][j];

mini = i;

minj = j;

}

}

}

sr[mini][minj] = 1001;

if (posl[mini]>pokl[minj])

v = pokl[minj];

else

v = posl[mini];

posl[mini] = posl[mini] - v;

pokl[minj] = pokl[minj] - v;

res[mini][minj] = v;

if (posl[mini] == 0)

{

for (int j = 0; j<atoi(c); j++)

sr[mini][j] = 1001;

}

if (pokl[minj] == 0)

{

for (int i = 0; i<atoi(d); i++)

sr[i][minj] = 1001;

}

h = 0;

for (int i = 0; i<atoi(d); i++)

h = h + posl[i];

for (int j = 0; j<atoi(c); j++)

h = h + pokl[j];

if (h == 0)

f = 1;

}

vivod(c, d, res);

ksk = (atoi(d))\*(atoi(c)) - 1;

s = 0;

cout << "Pl= ";

for (int i = 0; i < atoi(d); i++)

for (int j = 0; j < atoi(c); j++)

{

if (ksk--)

{

s = s + ((res[i][j]) \* (sp[i][j]));

cout << (res[i][j]) << " \* " << (sp[i][j]) << " + ";

}

}

s = s + (res[atoi(d) - 1][atoi(c) - 1]) \* (sp[atoi(d) - 1][atoi(c) - 1]);

cout << (res[atoi(d) - 1][atoi(c) - 1]) << " \* " << (sp[atoi(d) - 1][atoi(c) - 1]);

cout << " = " << s<<endl;

t2 = clock();

cout << "Время, потраченное на решение метода: " << ((double)(t2 - t1) / CLOCKS\_PER\_SEC) << "с."<<endl;

system("pause");

break; }

case 4: {

break; }

}

break;

}

}

case 4:

{

char ex[100];

system("cls");

cout << "Вы хотите выйти?" << endl;

cout << "1-Да" << endl;

cin >> ex;

if (check(ex))

{

switch (atoi(ex))

{

case 1:

{

return 0;

break;

}

default:

{

def();

}

}

}

}

}

system("cls");

}

else

{

def();

system("cls");

}

}

}

else

{

system("cls");

cout << "Неверный логин." << endl;

goto f;

}

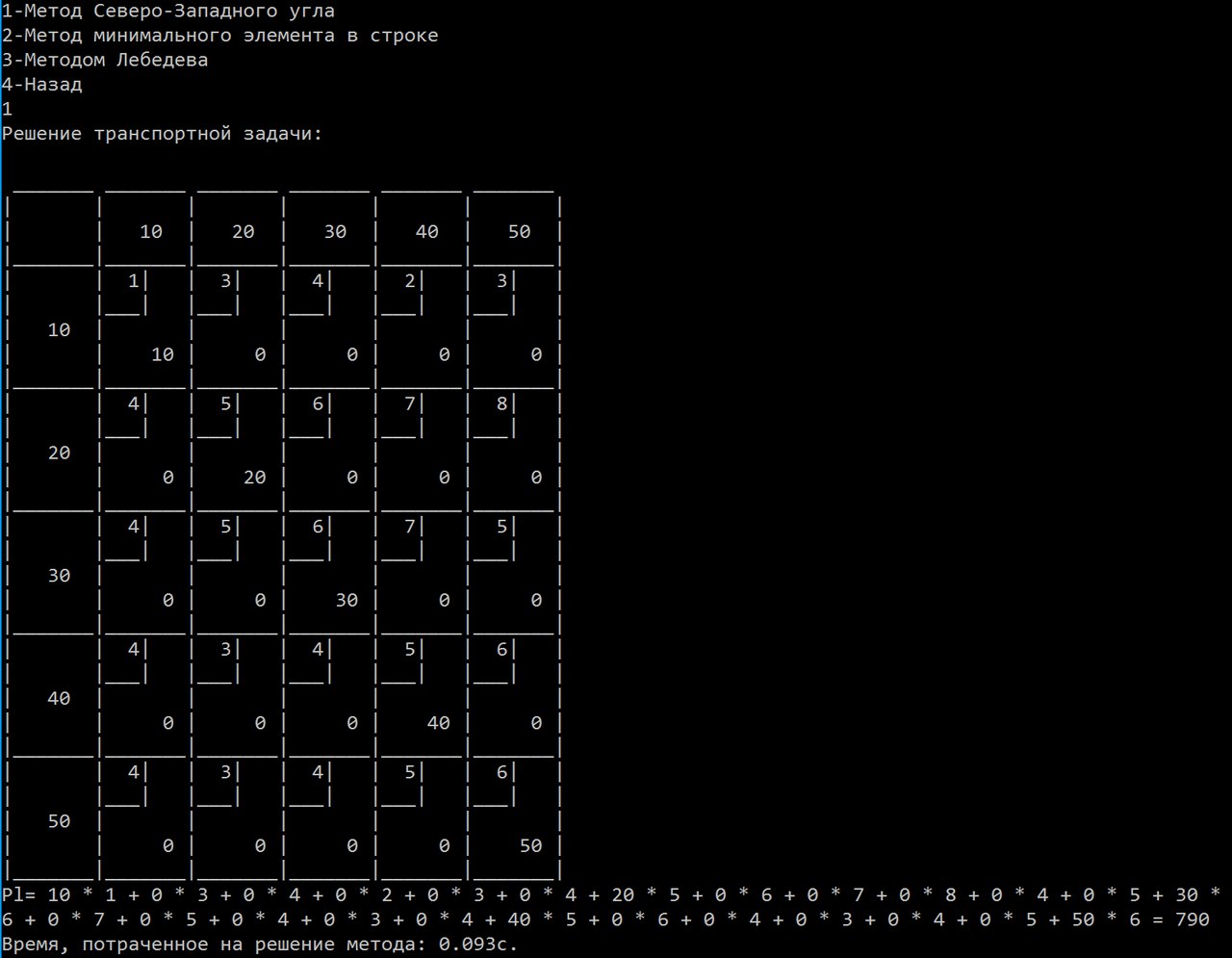
system ("pause");

