**ФГОУ СПО «Волгоградский государственный экономико-технический колледж»**

**Кафедра информационных технологий**

**КУРСОВОЙ ПРОЕКТ**

по дисциплине: Математические методы

на тему:

**Построение оптимальной последовательности операций в коммерческой деятельности**

**Пояснительная записка**

ВГЭТК. 401-П. 13. КП.13. ПЗ

Студента: Кролькова Ильи Владимировича

Шифр К-1205

Группа 401-П

Руководитель: Еловенко Н. А.

Волгоград, 2011

**Содержание**

[Введение 3](#_Toc286776230)

[1 Теоретическая часть 3](#_Toc286776231)

[1.1 Принцип оптимальности и математическое описание динамического процесса управления 3](#_Toc286776232)

[1.2 Построение сетевой модели последовательности операций в коммерческой деятельности и ее решение 3](#_Toc286776233)

[2 Практическая часть 3](#_Toc286776234)

[2.1 Решение задачи с помощью математического аппарата 3](#_Toc286776235)

[2.2 Решение задачи средствами прикладных программ 3](#_Toc286776236)

[2.3 Автоматизация решения задачи 3](#_Toc286776237)

[Заключение 3](#_Toc286776238)

[Библиографический список 3](#_Toc286776239)

[Приложение А Блок-схема алгоритма 3](#_Toc286776240)

[Приложение Б Листинг расчета в VBA 3](#_Toc286776241)

# Введение

Коммерческая деятельность в том или ином виде сводится к решению таких задач: как распорядиться имеющимися ресурсами для достижения наибольшей выгоды или какое следует предпринять действие для получения возможно лучшего финансового результата. Однако стало возможным часть этого искусства сделать наукой, базирующейся на математических методах.

Необходимость использования таких методов диктуется тем, что последствия принимаемых решений могут касаться большого числа людей и быть связаны с огромными затратами. Поэтому степень ответственности за принимаемые решения значительно возрастает. Перевод реального мира коммерческой деятельности на язык математики позволит получить наиболее точное представление о его существенных свойствах и предсказать будущие события.

Целью работы является создание программы позволяющей построить оптимальную последовательность операций в коммерческой деятельности.

Для достижения поставленной цели необходимо решить следующие задачи:

* изучить предметную область решаемой проблемы;
* разработать этапы решения задачи;
* рассмотреть принципы использования прикладных программ для расчета основных характеристик по теме задачи;
* разработать приложение, позволяющее автоматизировать процесс решения задачи.

Объект исследования – построение сетевой модели последовательности операций .

Предмет исследования – построение оптимальной последовательности операций в коммерческой деятельности.

# 1 Теоретическая часть

## 1.1 Принцип оптимальности и математическое описание динамического процесса управления

В основе метода ДП лежит принцип оптимальности, впервые сформулированный в 1953 г. американским математиком Р.Э. Беллманом: каково бы ни было состояние системы в результате какого-либо числа шагов, на ближайшем шаге нужно выбирать управление так, чтобы оно в совокупности с оптимальным управлением на всех последующих шагах приводило к оптимальному выигрышу на всех оставшихся шагах, включая выигрыш на данном шаге. При решении задачи на каждом шаге выбирается управление, которое приводит к оптимальному выигрышу. Если считать все шаги независимыми, тогда оптимальным управлением будет то управление, которое обеспечит максимальный выигрыш на каждом шаге. Однако, например, при покупке новой техники взамен устаревшей на ее приобретение затрачиваются определенные средства, поэтому доход от ее эксплуатации в начале может быть небольшой, а в следующие годы новая техника будет приносить больший доход. И наоборот, если принято решение оставить старую технику для получения дохода в текущем году, то в дальнейшем это приведет к значительным убыткам. Этот пример демонстрирует следующий факт: в многошаговых процессах управление на каждом конкретном шаге надо выбирать с учетом его будущих воздействий на весь процесс.

Кроме того, при выборе управления на данном шаге следует учитывать возможные варианты состояния предыдущего шага. Например, при определении количества средств, вкладываемых в предприятие в i-м году, необходимо знать, сколько средств осталось в наличии к этому году и какой доход получен в предыдущем -м году. Таким образом, при выборе шагового управления необходимо учитывать следующие требования:

1. возможные исходы предыдущего шага;

1. влияние управления на все оставшиеся до конца процесса шаги *(п — к).*

В задачах динамического программирования первое требование учитывают, делая на каждом шаге условные предположения о возможных вариантах окончания предьщущего шага и проводя для каждого из вариантов условную оптимизацию. Выполнение второго требования обеспечивается проведением безусловной оптимизации в обратном порядке.

Условная оптимизация. На первом этапе решения задачи, называемом условной оптимизацией, определяются функция Беллмана и оптимальные управления для всех возможных состояний на каждом шаге, начиная с последнего в соответствии с алгоритмом обратной прогонки. На последнем, n-м, шаге оптимальное управление определяется функцией Беллмана: в соответствии с которой максимум выбирается из всех возможных значений причем.

