**РЕФЕРАТ**

Пояснительная записка: 23 с., 2 рис., 4 источника, 5 таблиц

Целью работы является создание программного продукта на языке программирования Visual Basic for Applications согласно заданию, выданному на учебную практику. Данный программный продукт позволяет разместить в Microsoft Word исходный текст, введенный пользователем, на странице, каждая строка которой имеет желаемое пользователем количество символов (причем слова не разрываются на две строки) и в Microsoft Excel найти сумму двух разреженных полиномов, заданных ненулевыми коэффициентами и их номерами. Программный продукт должен обеспечивать удобство и простоту использования, быть функциональным и надежным, эффективным в эксплуатации.

Программный продукт может привлечь пользователей, которые хотят узнать дополнительную информацию об Visual Basic for Applications, изучить основную информацию и проследить за разработкой программ на данном языке программирования.

**ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Гарнаев А.Ю. Самоучитель VBA – 2-е изд., перераб. и доп. − СПб.: БВХ- Петербург 2004, 560с.
2. Гарнаев А.Ю. «Excel 2002: разработка приложений» − СПб.: БВХ-Петербург 2002, 450с.
3. С.Браун «VB 6» − СПб.: Москва – Харьков – Минск 2002, 573с.
4. С.Каммингс «VBA для Чайников» − СПб.: Москва – Санкт-Петербург – Киев 2003, 440с.

ВЫВОДЫ

В процессе работы над заданием по учебной практике была разработана информационная система. В данной системе реализованы следующие возможности:

* вывод справочной информации по языку программирования Visual Basic for Applications;
* нахождение в заданной строке слов, которые начинаются и заканчиваются одной и той же буквой; вывод слов на экран в Microsoft Word;
* с помощью определенных преобразований в исходной матрице изменить элементы четных сток, вывести конечную матрицу на экран в Microsoft Exel.

Впоследствии в ПП возможно увеличить объем теоретического материала по VBA, усовершенствовать интерфейс. Недостатков данной системы разработчиком не обнаружено.

### **ВВЕДЕНИЕ**

Visual Basic for Applications (VBA) – весьма популярная в последние годы среда разработки приложений, которая, являясь мощным программным средством, позволяет реализовать широкий спектр практических задач. Основное её достоинство состоит в оптимальном сочетании простоты использования, доступности и большого набора разнообразных возможностей, позволяющих охватить все основные направления деятельности разработчика прикладных программ.

Возможности VBA совсем не уступают другим системам программирования Windows и постоянно растут. К тому же многие системы программирования в Windows и используют общие библиотеки объектов, среди которых – объекты Word и Excel. К основным возможностям VBA можно отнести реализацию максимально гибкого и удобного интерфейса для приложения, создание самых разнообразных многоуровневых и всплывающих меню, обработку событий мыши и клавиатуры, работу с графикой. Разработчик может использовать в программе как стандартные, так и встроенные диалоги, применяющиеся для организации обратной связи с пользователем. Приложение может осуществлять работу с таймером, обрабатывать файлы и каталоги на жестком диске, а также предоставлять доступ к базам данных. В VBA имеется возможность внедрения в приложение различных объектов, которые поддерживаются другими программами. Помимо этого, разработчик может реализовать связь одного или нескольких компьютеров, на которых установлено созданное приложение, с сетью Internet.

Также язык программирования VBA – есть реализация объектно-ориентированного написания программ в системе MS Office, а в частности в процессе практики исследовалось использование VBA для MS Word и MS Excel. Синтаксис языка – издавна проложенные правила языка Basic – один из самых ранних языков программирования. Процесс написания программы на VBA – последовательное заполнение параметров различных объектов. Язык VBA, является мощным программным средством, осуществлена возможность обработки файловой информации. Но самым ярким достоинством языка является возможность занесения результатов вычисления и любых данных в Office приложения, в MS Word и Ms Excel.

То есть можно отметить, что в настоящее время большое внимание уделяется программированию на Visual Basic for Applications. Именно поэтому задание, выданное на учебную практику, разрабатывается на данном языке программирования.

## **1 ПОСТАНОВКА ЗАДАЧИ**

* 1. **Формулировка задачи**

Для MS WORD:

Исходный текст (слова, пробелы, знаки) задан одной строкой. Нахождение в ней слов, которые начинаются и заканчиваются одинаковой буквой.