return 0;

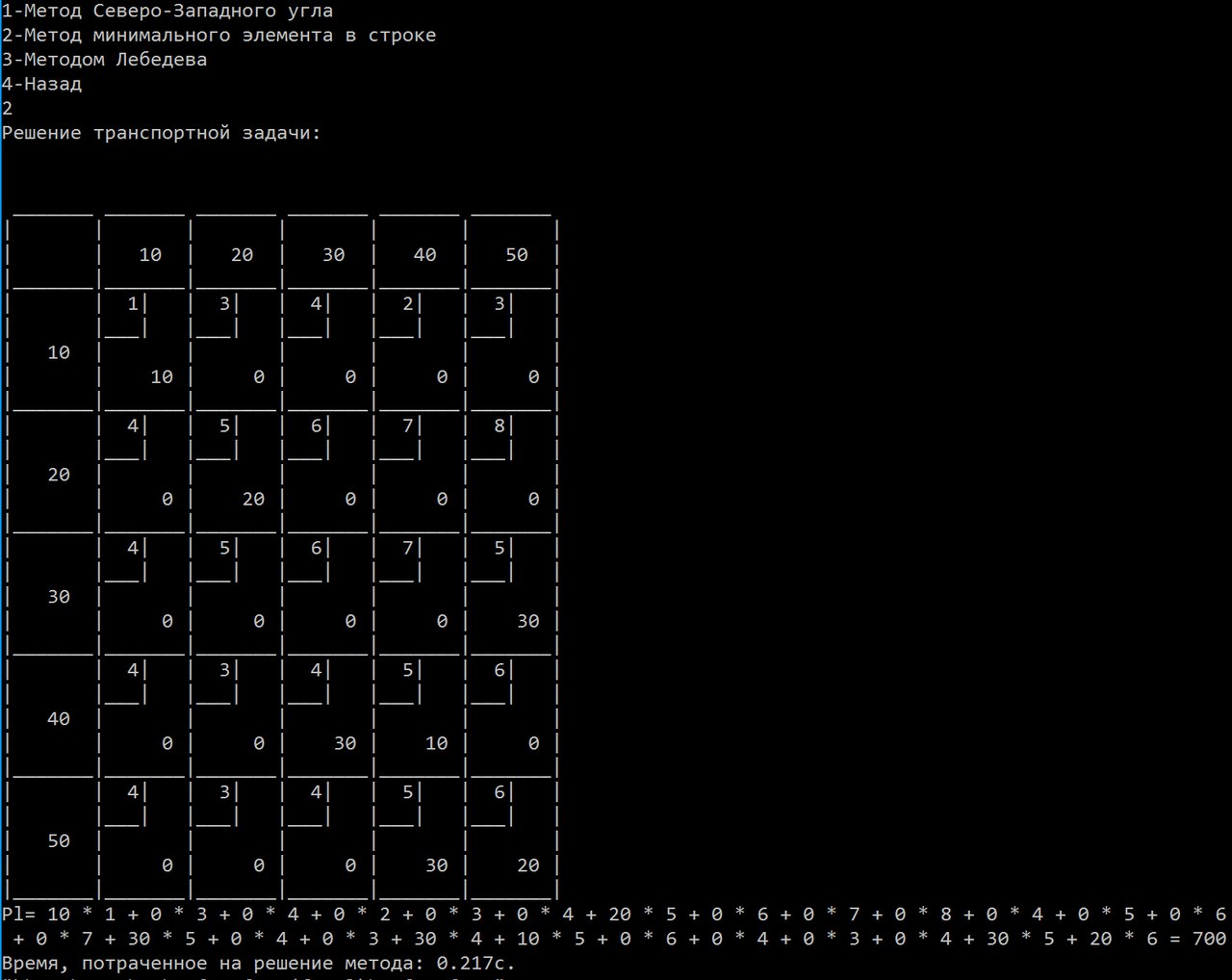
}

**7.2 Результаты тестирования и выполнения программы.**

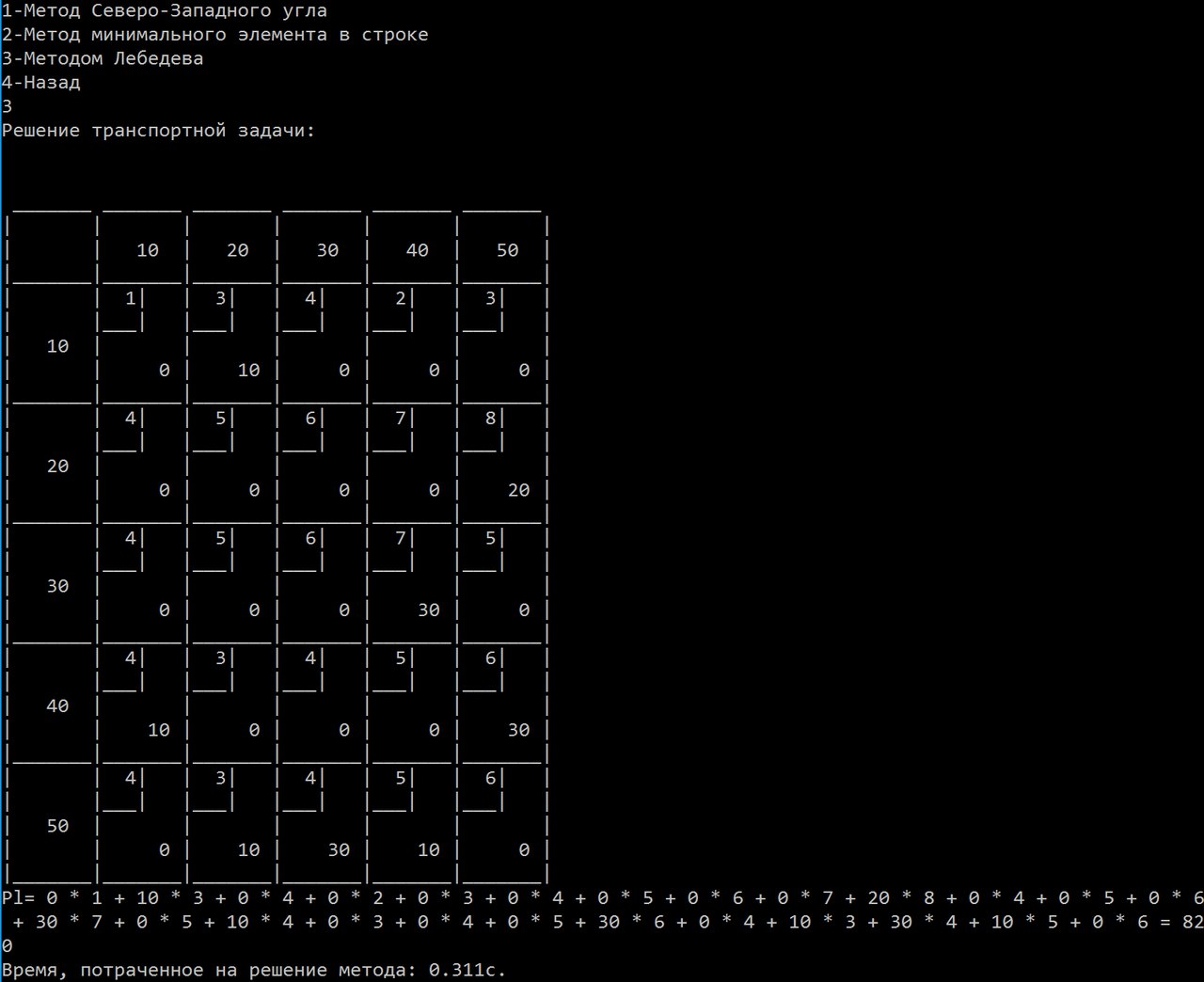
