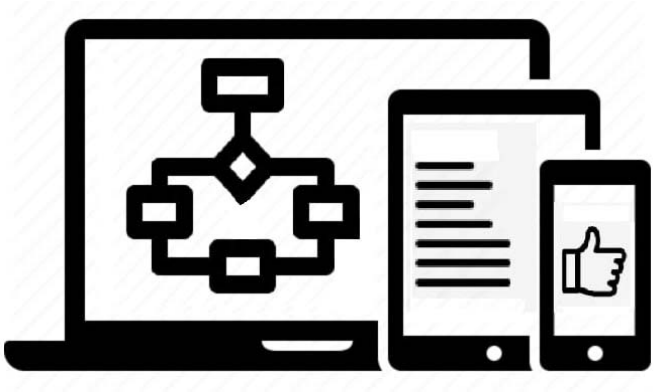
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

ИНФОРМАТИКА

Методические рекомендации к лабораторным работам для студентов направлений подготовки 15.03.03 «Прикладная механика» и 21.03.01 «Нефтегазовое дело» очной формы обучения

Часть 2



Могилев 2024

УДК 004
ББК 32.973
И74

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Программное обеспечение информационных технологий» «27» марта 2024 г., протокол № 9

Составитель ст. преподаватель В. М. Прудников

Рецензент канд. техн. наук, доц. И. В. Лесковец

Методические рекомендации к лабораторным работам по дисциплине «Информатика» предназначены для студентов направлений подготовки 15.03.03 «Прикладная механика» и 21.03.01 «Нефтегазовое дело» очной формы обучения. Приведен список литературы для подготовки к лабораторным работам.

Учебное издание

ИНФОРМАТИКА

Часть 2

Ответственный за выпуск	В. В. Кутузов
Корректор	И. В. Голубцова
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 16 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение

Цель методических рекомендаций – помочь студентам в подготовке к выполнению лабораторных работ по дисциплине «Информатика».

Цель преподавания дисциплины – изучение основных современных операционных систем и программных сред, пакетов прикладных программ для научных и инженерных расчетов, изучение основ программирования, общих вопросов алгоритмизации и приобретение навыков решения задач с применением средств вычислительной техники.

Дисциплина «Информатика» является неотъемлемой частью современных инженерных знаний и входит в состав естественно-научных дисциплин, компонентов учреждения высшего образования.

Полученные при изучении дисциплины знания и навыки востребованы при изучении специальных дисциплин инженерной направленности и станут инструментом для грамотного выполнения и оформления рефератов, курсовых и дипломных работ.

1 Лабораторная работа № 11. Алгоритмический язык VBA. Разработка разветвленной программы с использованием вложенного оператора IF

Цель работы: изучение оператора многозначного ветвления ***If***; разработка программы, реализующей выбор из нескольких альтернатив (более 2).

1.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.
2 Выполнить задание, полученное у преподавателя.
3 Оформить отчет в виде набора файлов.

1.2 Основные теоретические положения

Схема алгоритма конструкции оператора многозначных ветвлений ***If*** представлена на рисунке 1.1.

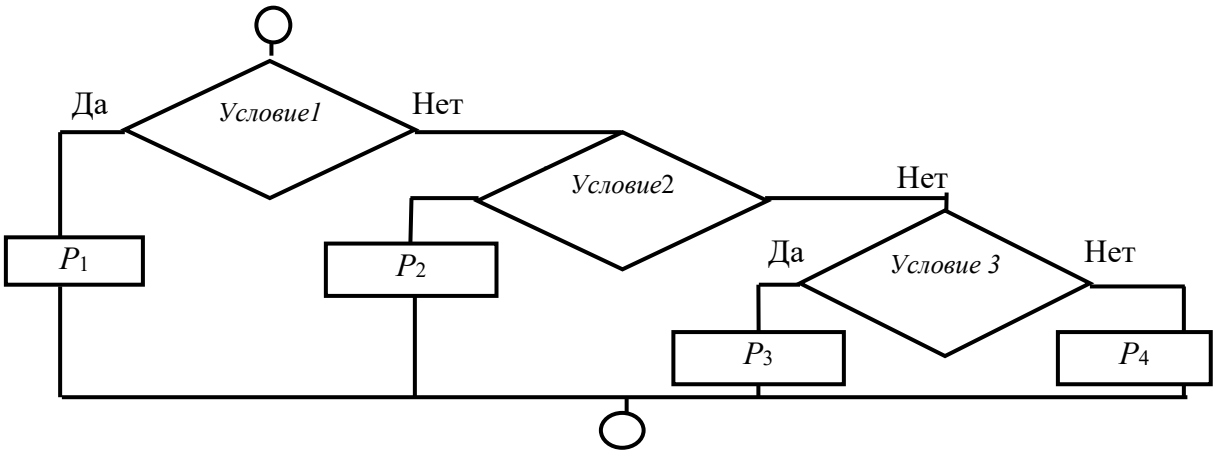


Рисунок 1.1 – Схема алгоритма конструкции оператора многозначных ветвлений ***If***

Синтаксис оператора многозначных ветвлений ***If*** в блочной (структурной) форме следующий:

If <лог.выражение1> Then
    P_1
ElseIf <лог.выражение2> Then
    P_2
Else
    P_3
End If

где *If*, *Then*, *Else*, *End If* – зарезервированные слова;
P_1, P_2, P_3 – операторы.
Алгоритм работы такой конструкции:
1) если логическое выражение **1** истинно, то выполняется оператор $\boldsymbol{P_1}$ (или блок операторов), следующий за конструкцией ***Then***, а остальные операторы

пропускаются;

2) если логическое выражение **1** ложно, то оператор P_1 пропускается и анализируется логическое выражение **2**, следующее за ***ElseIf***. Если оно истинно, то выполняется оператор P_2 (или блок операторов), следующий за ***Then***, а остальные операторы пропускаются;

3) оператор P_3 (или блок операторов), следующий за последним ***Else***, выполняется лишь в том случае, если ложны все логические выражения.

Далее представлены примеры использования условного оператора ***If*** и многозначного ветвления, позволяющего определить возможность поступления абитуриента в учреждения образования. На первом шаге происходит ввод пользователем балла, полученного на тестировании. На втором шаге, если полученный балл более 60, приложение выдает сообщение: «Вам можно поступать в ВУЗ». На третьем шаге, если полученный балл более 20, приложение выдает сообщение: «Вам можно поступать в СУЗ». На четвертом шаге, если полученный балл более 5, приложение выдает сообщение: «Вам можно поступать в ПТУ».

```
Sub ball()
Dim ball As Variant
ball = InputBox(«Введите полученный Вами балл на тестировании»)
If ball>=60 Then
MsgBox («Вам можно поступать в ВУЗ»)
ElseIf ball>=20 Then
        MsgBox («Вам можно поступать в СУЗ»)
ElseIf ball>=5 Then
        MsgBox («Вам можно поступать в ПТУ»)
End If
```

Задание

1 Запросить у пользователя ввод целого числа от 0 до 100 включительно – оценка по 100-балльной системе.