Дальнейшие вычисления проводятся согласно рекуррентному соотношению, связывающему функцию Беллмана на каждом шаге с этой же функцией, но вычисленной на предыдущем шаге. В общем виде это соотношение имеет вид

Этот максимум (или минимум) определяется по всем возможным для *k и S* значениям переменной управления*х*

Безусловная оптимизация. После того как функция Беллмана и соответствующие оптимальные управления найдены для всех шагов с n-го по первый, осуществляется второй этап решения задачи, называемый безусловной оптимизацией, проводимой в обратном порядке.

Пользуясь тем, что на первом шаге (k = 1) состояние системы известно - это ее начальное состояние можно найти оптимальный результат за все *п* шагов и оптимальное управление на первом шаге , которое этот результат доставляет. После применения этого управления система перейдет в другое состояние зная которое, можно, пользуясь результатами условной оптимизации, найти оптимальное управление на втором шаге и так далее до последнего *п-го* шага.

Вычислительную схему динамического программирования можно строить на сетевых моделях, а также по алгоритмам прямой прогонки (от начала) и обратной прогонки (от конца к началу). Рассмотрим примеры решения различных по своей природе задач, содержание которых требует выбора переменных состояния и управления.

## 1.2 Построение сетевой модели последовательности операций в коммерческой деятельности и ее решение

Пусть на оптовую базу прибыло *п* машин с товаром для разгрузки и *т* машин для загрузки товаров, направляемых в магазины. Материально ответственное лицо оптовой базы осуществляет оформление документов по операциям разгрузки или загрузки для одной машины, а затем переходит к обслуживанию другой машины. Издержки от операций обусловлены простоем транс порта, типом операции (прием или отправка товара) и не зависят от конкретной машины. Необходимо спланировать последовательность операций обоих видов таким образом, чтобы, суммарные издержки по приему и отправке товаров для всех машин были минимальными.

Из условия следует, что состояние экономической системы характеризуется двумя параметрами: количеством принятых и оформленных машин по разгрузке товара и количеством машин отправленных с товаром в магазины. Поэтому решение будем искать на плоскости *XOY*, ограниченной прямоугольником, который является областью допустимых состояний системы. Если по оси X отложить число *п* разгруженных машин, а по оси *Y—* число *т* загруженных товаром машин, то можно построить на плоскости граф состояний процесса, в котором каждая вершина характеризует состояние операции приема и отгрузки товара на оптовой базе. Ребра этого графа означают выполнение работы по приему или отправке товара на очередной машине. Каждому ребру можно сопоставить издержки, связанные с выполнением операции по разгрузке или загрузке машины.

**Пример.** Пусть n = 6, m = 4. Известны затраты по выполнению каждой операции, которые показаны на ребрах графа (рис. 5.7.1).

Точка определяет начало процесса, а — конечное состояние, соответствующее приему и отправке всех машин. Оптимизацию процесса будем производить с конечного состояния —. Весь процесс разобьем на шаги, их количество Каждый шаг представляет собой сечение графа состояний, проходящее через вершины (на рис. 1.2.1 сечения показаны косыми линиями).

*Рис 1.2.1 Графическая схема связи операций*

**I этап. Условная оптимизация**

1-й шаг: k = 1. На первом шаге с задаваемым сечением*,* из состояний *и* возможен только один вариант перехода вконечное состояние . Поэтому в вершинах и записываемсоответственно издержки 8 и 11. Ребра и обозначаемстрелкой, направленной в вершину -, как показано на рис. 1.2.2.

*Рис 1.2.2 Фрагмент связи операции (шаг 1)*

2-й шаг: *k = 2.* Второй шаг оптимизации задается сечением по вершинам. Из состояний и , возможен единственный переход в вершины *,* и соответственно, поэтому в вершинах и записываем суммарные издержки 17 и 22 на первых двух шагах перехода в конечное состояние .

Из вершины возможны два варианта перехода: в вершинy или вершину *.* При переходе сумма издержек составляет 10 + 8 = 18, на переходе сумма составляет 13 + 11 = 24. Из двух вариантов суммарных издержек выбираем наименьшую (18) и обозначаем стрелкой условно оптимальный переход*,* как показано на рис. 1.2.3.

*Рис 1.2.3 Сетевая модель операции (шаг 2)*

3-й шаг: *k* = 3. На третьем шаге сечение проходит через вершины , , ,. Из вершин и возможен единственный переход в вершины  соответственно. Суммарные издержки для состояния равны 22 + 12 = 34. Из вершины возможны два варианта перехода: в вершину издержки равны 17 + 8 = 25; в вершину 18 + 9 = 27.

Для вершины возможен переход в вершину (18 + 10 = 28) и в вершину (22 + 12 = 34). Выбираем для вершин и наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход, как показано на рис. 1.2.4.