7.2.1



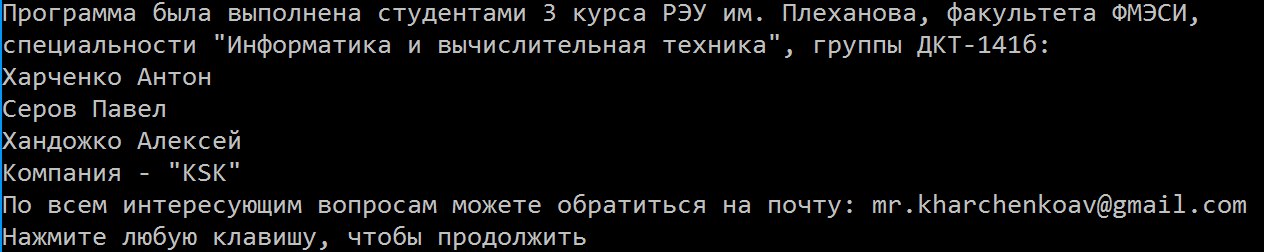
7.2.2

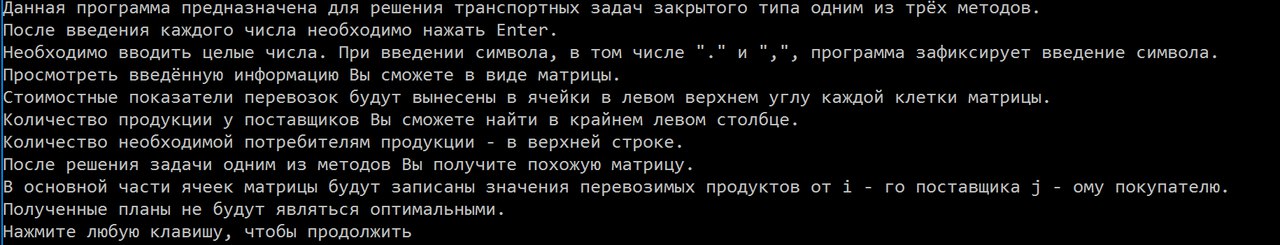


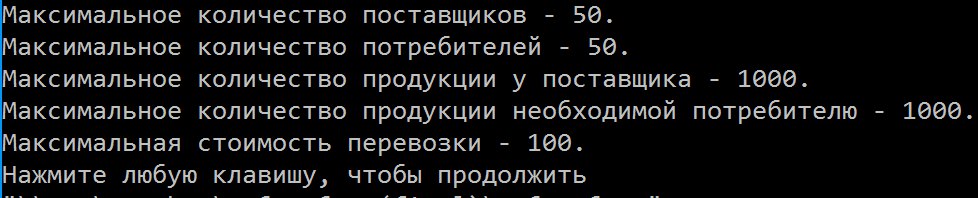
7.2.3

