ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Методические рекомендации к лабораторным работам для студентов направления подготовки 12.03.04 «Биотехнические системы и технологии» дневной формы обучения

Часть 2

Могилев 2018

УДК 004.4
ББК 32.97
И 56

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «11» сентября 2018 г., протокол № 3

Составитель ст. преподаватель Н. В. Выговская

Рецензент Ю. С. Романович

Методические рекомендации предназначены к лабораторным работам для студентов направления подготовки 12.03.04 «Биотехнические системы и технологии» дневной формы обучения по дисциплине «Информационные технологии».

Учебно-методическое издание

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Часть 2

Ответственный за выпуск	А. И. Якимов
Технический редактор	А. А. Подошевко
Компьютерная верстка	М. М. Дударева

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 36 экз. Заказ №

Издатель и полиграфическое исполнение:
Государственное учреждение высшего профессионального образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 24.01.2014.
Пр. Мира, 43, 212000, Могилев.

Содержание

Введение 4
1 Лабораторная работа № 22. Разработка схем алгоритмов для решения задач 5
2 Лабораторная работа № 23. Работа с главным меню системы C# в Visual Studio. Форматированный ввод-вывод информации. Программирование линейных алгоритмов. Работа с отладчиком..... 9
3 Лабораторная работа № 24. Программирование разветвляющихся алгоритмов. Оператор if. Программирование с использованием оператора switch 11
4 Лабораторная работа № 25. Оператор цикла for. Операторы цикла while и do while 17
5 Лабораторная работа № 26. Обработка одномерных массивов. Сортировка массивов 23
6 Лабораторная работа № 27. Обработка двухмерных массивов 29
7 Лабораторная работа № 28. Строковые типы. Обработка текстов и строк 31
8 Лабораторная работа № 29. Понятие класса 33
9 Лабораторная работа № 30. Разработка классов по индивидуальным вариантам 35
10 Лабораторная работа № 31. Работа с файлами на C#..... 36
11 Лабораторные работы № 32–33. Элементы форм WinForms – основные компоненты. Разработка приложений с формой. WinForms – дополнительные компоненты. Разработка приложений с формой и дополнительными элементами..... 37
12 Лабораторные работы № 34–35. Проектирование таблиц. Создание, связывание и заполнение таблиц базы данных 41
13 Лабораторная работа № 36. Разработка форм для работы с базой данных 42
14 Лабораторная работа № 37. Создание запросов по базе данных 42
15 Лабораторная работа № 38. Создание отчетов по базе данных 43
Список литературы.....43

Введение

Цель методических рекомендаций к выполнению лабораторных работ по дисциплине «Информационные технологии» заключается в овладении студентами практических навыков работы на персональном компьютере с использованием современных технологий. Во второй части рассмотрены лабораторные работы для изучения языка программирования C# и лабораторные работы для закрепления темы «СУБД Microsoft Access» и выполнения курсового проекта.

Студент, изучивший дисциплину, научится приемам применения технологии программирования для решения задач автоматизации обработки информации.

Рассматриваемый в методических рекомендациях материал нацелен на приобретение навыков разработки алгоритмов и программ на языке C#. Приведены краткие теоретические сведения по алгоритмам и базовым конструкциям языка программирования C#.

Методические рекомендации предназначены для выполнения лабораторных работ в компьютерном классе, а также для получения практических навыков разработки приложений в среде Visual Studio и с базами данных в MS ACCESS.

После выполнения каждой лабораторной работы студент оформляет отчет. Отчет по лабораторной работе выполняется на листах формата А4. В состав отчета входят:

– титульный лист;

– цель работы;

– текст индивидуального задания;

– результаты индивидуального задания и код программы.

Полученные при изучении дисциплины знания и навыки могут быть востребованы при курсовом проектировании и в дальнейшем процессе обучения студента в вузе.

1 Лабораторная работа № 22. Разработка схем алгоритмов для решения задач

Цель работы: программирование базовых конструкций алгоритмов; получение практических навыков по работе с алгоритмами.