Для MS EXCEL:

В матрице найти сумму элементов, которые расположены по главной диагонали. Элементы всех чётных строк разделить на эту сумму.

**1.2 Исходные данные**

Для WORD:

S1 : строка ; { строка, введенная пользователем }

Для EXCEL:

a : двумерный массив [5,5 ] вещ ; { массив коэффициентов первого полинома }

* 1. **Ограничения на исходные данные**

Для EXCEL:

0 < = i < 5

0 < = j < 5

* 1. **Промежуточные данные**

Для WORD:

S3: строка ; {содержит слово, нужное по условию}

k : цел ; { счетчик, определяющий количество символов в каждом слове }

start : цел ; { счетчик перебора символов в строке }

i : цел ; { счетчик, показывающий начиная с какого символа будет

осуществляться чтение слова }

Для EXCEL:

B: массив [10] вещественных чисел; {содержит элементы главной диагонали}

* 1. **Результаты**

Для WORD:

S3 : строка ; { содержит искомые слова}

Для EXCEL:

a : массив [5,5 ] вещ ; { конечная матрица чисел}

### **2 ОПИСАНИЕ ЯЗЫКА VBA И СИСТЕМЫ ПРОГРАММИРОВАНИЯ В OFFICE**

Операционная система Windows корпорации Microsoft обладает изумительно удобным интерфейсом и множеством интеллектуальных средств, которые освобождают пользователя от рутинной работы, присущей другим операционным системам. Естественным и незаменимым дополнением этой операционной системы являются программные продукты Microsoft Office, которые даже начинающему пользователю позволяют с легкостью создавать идеальную по своему оформлению документацию, производить финансовые, инженерные и прочие расчеты, создавать и работать с базами данных, конструировать презентации и слайды. Вместе, операционная система Windows и программные продукты Microsoft Office, покорили весь мир. Корпорация Microsoft интегрировала в свои офисные продукты, в саму операционную среду изумительный по простоте и необычайный по своей эффективности язык программирования Visual Basic for Applications или сокращенно VBA. С помощью этого языка теперь каждый пользователь может автоматизировать работу приложения и максимально приспособить его работу для решения текущих задач, не только добавив интерфейсу новую функциональность, но и удалив из него ненужные для данного приложения элементы, тем самым обеспечив ему дополнительную стабильность.

Возможности VBA совсем не уступают другим системам программирования Windows и постоянно растут. К тому же многие системы программирования в Windows и VBA используют общие библиотеки объектов, среди которых – объекты Excel, Word и т.д.

* 1. **Типы данных**

Типы данных относятся к фундаментальным понятиям любого языка. Тип данных определяет множество допустимых значений, которое может принимать указанная переменная. В VBA имеются следующие типы данных:

* Byte (байтовый);
* Boolean (логический);
* Integer (целый);
* Long (длинный целый);
* Single (с плавающей точкой обычной точности);
* Double (с плавающей точкой двойной точности);
* Currency (денежный);
* Decimal (масштабируемый целый);
* Date (даты и время);
* Object (объектный);
* String (строковый);
* Variant (произвольный);
* тип данных, определяемый пользователем;
* специфические типы объектов.

Для хранения целочисленных данных предназначены типы Integer (диапазон значений от –32768 до 32767, в оперативной памяти занимает два байта) и Long (диапазон значений от –2147483648 до 21447483648 и в оперативной памяти занимает четыре байта). Аналогично, для хранения десятичных чисел с плавающей точкой представлены типы данных Single (диапазон от –3,4Е38 до –1,4Е-45 для отрицательных чисел и от 1,4Е-45 до 3,4Е38 для положительных чисел, в оперативной памяти занимает четыре байта) и Double (диапазон от –1,7Е-308 до –4,9Е-324 для отрицательных чисел и от 4,9Е-324 до 1,7Е308 для положительных чисел, в оперативной памяти занимает 8 байтов). Для хранения чисел с фиксированной точкой служит тип Currency. Переменная данного типа занимает в оперативной памяти 8 байт. Целая часть числа может содержать до 15 цифр, а дробная – до 4.

Используя переменные строкового типа, можно хранить строки как фиксированной, так и переменной длины. Для переменной, имеющей тип строки с фиксированной длинной, указывается максимальная длина строки. Если длина строки, присваиваемая этой переменной, меньше заданной длины строки, то оставшиеся свободные места заполняются пробелами. При присвоении переменной строки, количество символов которой превышает максимальное значение, все лишние символы отбрасываются. Количество символов, хранящихся в строке с переменной длиной, определяется длинной присваиваемых ей данных. При использовании ASCCII кодировки для каждого символа строки выделяется один байт, а при использовании кодировки Unicode – два байта. Кроме этого выделяется 10 байт для строки в целом.