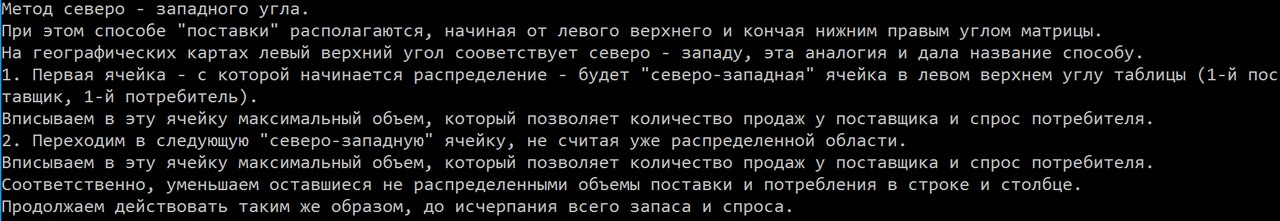


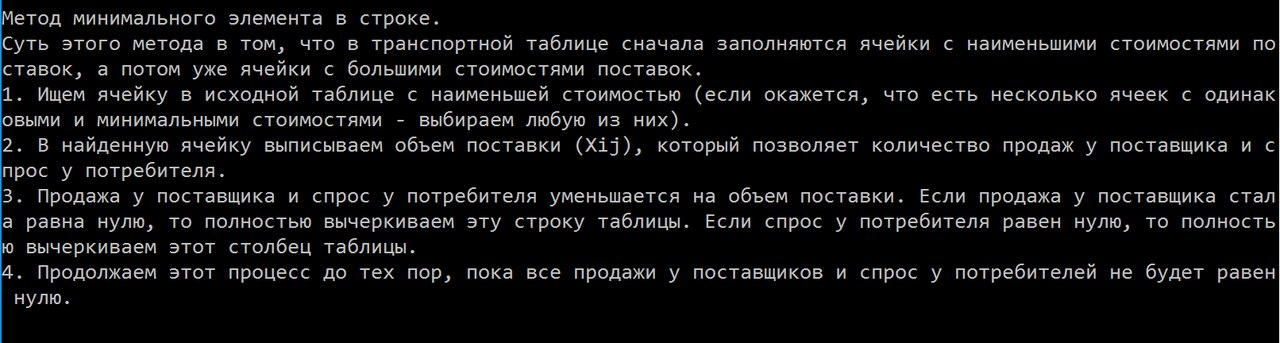
7.2.4

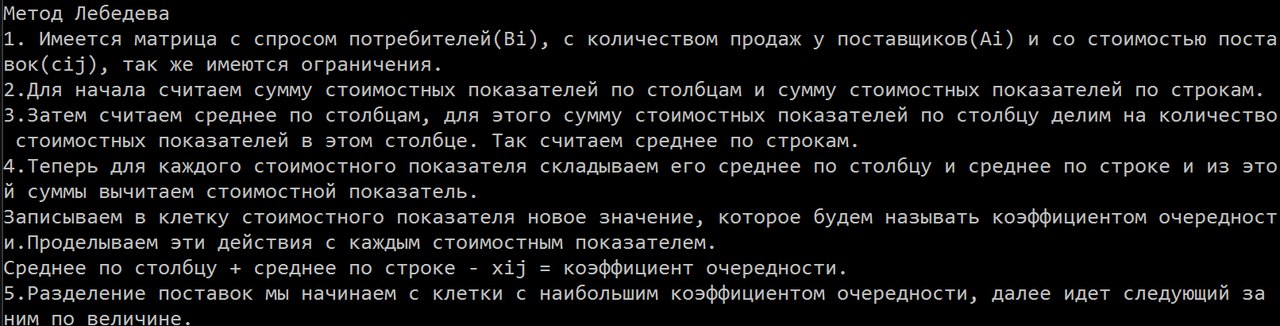