1.1 Краткие теоретические сведения

Происхождение понятия алгоритма связано с именем великого среднеазиатского ученого Аль Хорезми, жившего в IX в. н. э. Им были сформулированы впервые правила выполнения четырех арифметических действий.

Алгоритм – это точная инструкция, а инструкции встречаются во всех областях человеческой деятельности. Однако не всякую инструкцию можно назвать алгоритмом. Алгоритм применяется к искомому набору исходных величин, называемых аргументами.

Цель исполнения алгоритма – получение определенного результата; если в результате исполнения алгоритма не достигнута определенная цель, значит, алгоритм либо неверен, либо не завершен.

Алгоритм – это метод (способ) решения задачи, записанный по определенным правилам, обеспечивающим однозначность его понимания и механического исполнения при всех значениях исходных данных за конечное число шагов.

Или более коротко: **алгоритм** – это строго определенная последовательность действий, необходимых для решения данной задачи.

Способы записи алгоритмов

Для записи алгоритмов используют самые разнообразные средства. Выбор средства определяется типом исполняемого алгоритма. Выделяют следующие основные способы записи алгоритмов:

– *вербальный*, когда алгоритм описывается на человеческом языке;

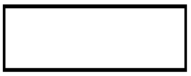
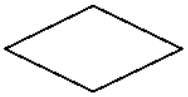
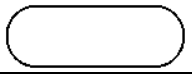
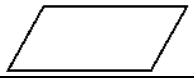
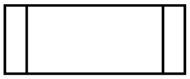
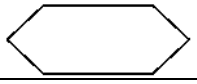
– *символьный*, когда алгоритм описывается с помощью набора символов;

– *графический*, когда алгоритм описывается с помощью набора графических изображений.

Общепринятыми способами записи являются графическая запись с помощью блок-схем и символьная запись с помощью какого-либо алгоритмического языка.

Описание алгоритма с помощью блок схем осуществляется рисованием последовательности геометрических фигур, каждая из которых подразумевает выполнение определенного действия алгоритма. Далее приводятся изображения символов схемы программы по ГОСТ 19.701–90. Внешний вид основных блоков, применяемых при написании блок–схем, приведен в таблице 1.1, а пример составления алгоритма с использованием этих блоков представлен на рисунке 1.1.

Таблица 1.1 – Графические изображения элементов схем алгоритмов

Изображение	Наименование этапа
	Символ «Процесс». Предназначен для изображения вычислительного действия или последовательности вычислительных действий, которые указываются внутри блока
	Символ «Решение». Предназначен для проверки условия и передачи управления в соответствии с ним. Внутри блока должен быть указан вопрос, решение, условие или сравнение
	Символ «Терминатор»
	Символ «Данные (ввод/вывод)»
	Символ предопределенного процесса (вызова подпрограммы)
	Символ «Подготовка (модификация)»

Основные алгоритмические конструкции: линейный алгоритм; разветвляющийся алгоритм; циклический алгоритм; вспомогательный алгоритм.