Для хранения двоичных данных рекомендуется использовать массив переменных типа Byte. Каждый элемент массива данного типа занимает один байт оперативной памяти.

Логический тип данных Boolean используется для хранения данных, принимающих одно из двух значений: True или False. По умолчанию переменной данного типа присваивается значение False. В оперативной памяти она занимает два байта.

Тип данных Date используется для хранения даты и времени. Переменная этого типа требует 8 байт в оперативной памяти.

Тип данных Variant является универсальным. Переменная этого типа может хранить любой из выше перечисленных типов данных. Все необходимые преобразования выполняются при присвоении переменной значения. Выделяемый переменной объем оперативной памяти зависит от типа присвоенного значения, но не может быть менее 16 байт.

Для хранения ссылок на объекты используется тип данных Object. Каждая переменная данного типа требует 4 байта.

При разработке программ в среде VBA в зависимости от типа данных переменных рекомендуется использовать префиксы, приведенные в таблице 2.1

Таблица 2.1 – Префиксы, используемые в наименованиях переменных

|  |  |  |
| --- | --- | --- |
| Тип данных | Префикс | Пример |
| Boolean | bin | binSuc |
| Byte | byt | bytImege |
| Currency | cur | curPrn |
| Date | dtm | dtmFinish |
| Double | dbl | dblSum |
| Integer | int | intKol |
| Long | Ing | IngLoop |
| Single | sng | sngTotal |
| String | str | strLast |
| Variant | vnt | vntValui |

* 1. **Описание переменных**

Термины время жизни и область видимости переменной означают место использования переменной в приложении, а также время существования переменной после ее создания.

Область видимости переменной определяет часть кода, которая «знает» о существовании данной переменной. При определении переменной в процедуре получить или изменить ее значение можно только из кода этой процедуры. Иногда, однако, необходимо использовать переменную с более обширной областью видимости. Например, переменную, значения которой доступны для всех процедур модуля или проекта. Существуют три типа области видимости переменной:

* переменные уровня процедуры распознаются только в процедуре, в которой они описаны. Они описываются при помощи инструкций Dim или Static. Такие переменные называются локальными;
* переменные уровня модуля используются только в модуле, в котором они описаны, но не в других модулях данного проекта. Описываются при помощи оператора Dim или Private в области описания модуля, т.е. перед описанием процедур;
* переменные уровня модуля, описанные при помощи инструкции Public, являются доступными для всех процедур проекта. Такие переменные называются открытыми.

Закрытая (Private) переменная сохраняет свое значение, только пока выполняется процедура, в которой эта переменная описана. При завершении процедуры значение переменной теряется, и при повторном запуске процедуры его надо заново инициализировать. Переменные, описанные оператором Static, сохраняют свое значение по выходу из процедуры, пока работает программа.

Для обязательного объявления всех переменных в начале модуля, в так называемой области модуля General Declarations, надо поместить директиву Option Explicit. Использование этой директивы не допускает возможности неправильного ввода имени переменной, которая применяется в одной или не − скольких процедурах модуля. Например, если переменная была объявлена как Ставка, а в коде при наборе вместо русской буквы с была использована латинская буква с, то это приведет к ошибке. В отсутствие директивы Option Explicit подобную ошибку было бы трудно отследить.

* 1. **Описание массивов**

Как и в других языках программирования, в VBA можно использовать массивы. Массив, как и любую переменную, надо объявлять, используя операторы Dim, Static, Private и Public, которые также задают область видимости переменной. В массиве допускается описание до 60 размерностей. При определении размерности надо указывать верхнюю, а также нижнюю границу. Если нижний индекс не задан явно, нижняя граница массива определяется директивой Option Base. Если отсутствует директива Option Base, нижняя граница массива равняется нулю. Например, в следующем операторе объявляется одномерный массив (вектор) из 12 целых чисел, причем по умолчанию первый элемент массива – А (0), а последний – А (11). В этом случае говоря, что 0 – базовый индекс.

Dim А (11) As Integer

Данный же оператор объявляет двухмерный массив 3×3 (матрицу), состоящий из действительных чисел.