*Рис 1.2.4 Сетевая модель операции (шаг 3)*

Продолжая процесс аналогичным образом для оставшихся шагов, приходим в точкуВ результате получим граф условно оптимальных переходов, представленный на рис. 1.2.5.

*Рис 1.2.5 Сетевая модель связи расходов операций*

**II этап. Безусловная оптимизация.**

Определяем оптимальную траекторию на исходном сетевом графе, просматривая результаты всех шагов в обратном порядке, учитывая, что выбор некоторого управления на *k*-м шаге приводит к тому, что состояние на (*к* — 1)-м шаге становится определенным.

В результате строим ориентированный граф перехода из состояния в состояние, представленный на рис. 1.2.6; на каждом шаге безусловной оптимизации переход почти всегда единственный и совпадает с построенными условно оптимальными переходами.

*Рис 1.2.6 Оптимальная последовательность операций*

# 2 Практическая часть

**Условие задачи:**

Определите оптимальную последовательность операций по приемке и отпуску товаров на предприятии оптовой торговли, позволяющую минимизировать суммарные издержки при условиях, приведенных в виде матрицы вариантов связей и затрат по каждой операции.

*Рис 2.1 Графическая схема связи операций*

## 2.1 Решение задачи с помощью математического аппарата

**I этап. Условная оптимизация**

1-й шаг: k = 1. На первом шаге с задаваемым сечением*,* из состояний *и* возможен только один вариант перехода вконечное состояние . Поэтому в вершинах и записываемсоответственно издержки 12 и 9. Ребра и обозначаемстрелкой, направленной в вершину -.

2-й шаг: *k = 2.* Второй шаг оптимизации задается сечением по вершинам. Из состояний и , возможен единственный переход в вершины *,* и соответственно, поэтому в вершинах и записываем суммарные издержки 25 и 19 на первых двух шагах перехода в конечное состояние .

Из вершины возможны два варианта перехода: в вершинy или вершину *.* При переходе сумма издержек составляет 12 + 10 = 22, на переходе сумма составляет 13 + 9 = 22. Из двух вариантов суммарных издержек выбираем наименьшую (22) и обозначаем стрелкой условно оптимальный переход,.

3-й шаг: *k* = 3. На третьем шаге сечение проходит через вершины , , ,. Из вершин и возможен единственный переход в вершины соответственно. Суммарные издержки для состояния равны 19 + 11 = 30, для состояния равны 25+11=36. Из вершины возможны два варианта перехода: в вершину издержки равны 25 + 11 = 36; в вершину 22 + 14 = 36.

Для вершины возможен переход в вершину (22 + 15 = 37) и в вершину (19 + 19 = 38). Выбираем для вершин и наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

4-й шаг: *k* = 4. На четвертом шаге сечение проходит через вершины , , ,, Из вершин и возможен единственный переход в вершины  соответственно. Суммарные издержки для состояния равны 30 + 19 = 49, для состояния равны 36+9=45. Из вершины возможны два варианта перехода: в вершину издержки равны 36 + 12 = 48; в вершину 36 + 15 = 51.

Для вершины возможен переход в вершину (36 + 13 = 49) и в вершину (37 + 18 = 55). Выбираем для вершин и наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

Для вершины возможен переход в вершину (30 + 18 = 48) и в вершину (37 + 14 = 51). Выбираем для вершины наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

5-й шаг: *k* = 5. На пятом шаге сечение проходит через вершины , , ,, Из вершины возможен единственный переход в вершину . Суммарные издержки для состояния равны 45 + 8 = 53. Из вершины возможны два варианта перехода: в вершину издержки равны 45+13 = 58; в вершину 48 + 14 = 62.

Для вершины возможен переход в вершину (48 + 14 = 62) и в вершину (49 + 21 = 70). Выбираем для вершин и наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

Для вершины возможен переход в вершину (48+ 13 = 61) и в вершину (49 + 12 = 61). Выбираем для вершины наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

Для вершины возможен переход в вершину (49 + 17 = 66) и в вершину (48 + 16 = 64). Выбираем для вершины наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

6-й шаг: *k* = 6. На шестом шаге сечение проходит через вершины , , ,, Из вершины возможен единственный переход в вершину . Суммарные издержки для состояния равны 53 + 10 = 63. Из вершины возможны два варианта перехода: в вершину издержки равны 53+14 = 67; в вершину 58 + 13 = 71.

Для вершины возможен переход в вершину (58 + 12 = 70) и в вершину (62 + 20 = 82). Выбираем для вершин и наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

Для вершины возможен переход в вершину (61+ 12 = 73) и в вершину (62 + 11 = 73). Выбираем для вершины наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

Для вершины возможен переход в вершину (64 + 16 = 80) и в вершину (61+ 13 = 74). Выбираем для вершины наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

7-й шаг: *k* = 7. На седьмом шаге сечение проходит через вершины , , ,. Из вершины  возможен переход в вершину (63 + 15 = 78) и в вершину (67 + 12 = 79).