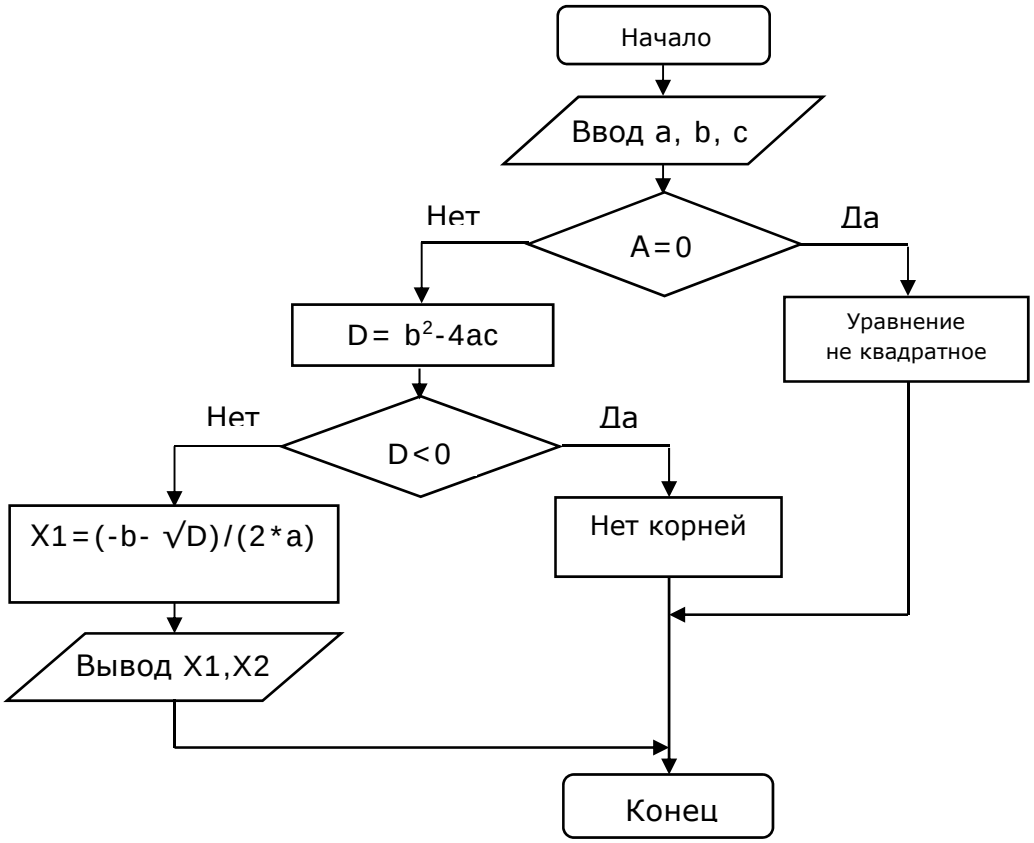


Рисунок 1.1 – Пример составления алгоритма

Задание 1

Составить блок-схему алгоритма решения задачи с условным переходом. Рассчитать значение искомой переменной по одному из двух альтернативных выражений в зависимости от переменной условия, значение которой необходимо предварительно вычислить согласно заданию. Значения переменной условия и переменной результата должны выводиться на экран. Исходные данные к заданию находятся в таблице 1.2.

Таблица 1.2 – Индивидуальные задания

Номер варианта	Данные 1	Данные 2	Переменная условия	Условие выполняется	Условие не выполняется
1	$A_1 = 2,35$	$A_2 = -5,89$	$B = (A_1^2 - A_2) > 0$	$C = 3,1B - A_1A_2^2$	$C = A_1^2A_2 - 3,1B$
2	$X_1 = -8,44$	$X_2 = 1,73$	$Y = (4X_1^2 - X_2) < 0$	$Z = 5Y^2 - X_2 + 2X_1$	$Z = X_1 + X_2 - 5Y^2$
3	$E_1 = 7,54$	$E_1 = -3,62$	$F = (3E_1 - E_1) > 0$	$Y = 4F/(E_1 - E_2)$	$Y = 4F/(E_2 - E_1)$
4	$B_1 = -6,71$	$B_1 = 4,57$	$E = (B_1 - B_2^2) < 0$	$X = 2E - B_1 + B_2^2$	$X = 2E - B_2 + B_1^2$
5	$C_1 = 3,26$	$C_1 = -5,1$	$X = (C_1 C_2) > 0$	$E = 7,5X - C_1/C_2$	$E = C_1/C_2 - 7,5X$
6	$D_1 = -9,08$	$D_1 = 6,35$	$I = (D_1^2 - D_2^2) < 0$	$B = 2,1D1 - 3,4$	$B = 1.2D_2 - 3,4$
7	$F_1 = 5,12$	$F_1 = -2,06$	$A = (H_1/H_2^2) < 0$	$D = 3,1A - 7E_1E_2$	$D = 7F_1F_2 - 3,1A$

Задание 2

Составить блок-схему алгоритма решения циклической задачи вычисления значения функции в зависимости от значения переменной аргумента. Значения переменной аргумента должны изменяться от начального до конечного значения с заданным шагом изменения. Исходные данные к заданию находятся в таблице 1.3.