Dim B (2, 2) As Single

Можно изменить базовый индекс, написав в области объявлений модуля директиву Option Base 1. После этого индексы массивов А и В будут начинаться с единицы. Например, в следующем операторе объявляется вектор, состоящий из 11 элементов.

Option Base 1

Dim A (11) As Integer

Другим способом изменения базового индекса является использование ключевого слова То при объявлении массива.

Dim B (1 To 3, 1 To 3) As Single

Dim A (1 To 12) As Integer

Инициализацию элементов массива можно производить по-разному:

* последовательностью операторов:

Dim B (1, 1) As Single

B (0, 0) = 2 : B (0, 1) = 4

B (1, 0) = 1 : B (1, 1) = 6

* оператором цикла:

Dim M (1 To 9, 1 To 9) As Integer

Dim i As Integer

Dim j As Integer

For i = 1 To 9

For j = 1 To 9

M (i, j) = i \* j

Next

Next

Иногда в процессе выполнения программы требуется изменять размер массива. В этом случае первоначально массив объявляют как динамический. Для этого при объявлении массива не надо указывать размерность, например:

Dim R () As Single

Затем в программе следует вычислить необходимый размер массива в некоторой переменной, например n, и изменить размер динамического массива с помощью оператора ReDim. В следующем примере сначала объявляется динамический массив, а затем устанавливаются границы его индекса.

Dim R () As Double

ReDim R (1 To 10)

* 1. **Операторы языка и управляющие конструкции**

Математические операторы позволяют выполнять в программе действия над числами. В таблице 2.2 приведены арифметические операторы и выполняемые ими функции.

Таблица 2.1 – Префиксы, используемые в наименованиях переменных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Операция | Знак | Запись | Типы данных | Действие |
| Сложение | + | a + b | Byte, Short, Integer, Long, Single, Double, Decimal | Складывает два числа |
| Вычитание | - | a - b | Byte, Short, Integer, Long, Single, Double, Decimal | Вычитает из одного  числа другое |
| Деление | / | a/b | Byte, Short, Integer, Long, Single, Double, Decimal | Делит два числа и  возвращает результат  с плавающей точкой |
| Деление  нацело | \ | a\b | Byte, Short, Integer, Long, Single, Double, Decimal | Делит два числа и  возвращает целый результат (остаток отбрасывается) |
| Остаток | Mod | a Mod b | Byte, Short, Integer, Long, Single, Double, Decimal | Вычисляется остаток  от деления одного числа на другое |
| Умножение | \* | а \* b | Byte, Short, Integer, Long, Single, Double, Decimal | Умножает два числа |
| Возведение  в степень | ^ | a ^ b | Byte, Short, Integer, Long, Single, Double, Decimal | Возводит число в степень |

Математические операторы предназначены для создания выражений. Выражения могут содержать переменные, константы, функции, связанные более чем одним оператором. Если в выражении отсутствуют скобки, то операторы выполняются в следующем порядке:

* возведение в степень;
* умножение и деление;
* деление нацело;
* взятие остатка от деления;
* сложение и вычитание.

Можно изменить порядок вычисления в выражении, используя круглые скобки.

В VBA, как и во всех языках программирования, существуют управляющие конструкции, предназначенные для управления порядком выполнения команд. Различают два основных типа управляющих операторов:

- If

- Select..Case

Конструкция If используется в том случае, когда необходимо, чтобы группа операторов выполнялась при соблюдении определенных условий. Конструкция Select..Case позволяет на основании анализа значения заданного выражения выполнять те или иные действия.

В свою очередь, управляющие операторы if бывают двух видов:

- If...Then

- If...Then...Else

Конструкция If...Then применяется, когда необходимо выполнить определенные действия в зависимости от некоторого условия. Управляющая конструкция If...Then...Else используется в том случае, когда необходимо выполнить разные действия в зависимости от условия. Конструкция If.. .Then.. .Else аналогична конструкции If...Then, но позволяет задать действия, исполняемые как при выполнении условий, так и в случае их невыполнения.

Основанием для принятия решений в управляющих конструкциях являются условные выражения, поэтому предварительно необходимо сказать несколько слов об этих выражениях и работе с ними.

Условные выражения – это такие выражения, которые возвращают одно из двух значений: Истина или Ложь. В условных выражениях используют операторы сравнения, приведенные в таблице 2.3.