2 С применением оператора многозначных ветвлений ***If*** осуществить преобразование введенной оценки по 100-балльной системе в 5-балльную по шкале, предварительно созданной на листе Excel (таблица 1.1).

Таблица 1.1 – Преобразование оценок

Значение оценки по 100-балльной системе	Значение оценки по 5-балльной системе
От 0 до 20	Оценка 1
От 20 до 40	Оценка 2
От 40 до 60	Оценка 3
От 60 до 80	Оценка 4
От 80 до 100 включ.	Оценка 5
Меньше 0 или больше 100	Ошибка ввода данных

3 По результатам вывести сообщение с введенной оценкой по 100-балльной системе и полученной оценкой по 5-балльной системе либо сообщение об ошибке.

Контрольные вопросы

1 В каких случаях в программе используется вложенный условный оператор?
2 Сколько инструкций ElseIf может быть в блочном варианте оператора ветвления?
3 Нарисуйте блок-схему алгоритма выполнения вложенного условного оператора.

2 Лабораторная работа № 12. Алгоритмический язык VBA. Разработка разветвленной программы с использованием оператора выбора

Цель работы: ознакомление с оператором выбора ***SelectCase*** и закрепление полученных знаний на практике.

2.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.
2 Выполнить задание, полученное у преподавателя.
3 Оформить отчет в виде набора файлов.

2.2 Основные теоретические положения

При наличии большого количества ветвлений конструкция многозначных ветвлений ***If*** становится тяжёлой для восприятия. В подобных случаях хорошей альтернативой оператору ***If*** служит оператор выбора ***SelectCase***, который позволяет выбрать одно из нескольких возможных продолжений программы.

В то время как ***If...Then...Else*** для каждой инструкции ***ElseIf*** оценивает разные выражения, инструкция ***SelectCase*** оценивает выражение только один раз, в начале управляющей структуры. ***SelectCase*** выполняет одну из нескольких групп инструкций в зависимости от значения выражения. Синтаксис:

SelectCase<*выражение*>
*[**Case** <списокВыражений-n>*
[инструкции-n]] ...
*[**CaseElse***
[инструкции_else]]
EndSelect

где ***<выражение>*** − обязательный, любое числовое выражение или строковое выражение; ***<списокВыражений-n>*** – обязательный при наличии предложения

Case. Список с разделителями, состоящий из одной или нескольких форм следующего вида:

а) ***<инструкции-n>*** − необязательный. Одна или несколько инструкций, выполняемых в том случае, если выражение совпадает с любым компонентом списка <***списокВыражений-n***>;

б) ***<инструкции_else***> − необязательный. Одна или несколько инструкций, выполняемых в том случае, если выражение не совпадает ни с одним из предложений ***Case***.

Синтаксис оператора ***SelectCase*** также может содержать следующие элементы:

– ***выражение To выражение***;

– ***Is*** оператор сравнения ***выражение***.

Ключевое слово ***To*** задает диапазон значений. При использовании ключевого слова ***To*** перед ним должно находиться меньшее значение. Ключевое слово ***Is*** с операторами сравнения (кроме ***Is*** и ***Like***) задает диапазон значений. Если ключевое слово ***Is*** не указано, оно вставляется по умолчанию.

Если выражение совпадает с любым выражением из ***списка Выражений*** в предложении ***Case***, выполняются все инструкции, следующие за данным предложением ***Case*** до следующего предложения ***Case***, или для последнего предложения до инструкции ***End Select***. Затем управление передается инструкции, следующей за ***End Select***. Если выражение совпадает с выражениями из списка в нескольких предложениях ***Case***, выполняется только первый подходящий набор инструкций.

Предложение ***CaseElse*** задает список ***инструкции_else***, которые будут выполнены, если не обнаружено ни одно совпадение выражения и компонента **список Выражений** ни в одном из остальных предложений ***Case***. Хотя данное предложение не является обязательным, рекомендуется помещать предложение ***CaseElse*** в блок ***SelectCase***, чтобы предусмотреть неожиданные значения выражения. Если ни в одном предложении ***Case список Выражений*** не содержит компонента, отвечающего аргументу выражения, и отсутствует инструкция ***CaseElse***, выполнение продолжается с инструкции, следующей за инструкцией ***EndSelect***.

Пример использования оператора ***SelectCase*** представлен в таблице 2.1.

Если значение переменной ***vozrast*** меньше или равно 7, отображается сообщение «*Ты дошкольник*»; если значение находится в диапазоне от 8 до 16 – сообщение «*Ты учишься в школе*»; если в диапазоне от 17 до 30 – сообщение «*Тебе пора заняться делом*»; если в диапазоне от 31 до 60 – сообщение «*Кто не работает, тот не ест*». Если значение возраста не равно ни одному из предложенных диапазонов значений, выводится сообщение «*Вы заслужили отдых*».

Из представленного в таблице 2.1 примера видно, что код этой процедуры более прост для восприятия, чем многозначные ветвления ***If***.

Таблица 2.1 – Синтаксис и пример использования оператора ***SelectCase***

Синтаксис оператора *SelectCase*	Пример использования оператора *SelectCase*
SelectCase Ключ Выбора *Case Is выражение* *оператор* *Case диапазон значений* *оператор* *Case диапазон значений* *оператор* *Case диапазон значений* *оператор* *CaseElse* *оператор* *End Select*	*SelectCase vozrast* *Case Is <=7* *Msgbox «Ты дошкольник»* *Case 8 to 16* *Msgbox «Ты учишься в школе»* *Case 17 to 30* *Msgbox «Тебе пора заняться делом»* *Case 31 to 60* *Msgbox «Кто не работает, тот не ест»* *CaseElse* *Msgbox «Вы заслужили отдых»* *EndSelect*

Контрольные вопросы

1 В каких случаях в программе используется оператор выбора?

2 В чем преимущество оператора выбора варианта перед многовариантным оператором ветвления?

3 Когда применение оператора ***SelectCase*** эффективнее оператора ***IfThenElseEndIf***?

3 Лабораторная работа № 13. Алгоритмический язык VBA. Разработка программы с использованием оператора выбора FOR. Ввод и вывод одномерного массива

Цель работы: изучение оператора цикла ***For***.