Таблица 1.3 – Индивидуальные задания

Номер варианта	Начальное значение	Конечное значение	Значение шага	Выражение для расчета значений функции
1	$Xn = -0,35$	$Xk = 1,75$	$Hx = 0,05$	$Y = 2,23\ X^2 + 3,12\ X - 1,95$
2	$Yn = -2,20$	$Yk = 2,30$	$Hy = 0,10$	$Z = 1,25\ Y^2 + 0,46\ Y - 0,88$
3	$Zn = -5,25$	$Zk = 4,75$	$Hz = 0,25$	$A = 2,85\ Z^2 - 4,15\ Z + 6,05$
4	$An = 10,50$	$Ak = 30,50$	$Ha = 0,50$	$B = -1,45\ A^2 + 9,15\ A + 12,5$
5	$Bn = -4,20$	$Bk = 3,80$	$Hb = 0,20$	$C = 0,52\ B^2 + 2,52\ B + 7,84$
6	$Cn = -3,50$	$Ck = 11,50$	$Hc = 0,35$	$D = -0,15\ C^2 - 5,33\ C - 9,21$
7	$Dn = -3,60$	$Dk = 12,40$	$Hd = 0,40$	$E = 1,08\ D^2 - 7,22\ D - 2,43$

Задание 3

Составить блок-схему алгоритма решения приведенной задачи.

1 Даны три целых числа. Найти количество положительных чисел в исходном наборе.

2 Даны три целых числа. Найти количество положительных и количество отрицательных чисел в исходном наборе.

3 Даны две переменные вещественного типа: A, B. Перераспределить значения данных переменных так, чтобы в A оказалось меньшее из значений, а в B – большее. Вывести новые значения переменных A и B.

4 Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной сумму этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

5 Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной большее из этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

6 Даны три числа. Найти наименьшее из них.

7 Даны три числа. Найти среднее из них (то есть число, расположенное между наименьшим и наибольшим).

8 Даны три числа. Вывести вначале наименьшее, а затем наибольшее из данных чисел.

9 Даны три числа. Найти сумму двух наибольших из них.

10 Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию, то удвоить их; в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных A, B, C.

Контрольные вопросы

1 Кто и когда впервые ввел понятие алгоритма?
2 Каковы способы записи алгоритмов?
3 В чем заключаются основные свойства алгоритма?
4 Перечислите основные алгоритмические структуры и опишите их.
5 Каковы основные принципы разработки алгоритмов?
6 Чем объясняется разнообразие форм записи алгоритмов?
7 Охарактеризуйте словесно-пошаговый способ записи алгоритмов.
8 Охарактеризуйте табличную форму записи алгоритмов.
9 Что такое результат выполнения алгоритма?
10 Что такое исходные данные?
11 Что представляет собой графическая форма записи алгоритма?
12 Каков порядок составления блок-схем?
13 Охарактеризуйте основные блоки блок-схем.

2 Лабораторная работа № 23. Работа с главным меню системы C# в Visual Studio. Форматированный ввод-вывод информации. Программирование линейных алгоритмов. Работа с отладчиком

Цель работы: изучение базовых концепций языка программирования C# и линейной структуры, а также основных принципов использования IDE MS Visual Studio.

Задание 1

Изучить структуру команд меню MS Visual Studio и создать программу, которая сразу после запуска выводит на консоль строку

```
«Hello , C# world!»
```

Листинг 1

Файл HelloApp.cs

```
using System;
namespace Hello
{
class HelloApp
{
static  void Main()
{
System.Console.WriteLine("Hello,   C# world!");
System.Console.ReadLine();
}
}
}
```

Задание 2

Разработать алгоритм и программу линейной структуры для решения задачи по своему варианту. Все входные и выходные данные в заданиях этой группы являются вещественными числами.

1 Дана сторона квадрата a. Найти его периметр P = 4·a.

2 Дана сторона квадрата a. Найти его площадь $S = a^2$.

3 Даны стороны прямоугольника a и b. Найти его площадь S = a·b и периметр P = 2·(a + b).

4 Дан диаметр окружности d. Найти ее длину L = π·d. В качестве значения π использовать 3,14.