Для вершины возможен переход в вершину (67 + 13 = 80) и в вершину (70 + 19 = 89).

Для вершины возможен переход в вершину (70 + 13 = 83) и в вершину (73 + 12 = 85).

Для вершины возможен переход в вершину (73 + 15 = 88) и в вершину (74 + 14 = 88).

Выбираем для вершин , , , наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

8-й шаг: *k* = 8. На восьмом шаге сечение проходит через вершины , , . Из вершины  возможен переход в вершину (78 + 11 = 89) и в вершину (80 + 18 = 98).

Для вершины возможен переход в вершину (80 + 10 = 90) и в вершину (83 + 10 = 93).

Для вершины возможен переход в вершину (83 + 12 = 95) и в вершину (88 + 10 = 98).

Выбираем для вершин , , наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

9-й шаг: *k* = 9. На девятом шаге сечение проходит через вершины , . Из вершины  возможен переход в вершину (89 + 10 = 99) и в вершину (90 + 10 = 100).

Для вершины возможен переход в вершину (90 + 10 = 100) и в вершину (95 + 15 = 110).

Выбираем для вершин , наименьшие суммарные издержки и обозначаем стрелкой условно оптимальный переход.

10-й шаг: *k* = 10. На девятом шаге сечение проходит через точку . Из вершины возможен переход в вершину (99 + 9 = 108) и в вершину (100 + 13 = 113).

Минимальные возможные суммарные издержки равны 108.

В результате получим граф условно оптимальных переходов, представленный на рис. 2.1.1.

*Рис 2.1.1. Сетевая модель связи расходов операций*

**II этап. Безусловная оптимизация.**

Определяем оптимальную траекторию на исходном сетевом графе, просматривая результаты всех шагов в обратном порядке, учитывая, что выбор некоторого управления на *k*-м шаге приводит к тому, что состояние на (*k* — 1)-м шаге становится определенным.

В результате строим ориентированный граф перехода из состояния в состояние, представленный на рис. 2.1.2; на каждом шаге безусловной оптимизации переход почти всегда единственный и совпадает с построенными условно оптимальными переходами.

*Рис 2.1.2. Оптимальная последовательность операций*

## 2.2 Решение задачи средствами прикладных программ

***2.2.1 Технология разработки формы для ввода исходных данных
средствами VBA***

Размещаем на рабочем листе Excel элемент управления Кнопка. Далее в открывшемся окне «Назначить макрос объекту», нажимаем на кнопку «Создать». В открывшемся окне MS Visual Basic создаем новую форму, для этого выберем команду Insert / UserForm.

В тексте процедуры CommandButton1\_Click() запишем команду UserForm1.Show, что позволит при нажатии на кнопку вызывать форму.

После этого размещаем объекты на форме UserForm1. Настройка объектов осуществляется в соответствии с параметрами таблицы 1 Настройка формы ввода данных в Excel в приложении А.

*Рис 2.2.1.1 Форма ввода данных в Excel*

В обработчике события CommandButton2\_Click() осуществим ввод данных согласно условию задачи.

***2.2.2 Описание процесса решения в Excel***

Ввод исходных данных в рабочую книгу Excel осуществляется на листе «Лист 1» по нажатию на кнопку «Форма для ввода данных». Далее нажимаем на кнопку «Определение оптимальной последовательности», после чего производится расчет.

*Рис 2.2.2.1 Таблица Excel для определения оптимальной последовательности*

В ячейки A1-M9 вводятся исходные данные, и рассчитываются минимальные издержки на каждом шаге.

В ячейке Е11 определяются минимальные возможные издержки.

Ячейки выделенные красным цветом показывают оптимальную последовательность операций по приемке и отпуску товаров с минимальными суммарными издержками.

Листинг расчетов в VBA представлен в приложении Б.

*Вывод.* В результате выполнения задачи в Excel’е на листе «Лист 1» выводится оптимальная последовательность операций по приемке и отпуску товаров с минимальными суммарными издержками (в ячейках выделенных красным цветом). Полученные результаты после решение задачи с помощью математического аппарата и решения задачи в Excel’е сходятся.

## 2.3 Автоматизация решения задачи

***2.3.1 Техническое задание на разработку программного продукта***

Полное наименование программного продукта – «Автоматизированная система решения задачи построения оптимальной последовательности операций в коммерческой деятельности». Краткое наименование программного продукта – «Optimum».

Данный программный продукт предназначен для минимизирования суммарных издержек.

Основанием для разработки данного проекта является задание на курсовое проектирование по дисциплине «Математические методы».

Наименование темы разработки − «Построение оптимальной последовательности операций в коммерческой деятельности».

Данная работа будет являться практической частью курсового проекта по дисциплине «Математические методы».

Целью данной разработки является оптимизации задачи сетевого моделирования.