3.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.

2 Выполнить задание, полученное у преподавателя.

3 Оформить отчет в виде набора файлов.

3.2 Основные теоретические положения

Часто в программах необходимо реализовать определённые операторы несколько раз. В этих случаях организуют циклические вычисления. Алгоритм называется ***циклическим***, если определенная последовательность шагов выполняется несколько раз в зависимости от заданных условий. Циклические алгоритмы могут быть осуществлены с применением следующих операторов цикла: ***For … Next***, ***While … Wend***, ***Do … Loop***, которые позволяют повторить группу операторов или один оператор заданное количество раз.

Общий вид алгоритма конструкции оператора цикла ***For … Next*** представлен на рисунке 3.1.

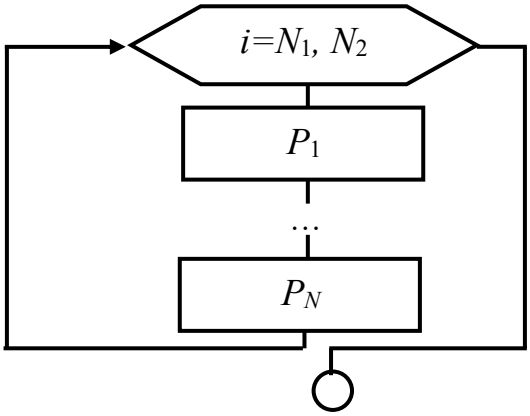


Рисунок 3.1 – Схема алгоритма конструкции оператора цикла ***For … Next***

Синтаксис конструкции оператора цикла ***For … Next*** следующий:

For i = N_1 To N_2 [Step h]
P_1
…
[Exit For]
P_N
Next i

Тело цикла

For (для), ***To*** (до), ***Step*** (шаг), ***Exit For*** (выход из ***For***), ***Next*** (следующий) – служебные слова, ***VBA***, ***P1***, ***PN*** – операторы.

Step является необязательным параметром. Если он опущен в программе, то значение параметра ***i*** увеличивается на **1**. Параметр ***Step*** может быть любым действительным числом: как целым, так и дробным, как положительным, так и отрицательным.

Оператор ***Exit For*** позволяет выйти из цикла ***For … Next*** до его завершения. Тем самым программа сможет среагировать на определённое событие, не выполняя цикл заданное число раз.

Контрольные вопросы

1 Может ли тело оператора цикла с параметром не выполниться ни разу?

2 Как должен быть оформлен оператор цикла с параметром, чтобы тело цикла выполнялось при уменьшающихся значениях параметра цикла?

3 Чему равно количество повторений тела оператора цикла с параметром, если параметр цикла принимает:

а) все целые значения от 1 до 10;

б) все значения от 10 до 100 с шагом 7;

в) все значения от а до б с шагом ***Step***.

4 Можно ли в теле оператора цикла использовать условный оператор?

5 Какие Вы знаете операторы для принудительного (преждевременного) выхода из оператора цикла?

4 Лабораторная работа № 14. Алгоритмический язык VBA. Обработка одномерного массива

Цель работы: изучение одномерных массивов; разработка программы с вводом, выводом и обработкой одномерных массивов.

4.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.
2 Выполнить задание, полученное у преподавателя.
3 Оформить отчет в виде набора файлов.

4.2 Основные теоретические положения

Массив – совокупность однотипных элементов данных (чисел, логических данных, символов), которой при обработке присвоено определенное имя. Совокупность представлена набором переменных с одним именем и с разными индексами. Массивы бывают разной размерности: одномерные, двумерные, трехмерные, …, *n*-мерные.

Массивы бывают статические и динамические. Статическими называются массивы, количество элементов в которых заранее известно и не изменяется в ходе выполнения программы. Динамические массивы – массивы, в которых либо не известно начальное количество элементов, либо размерность массива (количество элементов) изменяется при выполнении программы.

Описание массивов:

1) одномерный статический массив:

Dim<*имя массива*>**(**<*начальное значение индекса*>***To***<*конечное значение индекса*>**)** [***As***<*тип элементов массива*>]

или

Dim<*имя массива*>**(**<*количество элементов массива*>**)** [***As***<*тип элементов массива*>];

2) двумерный статический массив:

Dim<*имя массива*>**(**<*начальное значение индекса по строкам*>***To***<*конечное значение индекса по строкам* >**,**< *начальное значение индекса по столбцам*>***To***< *конечное значение индекса по столбцам*>**)** [***As***<*тип элементов массива*>]

или

Dim<*имя массива*>**(**<*количество строк*>**,**<*количество столбцов*>**)** [***As***<*тип элементов массива*>].

Первый способ отличается от второго тем, что в первом случае указывается индекс первого и последнего элементов, во втором же – только количество элементов, нумерация которых может начинаться как с 0, так и с 1. Это зависит

от опции ***Base*** (задает базовый индекс). Если опция не указана, то нумерация элементов массива начинается с нуля. Для изменения базового индекса в начале листа модуля необходимо написать ***OptionBase 1***.

Пример 1

DimA(1 To 10) As Integer – массив А состоит из 10 элементов целого типа, индексы которых 1, 2, …, 10;

Dim A(10) As Integer – массив состоит из 10 значений целого типа. Индексация зависит от опции ***Base***. Если опция не указана, то номера элементов − от 0 до 9, если же указана (т. е. вначале модуля записано ***OptionBase 1***), то номера элементов изменяются от 1 до 10;

3) динамический массив:

Dim<*имя массива*>**( )** [***As***<*тип элементов массива*>].

После определения количества элементов массива выполняется его переопределение:

ReDim<*имя массива*>**(**<задается размерность массива (одномерного/двумерного >**)**.

Пример 2

DimA( ) As Single – динамический массив А вещественных элементов $n = 7$;

ReDimA (1 Ton) – переопределение одномерного массива из n значений;

ReDimA (5,n) – переопределение двумерного динамического массива, состоящего из пяти строк и n столбцов (начало индексации элементов определяется по опции **Base**).

Обращение к элементу массива осуществляется следующим образом: указывается имя массива, а затем в круглых скобках − номер элемента в массиве. Если массив двумерный – указывается вначале номер строки, затем через запятую номер столбца.

Примеры объявления массивов:

DimA (12) As Integer ‘одномерный массив из 12 элементов типа Integer

Dim A(1 To 12) As Integer

DimB (3, 3) As Single ‘двумерный массив элементов 3 × 3 (матрица) типа Single

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

Причем по умолчанию первый элемент массива А будет А(0), а последний – А(11). В этом случае говорят, что 0 – базовый индекс. Можно изменить базовый индекс, написав в начале листа модуля инструкцию OptionBase 1. После этого индексы массивов А и В будут начинаться с единицы.