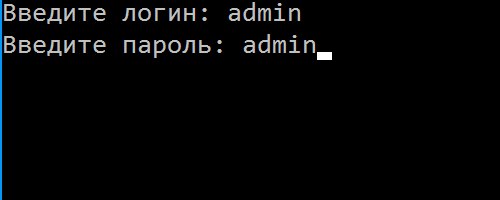


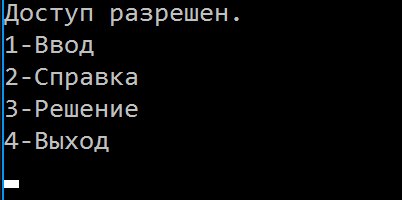




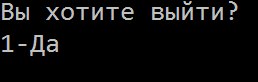


7.2.6

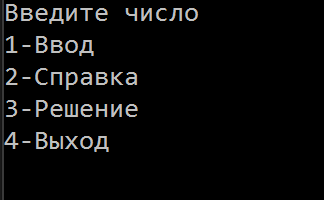




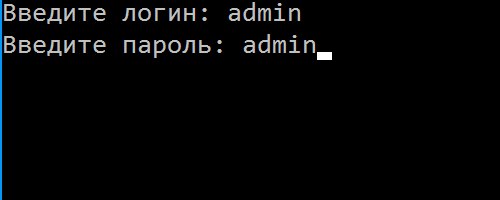
7.2.7

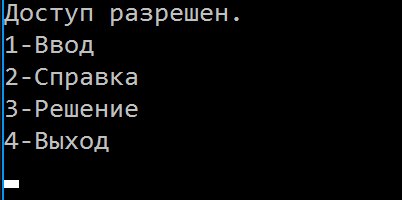


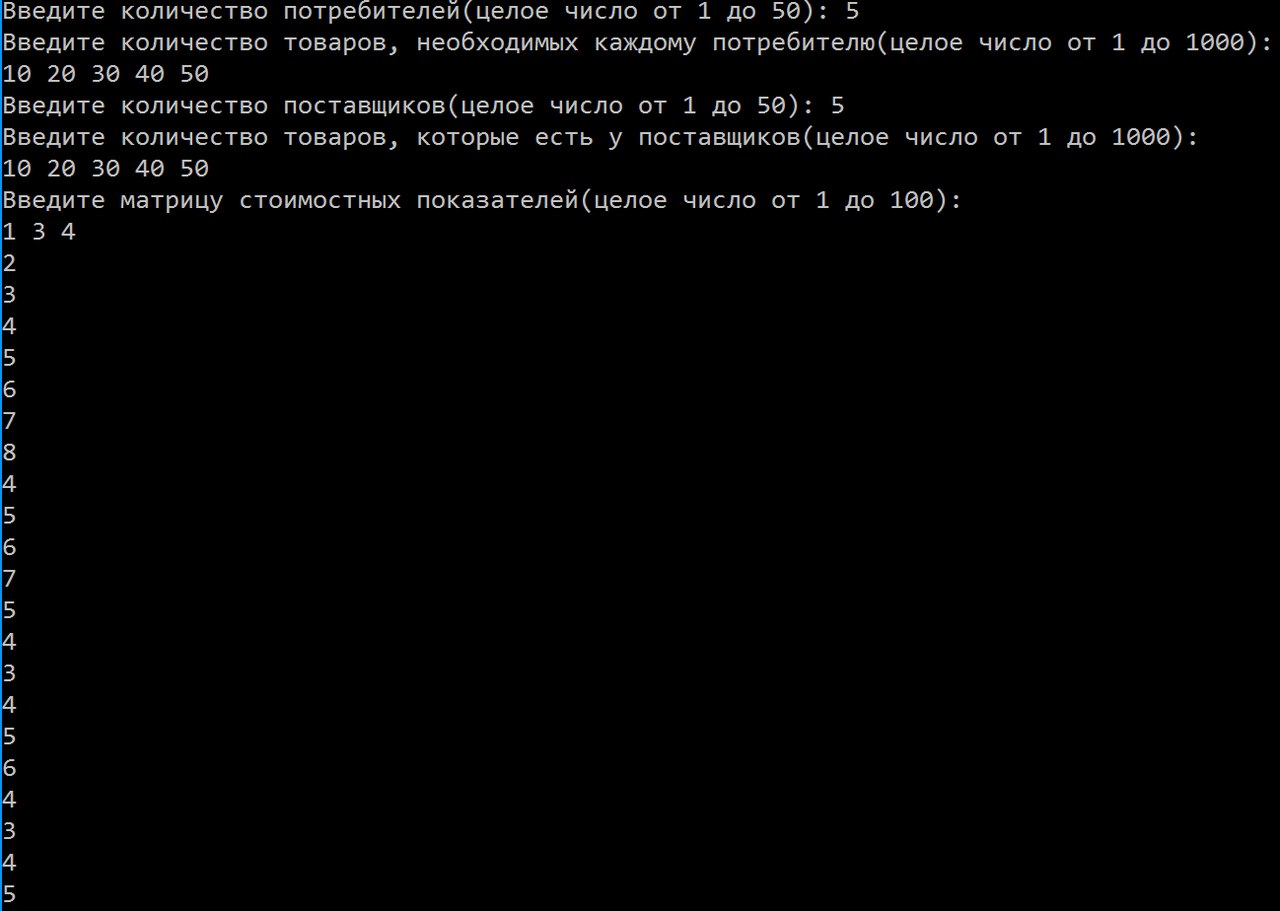
7.2.8