Таблица 2.3 – Операторы сравнения для условных выражений

|  |  |
| --- | --- |
| Операция | Знак |
| Меньше | < |
| Меньше или равно | <= |
| Больше | > |
| Операция | Знак |
| Больше или равно | >= |
| Равно | = |
| Не равно | <> |

Над условными выражениями можно выполнять действия логической математики (логические операции). К логическим побитовым операциям относятся операции And, Or, And Also, Or Else, Xor, Not.Операция Not имеет один операнд, остальные – по два операнда.

Ключевые слова IF и End If имеют тот же смысл, что и в конструкции If...Then. Если заданное в конструкции условие не выполняется (результат проверки равен False), и конструкция содержит ключевое слово Else, Visual Basic выполнит последовательность конструкций, расположенных следом за Else. После чего управление перейдет к конструкции, следующей после End If.

* 1. **Стандартные функции работы с файлами, строками, диалоговыми окнами**

В VBA имеется ряд функций и операторов по работе с файлами и каталогами. Одним из удобств является то, что не требуется устанавливать ссылку на используемую библиотеку. В таблице 2.4 эти функции и представлены.

Таблица 2.4 – Функции и операторы для работы с файлами

|  |  |
| --- | --- |
| Функция, оператор | Назначение |
| Open | Открывает файл |
| Close | Закрывает все файлы |
| Close # | Закрывает файл по идентификатору |
| Reset | Закрывает все открытые файлы, записывает содержимое буферов. |
| Print tt | Записывает данные в файл |
| FileCopy | Копирует файл |
| EOF | Определяет метку конца файла |
| FileAttr | Возвращает режим доступа открытого файла |
| FileDateTime | Возвращает дату и время создания файла |
| FileLen | Возвращает размер файла в байтах |
| FreeFile | Возвращает номер свободного идентификатора |
| GetAttr | Получает атрибуты файла |
| SetAttr | Устанавливает атрибуты файла |
| Loc | Возвращает номер текущей позиции в файле |
| LOF | Возвращает размер открытого файла в байтах |
| Seek | Устанавливает на заданную номером позицию или запись в файле |
| Dir | Возвращает содержимое текущей папки |
| Kill | Удаляет файл |
| Lock | Блокирует файл при работе в многопользовательской среде |
| Unlock | Снимает блокировку файла в многопользовательской среде |
| Name | Задает имя файла |
| Get # | Читает данные из файла |
| Input | Читает данные из файла |
| Input # | Читает данные из файла |
| Line Input # | Читает строку из файла |
| Put # | Записывает данные в файл |
| Write # | Записывает данные в файл |

В таблице 2.5 представлены функции обработки строковых выражений, которые позволяют произвести широкий спектр операций: от нахождения длины строки, до замены в строке подстрок.

Таблица 2.5 – Функции для работы со строками

|  |  |
| --- | --- |
| Функция | Назначение |
| Asc | Возвращает ASCII- код символа |
| Chr | Преобразовывает ASCII-код в символ |
| InStr, InStrRev | Осуществляет поиск одной строки в другой |
| Lcase | Изменяет регистр букв исходной строки на нижний |
| Left | Возвращает указанное количество символов с начала строки |
| Len | Возвращает количество символов в строке |
| Ltrim, RTrim, Trim | Удаляют пробелы, расположенные соответственно в начале, в конце и с обеих сторон символьной строки |
| Функция | Назначение |
| Mid | Возвращает заданное количество символов из произвольного места строки |
| Right | Возвращает указанное количество символов с конца строки |
| Str, CStr | Преобразовывают числовое выражение в строку |
| StrReverse | Изменяет порядок следования символов в строке на обратный |
| StrConv | Изменяет регистр букв символьной строки |
| Продолжение таблицы 2.5 | |
| Val | Преобразовывают строку в числовое выражение |
| UCase | Изменяет регистр букв исходной строки на нижний |

В проектах VBA часто встречаются две разновидности диалоговых окон: окна сообщений и окна ввода. Они встроены в VBA, и если их возможностей достаточно, то можно обойтись без проектирования диалоговых окон. Окно сообщений выводит простейшие сообщения для пользователя, а окно ввода обеспечивает ввод информации.

Функция InputBox выводит на экран диалоговое окно, содержащее сообщение, поле ввода, и две кнопки OK и Cancel. Она устанавливает режим ожидания ввода текста пользователем и нажатия кнопки, а затем, при нажатии на кнопку OK, возвращает значение типа String, содержащее текст, введенный в поле ввода. При нажатии кнопки Cancel возвращается пустая строка.