Массив в программе определяется поэлементно.

Удобным способом определения одномерных массивов является функция Array, преобразующая список элементов, разделенных запятыми, в вектор из этих значений и присваивающая их переменной тип Variant.

Задание

1 Объявить одномерный массив размерностью 10 элементов целочисленного типа.

2 Инициализировать все элементы массива произвольными числами: A(1) = 5 и т. д.

3 Реализовать расчет среднего арифметического всех чисел, содержащихся в массиве, с помощью оператора цикла.

4 Вывести результат расчета в виде текстового сообщения:

«*Среднее арифметическое*: ____».

Контрольные вопросы

1 Опишите синтаксис объявления массива с использованием оператора ***Dim***.

2 Какое значение нижнего индекса элемента массива принято в VBA по умолчанию? Каким образом можно для него задать значение 1?

3 Как установить или изменить размерность многомерного динамического массива?

4 Как изменить размерность динамического массива без потери имеющихся значений его элементов?

5 Сколько индексов характеризуют конкретный элемент двумерного массива?

5 Лабораторная работа № 15. Алгоритмический язык VBA. Разработка программы с вложенными циклами. Обработка многомерных массивов

Цель работы: изучение вложенных операторов цикла ***For***.

5.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.

2 Выполнить задание, полученное у преподавателя.

3 Оформить отчет в виде набора файлов.

5.2 Основные теоретические положения

Если телом цикла является циклическая структура, то такие циклы называются вложенными. Цикл, содержащий в себе другой цикл, – внешний, а цикл, содержащийся в теле другого цикла, – внутренний.

Синтаксис конструкции вложенных операторов цикла ***For … Next***:

For i = N_1 To N_2
For j = M_1 To M_2
P_1
... — *тело внутреннего цикла*
P_N
Next j
Next i — *тело внешнего цикла*

Общий вид алгоритма конструкции вложенных операторов цикла ***For … Next*** представлен на рисунке 5.1.

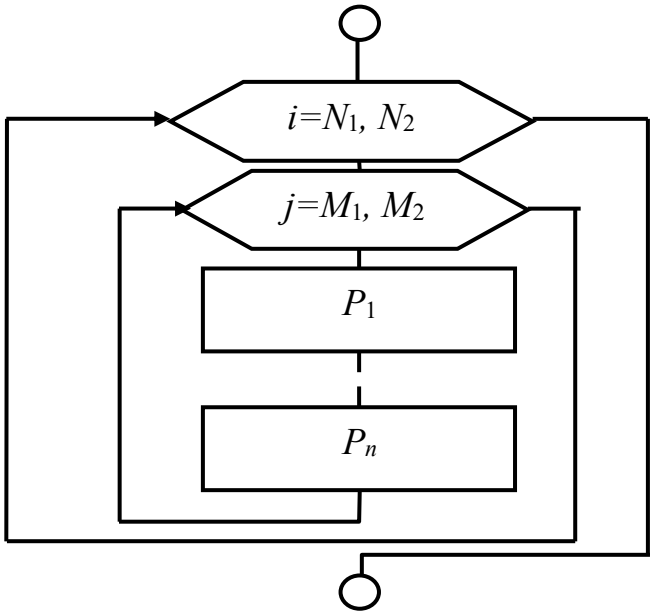


Рисунок 5.1 – Схема алгоритма конструкции вложенных операторов цикла ***For … Next***

При первом вхождении в цикл параметр внешнего цикла ***i*** принимает значение, равное N_1, и управление передаётся во внутренний цикл, в котором параметр цикла ***j*** принимает значение, равное M_1, и выполняется оператор (операторы), который записан во внутреннем цикле. Затем параметр внутреннего цикла ***j*** увеличивается на **1** и вновь повторяется тело цикла.

Операторы P_1, P_N будут реализовываться до тех пор, пока параметр цикла ***j*** не станет больше величины M_2. Затем параметр внешнего цикла ***i*** увеличивается на **1** и вновь начинает свою работу внутренний цикл, в котором параметр цикла ***j*** будет изменяться от M_1 до M_2, и при каждом прохождении цикла будут выполняться операторы P_1 и P_N. Внешний цикл закончит свою работу, когда параметр цикла ***i*** станет больше величины N_2.

Задание

С помощью вложенных операторов цикла ***For … Next*** реализовать запись на рабочий лист ***Excel*** матрицы размерностью $n \times n$ (значение числа n запро-

сить у пользователя), элементы которой должны являться произведением номера строки на номер колонки.

Контрольные вопросы

1 В каких случаях используются вложенные операторы цикла?

2 Как оформляются вложенные операторы цикла?

3 Нарисуйте алгоритмическую схему выполнения вложенных операторов цикла.

4 Вложенный цикл образован двумя операторами цикла с параметром. Что является телом?

5 Может ли внешний оператор вложенного цикла:

а) не выполниться ни разу;

б) выполняться бесконечное число раз (или до того момента, когда пользователь прервет его выполнение)?

6 Лабораторная работа № 16. Алгоритмический язык VBA. Разработка программы с использованием циклов While

Цель работы: получение навыков применения операторов цикла с предусловием и постусловием.

6.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.

2 Выполнить задание, полученное у преподавателя.

3 Оформить отчет в виде набора файлов.

6.2 Основные теоретические положения

Циклы данного вида используются, когда заранее неизвестно, сколько раз будет выполняться тело цикла.

Циклы с предусловием (DoWhile … Loop, DoUntil … Loop) представлены в таблице 6.1, а операторы циклов с постусловием (Do … LoopWhile, Do … LoopUntil) – в таблице 6.2.

Отличие циклов с предусловием от циклов с постусловием заключается в том, что тело цикла первых может не выполниться ни разу, в то время как тело цикла с постусловием всегда выполнится хотя бы один раз.