**7.3 Экранные формы.**

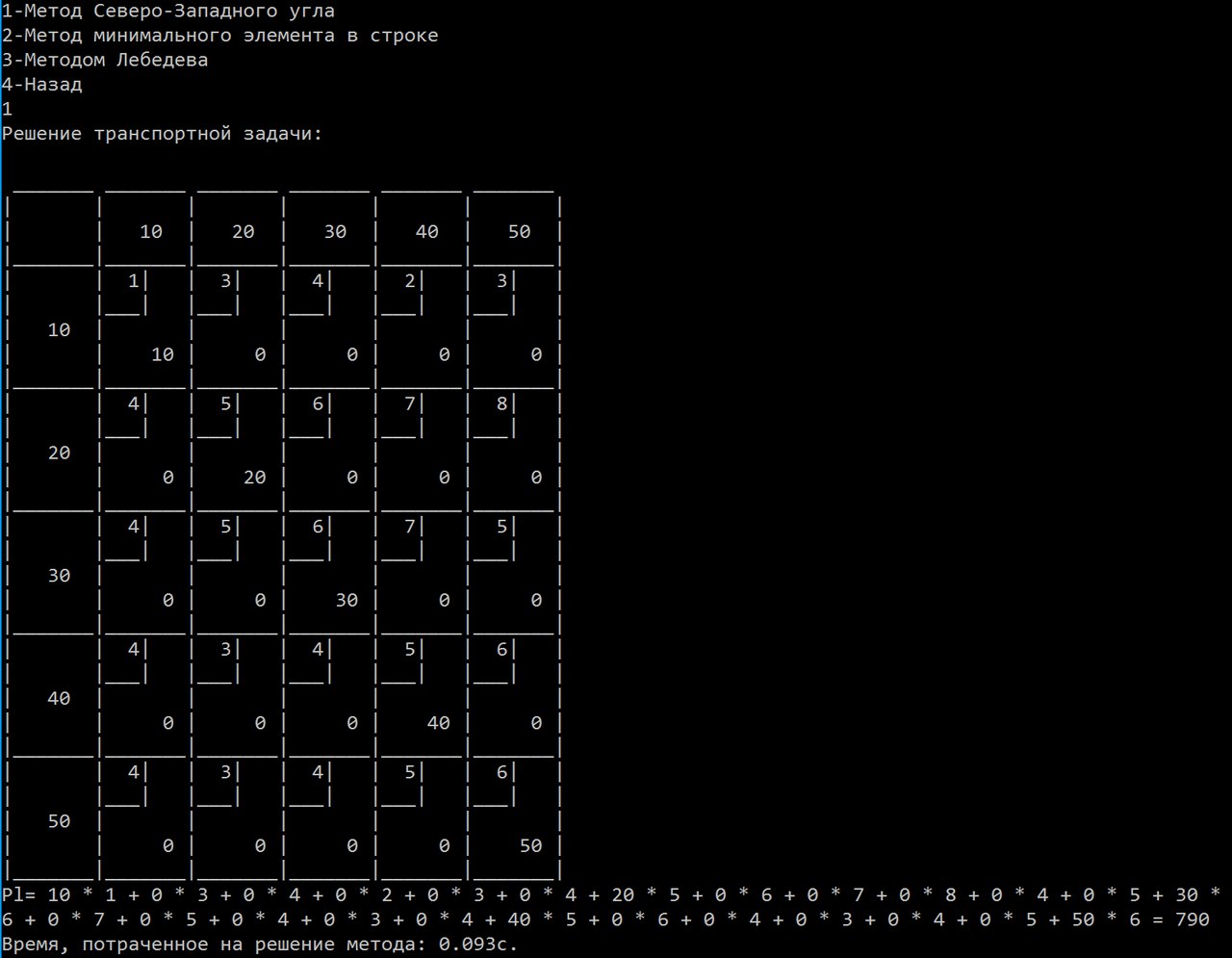




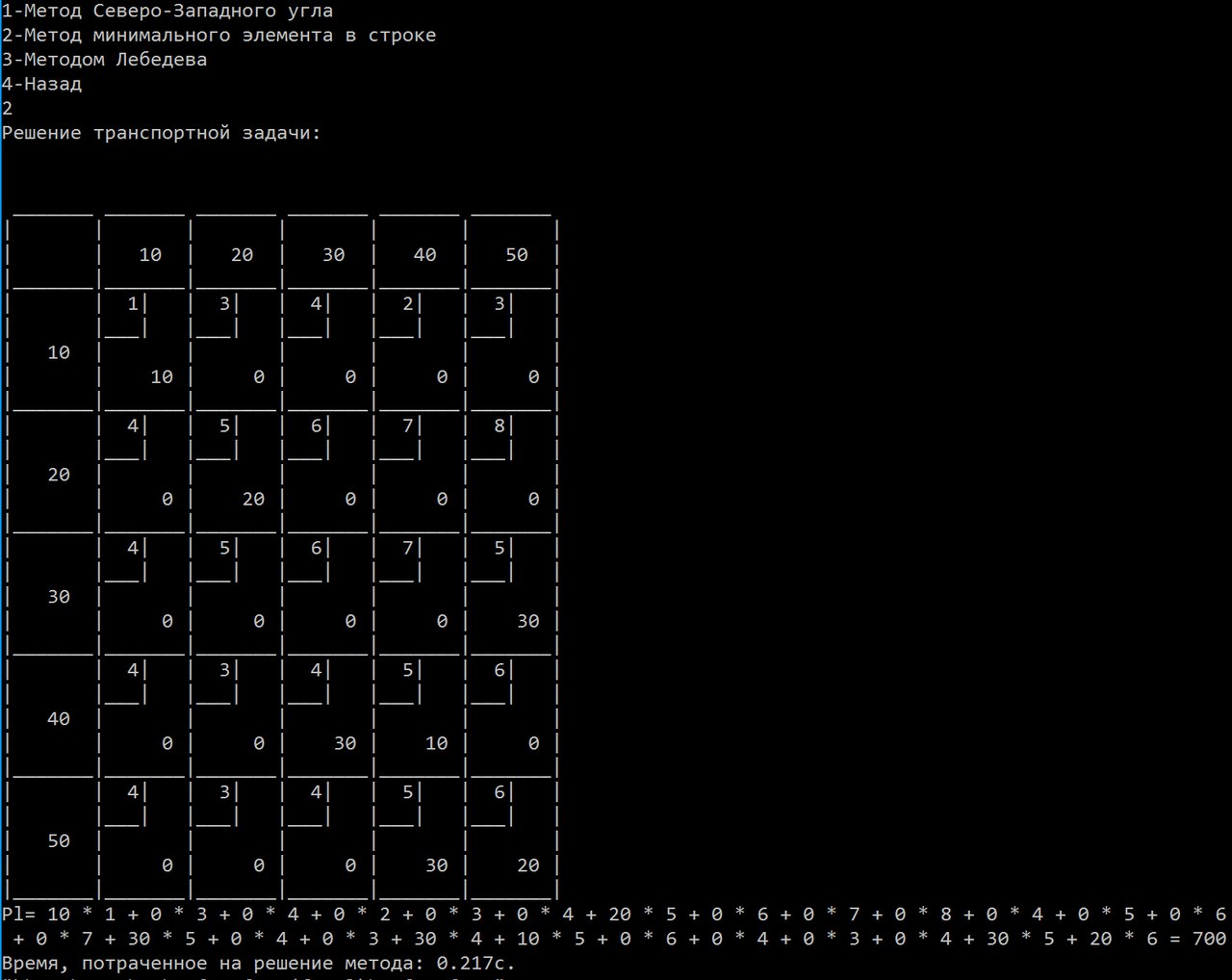


Тестирование на заданных наборах данных:

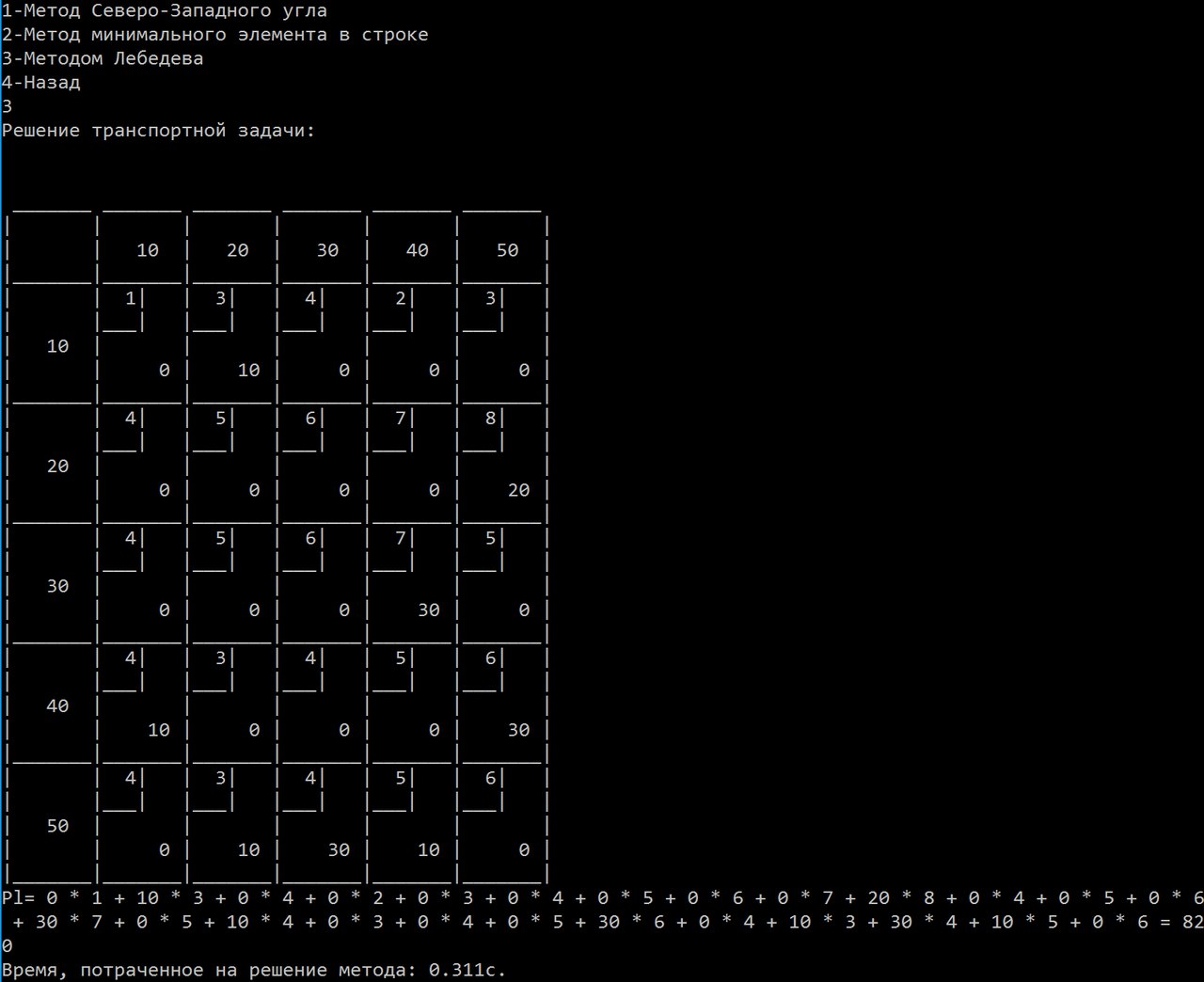
Метод северо-западного угла:



Метод минимального элемента в строке:



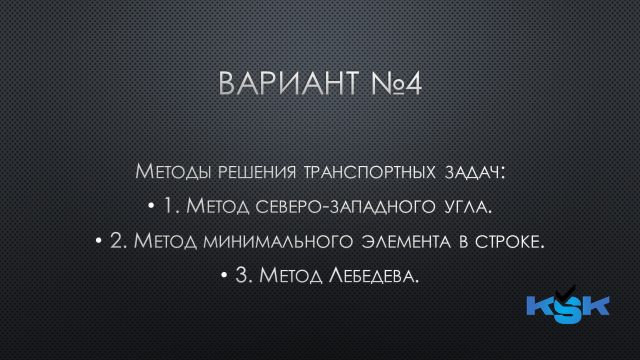
Метод Лебедева:



**7.4 Заставка и реклама.**



****

****

****