Процедура MsgBox выводит на экран диалоговое окно, содержащее сообщение, устанавливает режим ожидания нажатия кнопки пользователем, а затем возвращает значение типа Integer, указывающее, какая кнопка была нажата.

**3 АЛГОРИТМ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ В MS WORD**

S1

Для i=1 до len(s1)-1

Если mid(s1, i,1)=” ”

Если mid(s2,1,1)=mid(s2,len(s2),1)

S3=s3+s2+” ”

s2 = ""

Иначе

s2 = s2 + Mid(s1, i, 1)

s3

**4 АЛГОРИТМ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ В MS EXCEL**

Для i=1 до 5

Для j= 1 до 5

a(i,j)

sum = 0

Для i=1 до 5

b[i]=a[i,i]

sum=sum+b[i]

Для i=1 до 5

Для j=1 до 5

i=i+2

a[i,j]=a[i,j]/sum

Для i=1 до 5

Для j=1 до 5

a(i,j)

**5 ТЕСТОВЫЕ ПРИМЕРЫ**

MS WORD

№1

Дед Мороз - красный нос, ты подарки нам принёс?

Дед

№2

Трактат о несчастной любви.

Трактат

MS EXCEL

№1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 |
| 7 | 9 | 11 | 13 | 15 |
| 2 | 4 | 4 | 6 | 7 |
| 3 | 6 | 4 | 0 | 8 |

Результат

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 0,18 | 0,21 | 0,24 | 0,27 | 0,3 |
| 7 | 9 | 11 | 13 | 15 |
| 0,06 | 0,121 | 0,121 | 0,18 | 0,21 |
| 3 | 6 | 4 | 0 | 8 |

**Приложение А**

ЭКРАННЫЕ ФОРМЫ

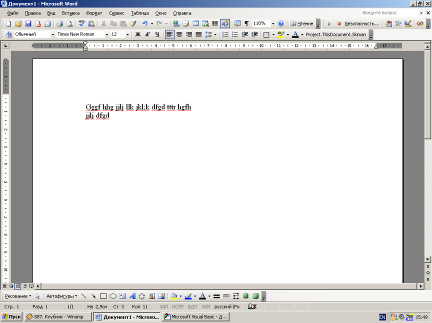


Рисунок А.1 – Работа программы в MS Word

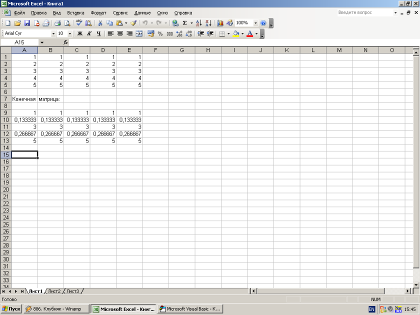


Рисунок А.2 – Пример работы программы в MS Word

**Приложение Б**

ЛИСТИНГ ПРОГРАММЫ В MS EXCEL

Sub Laba1()

Dim m, i, j, k As Integer

Dim sum As Double

Dim a(5, 5), b(10) As Long

For i = 1 To 5

For j = 1 To 5

a(i, j) = Cells(i, j)

Next j

Next i

sum = 0

For i = 1 To 5

k = i

j = i

b(k) = a(i, j)

sum = sum + b(k)

Next i

For i = 2 To 5 Step 2

For j = 1 To 5

a(i, j) = a(i, j) / sum

Next j

Next i

Cells(7, 1) = "Конечная"

Cells(7, 2) = "матрица:"

For i = 1 To 5

For j = 1 To 5

Cells(i + 8, j) = a(i, j)

Next j

Next i

End Sub

**Приложение В**

ЛИСТИНГ ПРОГРАММЫ В MS WORD

Sub Laba2()

Dim s1, s2, s3 As String

Dim i, j As Integer

Selection.WholeStory

s1 = Selection.Text

For i = 1 To Len(s1) - 1

If Mid(s1, i, 1) = " " Then

If Mid(s2, 1, 1) = Mid(s2, Len(s2), 1) Then

s3 = s3 + s2 + " "

End If

s2 = ""

Else

s2 = s2 + Mid(s1, i, 1)

End If

Next i

Selection.Text = s1 + s3

End Sub