Таблица 6.1 – Циклы с предусловием

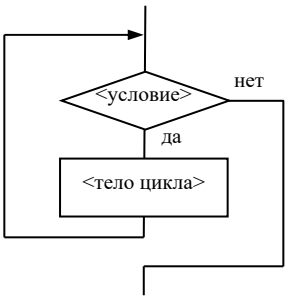
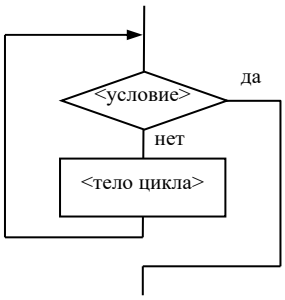
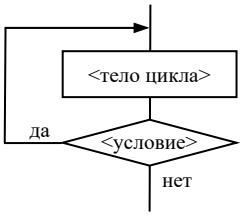
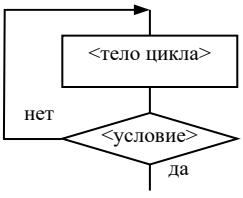
Синтаксис	**DoWhile**<*условие*> <*тело цикла*> [**ExitDo**] … ***Loop***	**DoUnlil**<*условие*> <*тело цикла*> [**ExitDo**] … ***Loop***
Порядок выполнения	**<***Тело цикла***>** будет выполняться в том случае, когда <*условие*> имеет значение Истина (TRUE) (цикл продолжается при истинном значении <*условия***>**). Если <*условие*> ложно (FALSE), то выполняются операторы, стоящие за циклом. В первом случае есть возможность досрочного выхода из цикла (это реализовано через **ExitDo)**	**<***Тело цикла***>** выполняется до тех пор, пока <*условие*> не примет значение Истина (цикл продолжается при ложном значении <*условия*>). Есть возможность досрочного выхода из цикла (это реализовано через **ExitDo**)
Изображение в блок-схемах		

Таблица 6.2 – Циклы с постусловием

Синтаксис	**Do** <*телоцикла*> [**Exit Do**] … **Loop While**<*условие*>	**Do** <*тело цикла*> [**Exit Do**] … **Loop Until <***условие*>
Порядок выполнения	<*Тело цикла*> будет выполняться в том случае, когда <*условие*> имеет значение Истина (цикл продолжается при истинном значении <*условия*>). Если <*условие*> ложно, то выполняются операторы, стоящие за циклом. Предоставлена возможность досрочного выхода из цикла (это реализовано через **ExitDo**)	**<***Тело цикла***>** выполняется до тех пор, пока <*условие*> не примет значение Истина (цикл продолжается при ложном значении <*условия*>). Есть возможность досрочного выхода из цикла (это реализовано через **ExitDo**)
Изображение в блок-схемах		

Пример − Организовать ввод последовательности целых чисел, пока их сумма не превысит целого числа m. Вывести количество введенных чисел. Текст программы с пояснениями представлен в таблице 6.3.

Таблица 6.3 – Программа по вводу последовательности целых чисел

Текст программы	Описание
Public Sub prog1()	–
Dim x As Integer, m As Integer	–
Dim s As Integer, i As Integer	–
m = InputBox(«Введите число m»)	Ввод предельного числа
i = 1	Номер вводимого числа последовательности
s =InputBox(«Введите 1 число»)	Ввод первого числа последовательности
Do While s <= m	Цикл с предусловием: тело цикла выполняется, пока условие s<=m имеет значение Истина (TRUE)
i = i + 1	Тело цикла: увеличение номера на 1
x=InputBox(«Введите» &i& «число»)	Ввод очередного (*i*-го) значения
s = s + x	Добавление введенного значения к предыдущему значению суммы
Loop	Конец тела цикла
MsgBox («Количество чисел» &i)	Вывод значения переменной *i*
EndSub	

Задание

1 В цикле реализовать запрос пароля: «Введите пароль:».

2 Сравнить введенный пользователем пароль с заданным.

3 Цикл должен выполняться до тех пор, пока пароль не совпадет.

4 Если пароль совпадает, цикл завершается и программа выдает что?

Контрольные вопросы

1 В каких случаях используются операторы цикла с условием?

2 В чём основное отличие между циклами с предусловием и с постусловием?

3 В каких случаях целесообразно использовать циклы с предусловием, циклы с постусловием и циклы по счётчику?

4 Может ли тело оператора цикла с предусловием:

а) не выполниться ни разу;

б) выполняться бесконечное число раз (или до тех пор, когда пользователь прервет его выполнение)?

7 Лабораторная работа № 17. Компьютерные презентации

Цель работы: изучение порядка построения и форматирования презентаций диаграмм в MS PowerPoint.

7.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.

2 Выполнить задание, полученное у преподавателя.

3 Оформить отчет в виде набора файлов.

7.2 Основные теоретические положения

Презентация (от англ. presentation – представление) – способ представления информации. Мультимедийная презентация создается для поддержки доклада и должна быть, прежде всего, информативной. Информация должна быть представлена в наиболее наглядной и убедительной форме. Для этого используют графику, видео и звуковую информацию. Основные идеи отражаются в текстовых фрагментах. Они обычно небольшие.

Презентации могут преследовать различные цели и их использование безгранично:

– огромна роль презентаций в рекламе для информационных функций, которая рассчитана на определенную категорию зрителей: потенциальных покупателей и заказчиков, акционеров, журналистов, читателей и т. п. Для предложения товаров и услуг, информаций;

– в научно-технической сфере для докладов (для изображения схем, формул, диаграмм и пр.);

– в учебных заведениях можно создавать и демонстрировать учебные и справочные слайд-фильмы;

– в бытовых целях с помощью PowerPoint сочинять, украшать и показывать сказки или создавать поздравления.

Существует множество программ подготовки презентаций. Хотя сами программы могут называться иначе, но по своей сути это программы подготовки презентаций.

Среди этой группы программ можно назвать следующие:

1) Microsoft PowerPoint;

2) OpenOffice.org Draw (OpenOffice) – упрощенный бесплатный аналог Microsoft PowerPoint;

3) ACTIVstudio – программа создания презентаций и электронных учебников для интерактивных досок;

4) Adobe Acrobat Pro, принтеры PDF – программы создания файлов *.pdf.

Существуют и другие программы создания презентаций.

Процесс создания презентации в Microsoft Power Point состоит из следующих этапов:

– выбор общего оформления;

– добавление новых слайдов и их содержимого;

– выбор разметки слайдов;

– изменение при необходимости оформления слайдов;

– изменение цветовой схемы;

– применение различных шаблонов оформления;

– создание эффектов анимации при демонстрации слайдов.

Можно выбрать следующие способы создания презентации.

1 Новая презентация. Позволяет создавать презентацию с помощью пустых слайдов.

2 Из шаблона оформления. Позволяет создать презентацию на основе имеющегося шаблона Microsoft Power Point, содержащего основные элементы

оформления, шрифты и цветовую схему.

3 Из мастера автосодержания. Позволяет создать презентацию на основе имеющегося шаблона оформления Microsoft Power Point, включающего основной предполагаемый текст слайдов.

4 Из имеющейся презентации. Презентация создается на основе уже имеющейся презентации с заданным оформлением.