5 Дана длина ребра куба a. Найти объем куба $V = a^3$ и площадь его поверхности $S = 6 \cdot a^2$.

6 Даны длины ребер a, b, c прямоугольного параллелепипеда. Найти его объем V = a·b·c и площадь поверхности S = 2·(a·b + b·c + a·c).

7 Найти длину окружности L и площадь круга S заданного радиуса R: L = 2·π·R, $S = \pi \cdot R^2$. В качестве значения π использовать 3,14.

8 Даны два числа a и b. Найти их среднее арифметическое: (a + b)/2.

9 Даны два неотрицательных числа a и b. Найти их среднее геометрическое, т. е. квадратный корень из их произведения: $\sqrt{a \cdot b}$.

10 Даны два ненулевых числа. Найти сумму, разность, произведение и частное их квадратов.

Задание 3

Все входные и выходные данные в заданиях этой группы являются целыми числами. Все числа, для которых указано количество цифр (двузначное число, трехзначное число и т. д.), считаются положительными.

Integer1. Дано расстояние L в сантиметрах. Используя операцию деления нацело, найти количество полных метров в нем (1 м = 100 см).

Integer2. Дана масса M в килограммах. Используя операцию деления нацело, найти количество полных тонн в ней (1 т = 1000 кг).

Integer3. Дан размер файла в байтах. Используя операцию деления нацело, найти количество полных килобайтов, которые занимает данный файл (1 килобайт = 1024 байта).

Integer4. Даны целые положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Используя операцию деления нацело, найти количество отрезков B, размещенных на отрезке A.

Integer5. Даны целые положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Используя операцию взятия остатка от деления нацело, найти длину незанятой части отрезка A.

Integer6. Дано двузначное число. Вывести вначале его левую цифру (десятки), а затем его правую цифру (единицы). Для нахождения десятков использовать операцию деления нацело, для нахождения единиц – операцию взятия остатка от деления.

Integer7. Дано двузначное число. Найти сумму и произведение его цифр.

Integer8. Дано двузначное число. Вывести число, полученное при перестановке цифр исходного числа.

Integer9. Дано трехзначное число. Используя одну операцию деления нацело, вывести первую цифру данного числа (сотни).

Integer10. Дано трехзначное число. Вывести вначале его последнюю цифру (единицы), а затем его среднюю цифру (десятки).

Контрольные вопросы

1 Как создать новый проект консольного типа в интегрированной системе разработки Microsoft Visual Studio?

2 Охарактеризуйте систему типов данных языка C#.

3 Как выполнить объявление переменных?

4 Какие Вы знаете арифметические и логические операции языка, операции сравнения C#.?

5 Назовите приоритеты операций языка C#..

6 Приведите синтаксис оператора присваивания языка C#.

3 Лабораторная работа № 24. Программирование разветвляющихся алгоритмов. Оператор if. Программирование с использованием оператора switch

Цель работы: изучение разветвляющейся структуры языка программирования C# и конструкций if и switch.

3.1 Краткие теоретические сведения

Операторы выбора.

Как в C++ и других языках программирования, в языке C# для выбора одной из нескольких возможностей используются две конструкции − if и switch. Первую из них обычно называют альтернативным выбором, вторую − разбором случаев.

Оператор if.

Начнем с синтаксиса оператора if:

```
if(выражение_1) оператор_1
else if(выражение_2) оператор_2
...
else if(выражение_K) оператор_K
else оператор_N
```

Какие особенности синтаксиса следует отметить? Выражения if должны заключаться в круглые скобки и быть булевого типа. Точнее, выражения должны давать значения true или false. Арифметический тип не имеет явных или неявных преобразований к булевому типу. По правилам синтаксиса языка C++, then-ветвь оператора следует сразу за круглой скобкой без ключевого слова then, типичного для большинства языков программирования. Каждый из операторов может быть блоком, в частности, if-оператором. Поэтому возможна и такая конструкция:

```
if(выражение1) if(выражение2) if(выражение3) ...
```

Ветви else и if, позволяющие организовать выбор из многих возможностей, могут отсутствовать. Может быть опущена и заключительная else-ветвь. В этом случае краткая форма оператора if задает альтернативный выбор − делать или не делать − выполнять или не выполнять then-оператор.