Данный программный продукт должен совершать расчет минимальных издержек на каждом шаге, определять оптимальную последовательность операций.

Для нормального функционирования данного программного продукта нужен компьютер следующей конфигурации:

* центральный процессор управления (1,4ГГц);
* оперативная память (128 Мб);
* материнская плата;
* жёсткий диск (10Гб);
* монитор (15 дюймовый);
* клавиатура (стандартная 102);
* мышь (можно шариковую);
* видеокарта (можно интегрированную);
* устройства считывания носителей;
* ОС Windows 2000/XP.

Разработка программы должна вестись на языке программирования Borland Delphi v 7.0.

Данная программа должна работать в ОС Windows.

Испытания и тестирование программы должны проводиться в процессе создания программы самими разработчиками:

1. с использованием контрольных тестов, позволяющих добиться проверки правильности работоспособности и взаимной совместимости максимального числа функций и операторов программы или модуля при минимальных затратах временных и финансовых ресурсов.
2. путем пошагового исполнения программы или модуля (и непрерывного контроля значений переменных) в соответствии с набором тестовых примеров и сравнений полученных в процессе тестирования значений с контрольными значениями тестовых примеров.

Приемка программы должна осуществляться руководителем курсового проектирования. Программа должна считаться годной, если она удовлетворяет всем пунктам данного технического задания.

***2.3.2 Алгоритм решения задачи***

Блок-схема алгоритма разработана в соответствии с ГОСТ 19.701-90 и находится в приложении А.

# Заключение

В курсовом проекте была поставлена проблема построения оптимальной последовательности операций в коммерческой деятельности, обоснована её актуальность и практическая значимость.

В первой части были рассмотрены теоретические вопросы, раскрывающие суть проблемы курсового проекта.

Во второй части составлена математическая модель задачи, предложенной для курсового проекта, выполнено её решение с помощью математического аппарата, рассмотрены принципы использования прикладной программы MS Excel 2007 для ввода исходных данных и расчёта основных параметров указанной модели.

Для автоматизации процесса решения задачи курсового проекта не было разработано приложение«Optimum» с помощью среды программирования Borland Delphi 7.

Таким образом, не все поставленные задачи были выполнены, цель курсового проектирования достигнута частично.

# Библиографический список

1. Акулич, И. Л. Математическое программирование в примерах и задачах: Учеб. пособие / И. Л. Акулич. – СПб.: Лань, 2009.
2. Замков, О. О. Математические методы в экономике: Учеб. пособие для вузов / О. О. Замков, А. В. Толстопятенко, Ю. Н Черемных. – М.: Дело и Сервис, 2004.
3. Интрилигатор, М. Н. Математические методы оптимизации и экономическая теория: Учеб. пособие для вузов / М. Н. Интрилигатор. – М.: Айрис-Пресс, 2002.
4. Минюк, С. А. Дифференциальные уравнения и экономические модели: Учеб. пособие для вузов / С. А. Минюк, Н. С. Берёзкина. – Минск: [Вышэйшая школа](http://my-shop.ru/shop/producer/66.html), 2007.
5. Михеев, Р. Н. VBA и программирование в MS Office для пользователей / Р. Н. Михеев. – СПб.: БХВ-Петербург, 2006.
6. Пелих, А. С. Экономико-математические методы и модели в управлении производством: Учеб. пособие для вузов / А. С. Пелих, Л. Л. Терехов, Л. А. Терехова. – Ростов-на-Дону: Феникс, 2005.
7. Фомин, Г. П. Математические методы и модели в коммерческой деятельности: Учеб. пособие для вузов / Г. П. Фомин. – М.: Инфра-М, 2009.

# Приложение А Блок-схема алгоритма


# Приложение Б Листинг расчета в VBA

Private Sub CommandButton1\_Click()

UserForm1.Show

End Sub

Private Sub CommandButton2\_Click()

i1 = 1

l1 = 9

Do

 If (i1 <> 13) And (l1 <> 1) Then

 If (Cells(l1 - 2, i1) <= Cells(l1, i1 + 2)) And (Cells(l1 - 1, i1).Font.Italic = True) Then

 Cells(l1 - 2, i1).Interior.Color = 150

 l1 = l1 - 2

 i1 = i1

 ElseIf (Cells(l1 - 2, i1) >= Cells(l1, i1 + 2)) And (Cells(l1, i1 + 1).Font.Italic = True) Then