Требования к оформлению презентации:

1) первый слайд – название;

2) общий стиль (исключение – первый слайд);

3) анимированная смена слайдов в общем стиле;

4) наличие заголовков у слайдов;

5) лаконичность (минимум текста). Каждый слайд должен быть заполнен текстом не более чем на треть;

6) на слайдах должны присутствовать объекты: аудио-(видео-)фрагменты, анимированные изображения (в том числе двигающиеся по заданной траектории);

7) постоянный шрифт;

8) крупный шрифт;

9) тёмный текст на светлом фоне или наоборот;

10) не использовать стандартный клипарт;

11) наличие четкой структуры и навигации, созданной при помощи кнопок и гиперссылок;

12) непрерывный музыкальный фон.

Запуск программы MS PowerPoint можно осуществить разными способами:

1). наиболее простой из них заключается в использовании кнопки ***Панели быстрого запуска*** или ***Рабочего стола***;

2). можно выполнить команду *Пуск – Программы – Microsoft PowerPoint*.

После запуска появляется окно программы с открытым диалоговым окном MS Power Point. В окне предлагается выбрать порядок работы по созданию презентации.

1 ***Мастер автосодержания*** можно использовать для быстрого создания презентации с типовой структурой. В этом случае на экран поступит диалоговое окно Мастера, который будет задавать вопросы. Пользуясь вашими ответами, Мастер за несколько шагов создаст «черновик» профессиональной презентации из 8–15 слайдов, который приблизительно будет соответствовать вашему замыслу. Затем эту презентацию следует отредактировать.

2 ***Шаблон*** оформления позволяет взять за основу своей презентации один из готовых шаблонов Power Point. При выборе этого раздела и нажатии кнопки **Ок**, на экране появится диалоговое окно ***Создать презентацию*** с тремя вкладками. На вкладке ***Шаблон оформления*** можно выбрать дизайн оформления слайдов.

3 ***Пустую презентацию***. Если активизировать этот раздел, то о создании своей презентации вам придется позаботиться самим.

4 ***Открыть презентацию***. Этот раздел позволяет загрузить готовую презентацию с жесткого диска или другого съемного носителя.

В левой части окна приложения (рисунок 7.1) находится область ***Структура*** или ***Слайды*** для переключения между режимами *Слайды* и *Структура*. По умолчанию в области ***Структура*** – ***Слайды*** устанавливается режим *Слайды*, т. е. отображается панель ***Слайды***. В этом режиме в этой области отображаются миниатюрные изображения слайдов, входящих в презентацию.

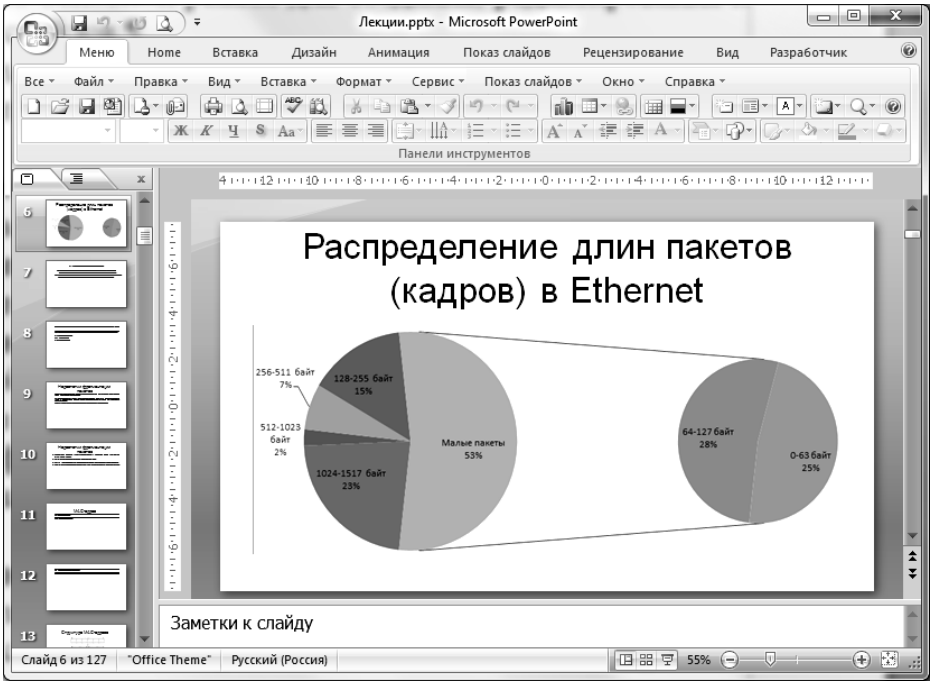


Рисунок 7.1 – Окно программы MS PowerPoint

В режиме *Структура* в этой области отображается иерархическая структура, содержащая заголовки и тексты слайдов презентации. Перед заголовком каждого слайда стоит номер и значок. Основной текст, включающий до пяти уровней отступов, расположен после каждого заголовка.

В центре приложения находится область слайда, в которой отображается слайд. Режим *Обычный* – это основной режим для создания, редактирования и форматирования отдельных слайдов.

Ниже главного окна находится область заметок. В этой области к каждому слайду можно добавить заметки докладчика, которые не отображаются в режиме показа слайдов.

Строка меню предоставляет доступ ко всем важным командам программы PowerPoint.

Панели инструментов предоставляют быстрый доступ к используемым командам. В Power Point используется группа команд меню Показ слайдов вместо меню Таблица редактора Word.

На панели форматирования размещены следующие инструменты: *Конструктор* и *Создать слайд*. При выборе кнопки *Конструктор* в области задач отображается панель ***Дизайн слайда***, в которой размещены три раздела: ***Шаблоны оформления***, ***Цветовые схемы***, ***Эффекты анимации***. С помощью команд этих разделов можно к слайду применить шаблон оформления, цветовые схемы и эффекты анимации.

При выборе на панели инструментов команды *Создать слайд* в области задач отображается панель ***Разметка слайда***, с помощью которой можно изменять разметку слайдов (***Макет текста***, ***Макет содержимого***, ***Макет текста и содержимого***).

Бегунок линии прокрутки позволяет переходить между слайдами, а не по тексту в пределах одного слайда. Кроме того, во время перетаскивания бегунка редактор показывает номер и название каждого слайда.

Кнопки режима просмотра слева от горизонтальной полосы прокрутки, позволяют быстро переключиться в один из режимов просмотра Power Point (*Обычный режим*, *Режим сортировщика слайдов*, *Показ слайдов*). В левой части строки состояния отображается номер слайда, над которым идет работа в данный момент, и тип создаваемой презентации.

Режимы просмотра.

Для эффективного применения PowerPoint при создании и редактировании презентаций необходимо использовать различные режимы просмотра документов. Режимы представляют собой разные способы отображения слайдов на экране. К основным режимам, применяемым в PowerPoint, относятся: ***Обычный режим*** и ***Режим сортировщика слайдов***.

Переключение режимов отображения можно осуществлять в меню ***Вид*** (***Обычный***, ***Сортировщик слайдов***, ***Показ слайдов***, ***Страницы заметок***). Переключение режимов можно также осуществлять с помощью кнопок, расположенных слева от горизонтальной полосы прокрутки (***Обычный режим***, Р***ежим сортировщика слайдов***, ***Показ слайдов***).

Режимы отображения слайдов.

Режим ***Обычный***. В этом режиме в окне приложения отображаются три области: ***Структура – Слайды***, ***область Слайда***, ***Заметки к слайду***. Размеры областей можно изменять, перетаскивая их границы.

Режим ***Сортировщик слайдов*** – это режим, в котором все слайды презентации отображаются виде миниатюр. В этом режиме можно легко перемещать слайды, изменяя порядок их следования в презентации.

Режим ***Показ слайдов*** – это режим, с помощью которого можно просмотреть презентацию на экране.

Режим ***Страницы заметок*** – режим просмотра, в котором к каждому из слайдов можно добавить заметки докладчика. В верхней половине страницы появляется уменьшенное изображение слайда, а в нижней половине отображается большая панель для текста заметок.

Создание презентации.

Откроем программу и начнем создавать презентацию. Начнем с фона. Нужно выбрать вверху кнопку ***Формат*** – ***Оформление слайда***, затем на панели справа, выбрать ***Тему***, которую Вы желаете увидеть в своей презентации (на всех слайдах или на одном слайде); или можно выбрать ***Формат*** – ***Фон***, и выбрать заливку или текстуру.

Далее, нужно выбрать ***Формат*** – ***Разметка слайда***, затем справа выбрать понравившийся шаблон. Это нужно для того, чтобы было легче и быстрее вставлять текст, картинки, видеоролики и т. д.

Намного лучше будет, если презентацию «оживить». Для этого применяйте больше анимации. Представьте картину: заголовок презентации красиво меняет свой цвет, в центре начинает высвечиваться текст, в верхнем правом углу кружатся звездочки, внизу текст начинает менять форму и т. д. Для этого требуется только выделить объект (текст, картинку и т. д.). Затем выбрать ***Показ слайдов-Настройка анимации*** и справа ***Добавить эффект***.

Далее выбираете анимацию. *Пути перемещения* – Вы рисуете линию, по которой объект будет перемещаться. *Выделение* – объект будет выделяться, только недолго. Если вы хотите, чтобы объект выделялся или перемещался бесконечно, то нужно:

1) выбрать анимацию;
2) справа выбрать название объекта (в списке);
3) кликнуть по нему и выбрать *Время*;
4) там, где *Повторение*, найти *До окончания слайда*;
5) **Ок**.

Слайд – это «страница» документа. Слайды отображаются в левом окне. Можно создать новый слайд, путем ***Вставка*** – ***Создать слайд***. Во время показа презентации, слайды будут сменяться после того, как пользователь будет кликать мышью. А у вас может произойти одна проблема: если в презентации много слайдов и анимации; во время показа, какой-то неопытный пользователь не вовремя кликнет мышью, и анимация испортится, остановится. Чтобы такого не случилось, можно презентацию превратить в «программу».

Для этого следует выбрать ***Показ слайдов*** – ***Настройка презентации*** и поставить галочку рядом с ***Автоматический*** (полный экран). Далее, презентация никак не будет реагировать на клики мыши, а это значит, что слайды не смогут сменяться и презентация не закроется.

Гиперссылка – это ссылка на другой слайд, документ или сайт в сети Интернет. Когда вы «заблокировали» презентацию, Вы должны вставить «кнопки». ***Показ слайдов*** – ***управляющие кнопки*** и выбирайте кнопку. Можете просто кликнуть правой кнопкой по объекту и выбрать ***Настройка действия***, затем сверху в окне выбрать место, куда хотите перейти (например: следующий слайд, предыдущий слайд, завершить показ и т. д.).

Контрольные вопросы

1 Этапы создания презентации.
2 Способы создания презентации.
3 Требования к оформлению презентации.
4 Порядок работы по созданию презентации.
5 Режимы просмотра презентации.

8 Лабораторная работа № 18. Основы компьютерных сетей

Цель работы: изучение основ компьютерных сетей.

8.1 Порядок выполнения работы

1 Ознакомиться с основными теоретическими положениями.
2 Выполнить задание, полученное у преподавателя.
3 Оформить отчет в виде набора файлов.

8.2 Основные теоретические положения

8.2.1 Сетевой уровень модели OSI.

Сетевой уровень модели OSI отвечает за возможность доставки пакетов по сети передачи данных – совокупности сегментов сети (подсетей), объединенных в единую сеть любой сложности посредством узлов связи (шлюзов), в которой имеется возможность достижения из любой точки сети в любую другую.

В связи с необходимостью перенаправлять пакеты из одного сегмента сети в другой, сетевые адреса должны удовлетворять следующим требованиям:

– адреса должны быть уникальны. В сети не может быть нескольких участников с одинаковыми адресами во избежание неоднозначности;
– сетевой адрес должен содержать информацию о том, как достичь получателя по сети.

Это приводит к структурности адреса – адрес разбивается на части, позволяющие определить местоположение участника внутри сети.

Структура может быть сложной многоуровневой, например адрес содержит информацию о стране, области, населенном пункте, предприятии, здании, отделе и т. д., или простой, содержащей номер сети и номер компьютера в сети.

По сложной структуре легче построить маршрут прохождения пакета, но адрес оказывается сложным и перегруженным часто ненужной информацией. Примером такой адресации может служить доменная адресация в Интернет. По адресу `poit.bru.by` нетрудно понять, где находится данный участник сети и как до него добраться.

Простая структура позволяет значительно сократить размер адреса и сохраняет возможность работы в сети любой структуры, но для этого могут потребоваться сложные и, часто, не столь очевидные алгоритмы, как в предыдущем случае.

Следовательно, адрес получателя должен содержать в себе:

– номер (адрес) подсети;
– номер (адрес) участника (хоста) внутри подсети.

8.2.2 Протокол IP.

Архитектуру сетевого уровня удобно рассматривать на примере сетевого протокола IP (Internet Protocol) – самого распространенного в настоящее время протокола сети Интернет. Термин «стек протоколов TCP/IP» означает «набор протоколов, связанных с IP и TCP (протоколом транспортного уровня)».