 Cells(l1, i1 + 2).Interior.Color = 150

 l1 = l1

 i1 = i1 + 2

 ElseIf Cells(l1, i1 + 1).Font.Italic = True Then

 Cells(l1, i1 + 2).Interior.Color = 150

 l1 = l1

 i1 = i1 + 2

 ElseIf Cells(l1 - 1, i1).Font.Italic = True Then

 Cells(l1 - 2, i1).Interior.Color = 150

 l1 = l1 - 2

 i1 = i1

 End If

 ElseIf (i1 = 13) And (l1 <> 1) Then

 Cells(l1 - 2, i1).Interior.Color = 150

 l1 = l1 - 2

 i1 = i1

 ElseIf (i1 <> 13) And (l1 = 1) Then

 Cells(l1, i1 + 2).Interior.Color = 150

 l1 = l1

 i1 = i1 + 2

 ElseIf (i1 = 13) And (l1 = 1) Then

 Cells(l1, i1).Interior.Color = 150

 Exit Do

 End If

Loop While (Cells(1, 13).Interior.Color <> 150)

End Sub

Function vvod()

Worksheets("Лист1").Range("B1") = TextBox1.Text

Worksheets("Лист1").Range("D1") = TextBox2.Text

Worksheets("Лист1").Range("F1") = TextBox3.Text

Worksheets("Лист1").Range("H1") = TextBox4.Text

Worksheets("Лист1").Range("J1") = TextBox5.Text

Worksheets("Лист1").Range("L1") = TextBox6.Text

Worksheets("Лист1").Range("A2") = TextBox7.Text

Worksheets("Лист1").Range("C2") = TextBox8.Text

Worksheets("Лист1").Range("E2") = TextBox9.Text

Worksheets("Лист1").Range("G2") = TextBox10.Text

Worksheets("Лист1").Range("I2") = TextBox11.Text

Worksheets("Лист1").Range("K2") = TextBox12.Text

Worksheets("Лист1").Range("M2") = TextBox13.Text

Worksheets("Лист1").Range("B3") = TextBox14.Text

Worksheets("Лист1").Range("D3") = TextBox15.Text

Worksheets("Лист1").Range("F3") = TextBox16.Text

Worksheets("Лист1").Range("H3") = TextBox17.Text

Worksheets("Лист1").Range("J3") = TextBox18.Text

Worksheets("Лист1").Range("L3") = TextBox19.Text

Worksheets("Лист1").Range("A4") = TextBox21.Text

Worksheets("Лист1").Range("C4") = TextBox22.Text

Worksheets("Лист1").Range("E4") = TextBox23.Text

Worksheets("Лист1").Range("G4") = TextBox24.Text

Worksheets("Лист1").Range("I4") = TextBox25.Text

Worksheets("Лист1").Range("K4") = TextBox26.Text

Worksheets("Лист1").Range("M4") = TextBox20.Text

Worksheets("Лист1").Range("B5") = TextBox28.Text

Worksheets("Лист1").Range("D5") = TextBox29.Text

Worksheets("Лист1").Range("F5") = TextBox30.Text

Worksheets("Лист1").Range("H5") = TextBox31.Text

Worksheets("Лист1").Range("J5") = TextBox32.Text

Worksheets("Лист1").Range("L5") = TextBox27.Text

Worksheets("Лист1").Range("A6") = TextBox34.Text

Worksheets("Лист1").Range("C6") = TextBox35.Text

Worksheets("Лист1").Range("E6") = TextBox36.Text

Worksheets("Лист1").Range("G6") = TextBox37.Text

Worksheets("Лист1").Range("I6") = TextBox38.Text

Worksheets("Лист1").Range("K6") = TextBox39.Text

Worksheets("Лист1").Range("M6") = TextBox33.Text

Worksheets("Лист1").Range("B7") = TextBox41.Text

Worksheets("Лист1").Range("D7") = TextBox42.Text

Worksheets("Лист1").Range("F7") = TextBox43.Text

Worksheets("Лист1").Range("H7") = TextBox44.Text

Worksheets("Лист1").Range("J7") = TextBox45.Text

Worksheets("Лист1").Range("L7") = TextBox40.Text

Worksheets("Лист1").Range("A8") = TextBox47.Text

Worksheets("Лист1").Range("C8") = TextBox48.Text

Worksheets("Лист1").Range("E8") = TextBox49.Text

Worksheets("Лист1").Range("G8") = TextBox50.Text

Worksheets("Лист1").Range("I8") = TextBox51.Text

Worksheets("Лист1").Range("K8") = TextBox52.Text

Worksheets("Лист1").Range("M8") = TextBox46.Text

Worksheets("Лист1").Range("B9") = TextBox54.Text

Worksheets("Лист1").Range("D9") = TextBox55.Text

Worksheets("Лист1").Range("F9") = TextBox56.Text

Worksheets("Лист1").Range("H9") = TextBox57.Text

Worksheets("Лист1").Range("J9") = TextBox58.Text

Worksheets("Лист1").Range("L9") = TextBox53.Text

Worksheets("Лист1").Range("m1") = "0"

Range("a9").Interior.Color = 150

Range("k1") = Range("m1") + Range("l1")

Range("i1") = Range("k1") + Range("j1")

Range("g1") = Range("i1") + Range("h1")

Range("e1") = Range("g1") + Range("f1")

Range("c1") = Range("e1") + Range("d1")

Range("a1") = Range("c1") + Range("b1")

Range("m3") = Range("m1") + Range("m2")

Range("m5") = Range("m3") + Range("m4")

Range("m7") = Range("m5") + Range("m6")

Range("m9") = Range("m7") + Range("m8")

If (Range("K1") + Range("K2")) <= (Range("M3") + Range("L3")) Then

 Range("K3") = Range("K1") + Range("K2")

 Range("k2").Font.Italic = True

 Else: Range("K3") = Range("M3") + Range("L3")

 Range("l3").Font.Italic = True

End If

If (Range("I1") + Range("I2")) <= (Range("K3") + Range("J3")) Then

 Range("I3") = Range("I1") + Range("I2")

 Range("i2").Font.Italic = True

 Else: Range("I3") = Range("K3") + Range("j3")

 Range("j3").Font.Italic = True

End If

If (Range("g1") + Range("g2")) <= (Range("i3") + Range("h3")) Then

 Range("g3") = Range("g1") + Range("g2")

 Range("g2").Font.Italic = True

 Else: Range("g3") = Range("i3") + Range("h3")

 Range("h3").Font.Italic = True

End If

If (Range("e1") + Range("e2")) <= (Range("g3") + Range("f3")) Then

 Range("e3") = Range("e1") + Range("e2")

 Range("e2").Font.Italic = True

 Else: Range("e3") = Range("g3") + Range("f3")

 Range("f3").Font.Italic = True

End If

If (Range("c1") + Range("c2")) <= (Range("e3") + Range("d3")) Then

 Range("c3") = Range("c1") + Range("c2")

 Range("c2").Font.Italic = True

 Else: Range("c3") = Range("e3") + Range("d3")

 Range("d3").Font.Italic = True

End If

If (Range("a1") + Range("a2")) <= (Range("c3") + Range("b3")) Then

 Range("a3") = Range("a1") + Range("a2")

 Range("a2").Font.Italic = True

 Else: Range("a3") = Range("c3") + Range("b3")

 Range("b3").Font.Italic = True

End If

If (Range("K3") + Range("K4")) <= (Range("M5") + Range("L5")) Then

 Range("K5") = Range("K3") + Range("K4")

 Range("k4").Font.Italic = True

 Else: Range("K5") = Range("M5") + Range("L5")

 Range("l5").Font.Italic = True

End If

If (Range("i3") + Range("i4")) <= (Range("k5") + Range("j5")) Then

 Range("i5") = Range("i3") + Range("i4")

 Range("i4").Font.Italic = True

 Else: Range("i5") = Range("k5") + Range("j5")

 Range("j5").Font.Italic = True

End If

If (Range("g3") + Range("g4")) <= (Range("i5") + Range("h5")) Then

 Range("g5") = Range("g3") + Range("g4")

 Range("g4").Font.Italic = True

 Else: Range("ig") = Range("i5") + Range("h5")

 Range("h5").Font.Italic = True

End If

If (Range("e3") + Range("e4")) <= (Range("g5") + Range("f5")) Then

 Range("e5") = Range("e3") + Range("e4")

 Range("e4").Font.Italic = True

 Else: Range("e5") = Range("g5") + Range("f5")

 Range("f5").Font.Italic = True

End If

If (Range("c3") + Range("c4")) <= (Range("e5") + Range("d5")) Then

 Range("c5") = Range("c3") + Range("c4")

 Range("c4").Font.Italic = True

 Else: Range("c5") = Range("e5") + Range("d5")

 Range("d5").Font.Italic = True

End If

If (Range("a3") + Range("a4")) <= (Range("c5") + Range("b5")) Then

 Range("a5") = Range("a3") + Range("a4")

 Range("a4").Font.Italic = True

 Else: Range("a5") = Range("c5") + Range("b5")

 Range("b5").Font.Italic = True

End If

If (Range("k5") + Range("k6")) <= (Range("m7") + Range("l7")) Then

 Range("k7") = Range("k5") + Range("k6")

 Range("k6").Font.Italic = True

 Else: Range("k7") = Range("m7") + Range("l7")

 Range("l7").Font.Italic = True

End If

If (Range("i5") + Range("i6")) <= (Range("k7") + Range("j7")) Then

 Range("i7") = Range("i5") + Range("i6")

 Range("i6").Font.Italic = True

 Else: Range("i7") = Range("k7") + Range("j7")

 Range("j7").Font.Italic = True

End If

If (Range("g5") + Range("g6")) <= (Range("i7") + Range("h7")) Then

 Range("g7") = Range("g5") + Range("g6")

 Range("g6").Font.Italic = True

 Else: Range("g7") = Range("i7") + Range("h7")

 Range("h7").Font.Italic = True

End If

If (Range("e5") + Range("e6")) <= (Range("g7") + Range("f7")) Then

 Range("e7") = Range("e5") + Range("e6")