Архитектура протоколов TCP/IP предназначена для объединенной сети, состоящей из соединенных друг с другом шлюзами отдельных разнородных пакетных подсетей, к которым подключаются хосты.

Каждая из подсетей работает в соответствии со своими специфическими требованиями и имеет свою природу средств связи. Однако предполагается, что каждая подсеть может принять пакет информации (данные с соответствующим сетевым заголовком) и доставить его по указанному адресу в этой конкретной подсети.

Не требуется, чтобы подсеть гарантировала обязательную доставку пакетов и имела надежный сквозной протокол.

Таким образом, два хоста, подключенные к одной подсети, могут обмениваться пакетами.

Когда необходимо передать пакет между хостами, подключенными к разным подсетям, то хост-отправитель посылает пакет в соответствующий шлюз (шлюз подключен к подсети также как обычный хост). Оттуда пакет направляется по определенному маршруту через систему шлюзов и подсетей, пока не достигнет шлюза, подключенного к той же подсети, что и хост-получатель: там пакет направляется к получателю.

IP-адреса представляют собой 32-разрядные двоичные числа. Для удобства их записывают в виде четырех десятичных чисел, разделенных точками. Каждое число является десятичным эквивалентом соответствующего байта адреса (для удобства записываются точки и в двоичном изображении).

192.168.200.47

является десятичным эквивалентом двоичного адреса

11000000.10101000.11001000.00101111.

Иногда применяют десятичное значение IP-адреса. Его легко вычислить:

192*2563+168*2562+200*256+47=3232286767

или с помощью метода Горнера:

(((192*256)+168)*256+200)*256+47=3232286767.

IP-адрес состоит из префикса – сетевой части, общей для всех узлов данной сети (адреса подсети), и суффикса – хост-части, уникальной для каждого хоста.

Количество разрядов адреса подсети (префикса) может быть различным и определяется маской сети, которая также является 32-разрядным двоичным числом. Разряды маски имеют следующий смысл:

– если разряд маски равен 1, то соответствующий разряд адреса является разрядом адреса подсети;

– если разряд маски равен 0, то соответствующий разряд адреса является разрядом адреса хоста внутри подсети.

Все единичные разряды маски (если они есть) находятся в старшей (левой) части маски и соответствуют префиксу, а нулевые (если они есть) – в правой (младшей) и соответствуют суффиксу – локальной хост-части.

Исходя из вышесказанного, маску часто записывают в виде числа единиц, в ней содержащихся.

255.255.248.0 (11111111.11111111.11111000.00000000) – является правильной

маской подсети (/21), а 255.255.250.0 (11111111.11111111.11111010.00000000) – является неправильной, недопустимой.

Перед ненулевым байтом маски могут быть только байты со значением 255, после байта, отличного от 255, – только нули.

Нетрудно увидеть, что максимальный размер подсети может быть только степенью двойки (двойку надо возвести в степень, равную количеству нулей в маске – значению суффикса).

$S = 2^{(32-P)}$, где P – длина префикса.

При передаче пакетов используются правила маршрутизации, главное из которых звучит так: «Пакеты участникам своей подсети доставляются напрямую, а остальным – по другим правилам маршрутизации».

Таким образом, прежде чем отправлять пакет, требуется определить, является ли получатель членом подсети или нет.

8.2.3 Определение диапазона адресов подсети.

Определение диапазона адресов подсети можно произвести из определения понятия маски:

– те разряды, которые относятся к адресу подсети, у всех хостов подсети должны быть одинаковы;

– адреса хостов в подсети могут быть любыми.

То есть, если наш адрес 192.168.200.47 и маска равна /20, то диапазон можно посчитать:

11000000.10101000.11001000.00101111 – адрес;

11111111.11111111.11110000.00000000 – маска;

11000000.10101000.1100XXXX.XXXXXXXX – диапазон адресов,

где 0,1 – определенные значения разрядов;

x – любое значение,

Что приводит к диапазону адресов:

от 11000000.10101000.11000000.00000000 (192.168.192.0)

до 11000000.10101000.11001111.11111111 (192.168.207.255).

Следует учитывать, что некоторые адреса являются запрещенными или служебными и их нельзя использовать для адресов хостов или подсетей. Это адреса, содержащие:

0 – в первом или последнем байте;

255 – в любом байте (это широковещательные адреса);

127 – в первом байте (внутренняя петля – этот адрес имеется в каждом хосте и служит для связывания компонентов сетевого уровня).

Поэтому доступный диапазон адресов будет несколько меньше.

Количество допустимых адресов хостов в (под)сети (с учетом запрещения крайних значений адреса) определяется по формуле

$N = 2^{(32-P)} – 2$, где P – длина префикса.

Префиксы длиной 31 или 32 бит непригодны для употребления, префикс длиной 30 бит позволяет адресовать только два хоста (используется при двухточечных соединениях по PPP). Адресом (под)сети можно считать адрес любого ее хоста с обнуленными битами хост-части.

Также надо учитывать, что диапазоны адресов:

10.X.X.X – для больших локальных сетей;

172.16.X.X – для больших локальных сетей, но применяется реже;

192.168.X.X – для маленьких (небольших) локальных сетей,

не могут быть использованы в сети Интернет, т. к. отданы для использования в сетях, непосредственно не подключенных к глобальной сети.

Контрольные вопросы

1 За что отвечает сетевой уровень модели OSI?

2 Требования к сетевым адресам.

3 Структура IP-адреса.

4 Что определяет маска сети?

Список литературы

1 **Гвоздева, В. А.** Информатика, автоматизированные технологии и системы: учебник / В. А. Гвоздева. – Москва: ФОРУМ; ИНФА-М, 2022. – 542 с.

2 **Гуриков, С. Р.** Информатика: учебник / С. Р. Гуриков. – 2-е изд., перераб. и доп. – Москва: ФОРУМ; ИНФА-М, 2022. – 630 с.

3 Информатика. Базовый курс: учебное пособие / Под ред. С. В. Симоновича. – 3-е изд. – Санкт-Петербург: Питер, 2017. – 640 с.: ил.

4 **Гуриков, С. Р.** Интернет-технологии: учебное пособие / С. Р. Гуриков. – Москва: ФОРУМ; ИНФРА-М, 2017. – 184 с.

5 **Гуриков, С. Р.** Введение в программирование на языке Visual Basic for Applications (VBA): учебное пособие / С. Р. Гуриков. – Москва: ИНФРА-М, 2020. – 317 с.

6 **Скитер, Н. Н.** Информационные технологии: учебное пособие / Н. Н. Скитер, А. В. Костикова. – Волгоград: ВолгГТУ, 2019. – 96 с.