 Range("e6").Font.Italic = True

 Else: Range("e7") = Range("g7") + Range("f7")

 Range("f7").Font.Italic = True

End If

If (Range("c5") + Range("c6")) <= (Range("e7") + Range("d7")) Then

 Range("c7") = Range("c5") + Range("c6")

 Range("c6").Font.Italic = True

 Else: Range("c7") = Range("e7") + Range("d7")

 Range("d7").Font.Italic = True

End If

If (Range("a5") + Range("a6")) <= (Range("c7") + Range("b7")) Then

 Range("a7") = Range("a5") + Range("a6")

 Range("a6").Font.Italic = True

 Else: Range("a7") = Range("c7") + Range("b7")

 Range("b7").Font.Italic = True

End If

If (Range("k5") + Range("k6")) <= (Range("m7") + Range("l7")) Then

 Range("k7") = Range("k5") + Range("k6")

 Range("k6").Font.Italic = True

 Else: Range("k7") = Range("m7") + Range("l7")

 Range("l7").Font.Italic = True

End If

If (Range("k7") + Range("k8")) <= (Range("m9") + Range("l9")) Then

 Range("k9") = Range("k7") + Range("k8")

 Range("k8").Font.Italic = True

 Else: Range("k9") = Range("m9") + Range("l9")

 Range("l9").Font.Italic = True

End If

If (Range("i7") + Range("i8")) <= (Range("k9") + Range("j9")) Then

 Range("i9") = Range("i7") + Range("i8")

 Range("i8").Font.Italic = True

 Else: Range("i9") = Range("k9") + Range("j9")

 Range("j9").Font.Italic = True

End If

If (Range("g7") + Range("g8")) <= (Range("i9") + Range("h9")) Then

 Range("g9") = Range("g7") + Range("g8")

 Range("g8").Font.Italic = True

 Else: Range("g9") = Range("i9") + Range("h9")

 Range("h9").Font.Italic = True

End If

If (Range("e7") + Range("e8")) <= (Range("g9") + Range("f9")) Then

 Range("e9") = Range("e7") + Range("e8")

 Range("e8").Font.Italic = True

 Else: Range("e9") = Range("g9") + Range("f9")

 Range("f9").Font.Italic = True

End If

If (Range("c7") + Range("c8")) <= (Range("e9") + Range("d9")) Then

 Range("c9") = Range("c7") + Range("c8")

 Range("c8").Font.Italic = True

 Else: Range("c9") = Range("e9") + Range("d9")

 Range("d9").Font.Italic = True

End If

If (Range("a7") + Range("a8")) <= (Range("c9") + Range("b9")) Then

 Range("a9") = Range("a7") + Range("a8")

 Range("a8").Font.Italic = True

 Else: Range("a9") = Range("c9") + Range("b9")

 Range("b9").Font.Italic = True

End If

End Function

Private Sub CommandButton2\_Click()

TextBox1.Text = "10"

TextBox2.Text = "8"

 TextBox3.Text = "9"

TextBox4.Text = "11"

 TextBox5.Text = "13"

 TextBox6.Text = "12"

 TextBox7.Text = "15"

TextBox8.Text = "14"

 TextBox9.Text = "13"

 TextBox10.Text = "12"

 TextBox11.Text = "11"

 TextBox12.Text = "10"

 TextBox13.Text = "9"

 TextBox14.Text = "12"

TextBox15.Text = "13"

TextBox16.Text = "14"

 TextBox17.Text = "15"

 TextBox18.Text = "14"

 TextBox19.Text = "13"

 TextBox21.Text = "11"

 TextBox22.Text = "13"

 TextBox23.Text = "12"

 TextBox24.Text = "14"

 TextBox25.Text = "13"

 TextBox26.Text = "15"

 TextBox20.Text = "10"

 TextBox28.Text = "18"

 TextBox29.Text = "19"

 TextBox30.Text = "20"

 TextBox31.Text = "21"

 TextBox32.Text = "18"

 TextBox27.Text = "19"

 TextBox34.Text = "10"

 TextBox35.Text = "10"

 TextBox36.Text = "13"

 TextBox37.Text = "11"

 TextBox38.Text = "12"

 TextBox39.Text = "14"

 TextBox33.Text = "11"

 TextBox41.Text = "10"

 TextBox42.Text = "10"

 TextBox43.Text = "12"

TextBox44.Text = "12"

 TextBox45.Text = "13"

 TextBox40.Text = "18"

TextBox47.Text = "9"

 TextBox48.Text = "10"

 TextBox49.Text = "12"

 TextBox50.Text = "15"

 TextBox51.Text = "13"

TextBox52.Text = "16"

TextBox46.Text = "19"

 TextBox54.Text = "13"

 TextBox55.Text = "15"

 TextBox56.Text = "10"

 TextBox57.Text = "14"

TextBox58.Text = "16"

TextBox53.Text = "17"

End Sub