Семантика оператора if проста и понятна. Выражения if проверяются в порядке их написания. Как только получено значение true, проверка прекращается и выполняется оператор (это может быть блок), который следует за выражением, получившим значение true. С завершением этого оператора завершается и оператор if. Ветвь else, если она есть, относится к ближайшему открытому if.

Оператор switch.

Частным, но важным случаем выбора из нескольких вариантов является ситуация, при которой выбор варианта определяется значениями некоторого выражения. Соответствующий оператор C#, унаследованный от C++, но с небольшими изменениями в синтаксисе, называется оператором switch. Синтаксис оператора:

```
switch(выражение)
{
  case константное_выражение_1: [операторы_1 оператор_перехода_1]
  ...
  case константное_выражение_K: [операторы_K оператор_перехода_K]
  [default: операторы_N оператор_перехода_N]
}
```

Ветвь default может отсутствовать. Следует отметить, что по синтаксису допустимо, чтобы после двоеточия следовала пустая последовательность операторов, а не последовательность, заканчивающаяся оператором перехода. Константные выражения в case должны иметь тот же тип, что и switch-выражение.

Семантика оператора switch чуть запутана. Вначале вычисляется значение switch-выражения. Затем оно поочередно в порядке следования case сравнивается на совпадение с константными выражениями. Как только достигнуто совпадение, выполняется соответствующая последовательность операторов case-ветви. Поскольку последний оператор этой последовательности является оператором перехода (чаще всего это оператор break), то обычно он завершает выполнение оператора switch. Использование операторов перехода нежелательно. Таким оператором может быть оператор goto, передающий управление другой case-ветви, которая, в свою очередь, может передать управление еще куда-нибудь. Семантика осложняется еще и тем, что case-ветвь может быть пустой последовательностью операторов. Тогда в случае совпадения константного выражения этой ветви со значением switch-выражения будет выполняться первая непустая последовательность очередной case-ветви. Если значение switch-выражения не совпадает ни с одним константным выражением, то выполняется последовательность операторов

ветви default, если же таковой ветви нет, то оператор switch эквивалентен пустому оператору.

Еще одна неудача в синтаксической конструкции switch связана с существенным ограничением, накладываемым на case-выражения, которые могут быть только константным выражением.

Разбор случаев − это часто встречающаяся ситуация в самых разных задачах. Применяя оператор switch, следует помнить о недостатках его синтаксиса, использовать его в правильном стиле. Каждую case-ветвь нужно заканчивать оператором break.

Когда разбор случаев предполагает проверку попадания в некоторый диапазон значений, приходится прибегать к оператору if для формирования специальной переменной. Этот прием демонстрируется в следующем примере, где идет работа над данными класса Testing:

```
public void SetPeriod()
{
   if ((age > 0)&& (age <7))period=1;
   else if ((age >= 7)&& (age <17))period=2;
   else if ((age >= 17)&& (age <22))period=3;
   else if ((age >= 22)&& (age <27))period=4;
   else if ((age >= 27)&& (age <37))period=5;
   else period =6;
}
```

Этот пример демонстрирует применение ветвящегося оператора if. С содержательной точки зрения он интересен тем, что в поля класса пришлось ввести специальную переменную period, позволяющую в дальнейшем использовать разбор случаев в зависимости от периода жизни:

```
public void SetStatus()
{
   switch (period)
   {
      case 1:
         status = "child";
         break;
      case 2:
         status = "schoolboy";
         break;
      case 3:
         status = "student";
         break;
      case 4:
         status = "junior researcher";
         break;
      case 5:
         status = "senior researcher";
         break;
      case 6:
         status = "professor";
```

```
            break;
         default :
            status = "не определен";
            break;
      }
}
```

Этот пример демонстрирует корректный стиль использования оператора switch.

Задание 1

Составьте программу по алгоритму разветвляющейся структуры из лабораторной работы № 22 и создайте проект консольного типа в MS Visual Studio для ее решения.

Задание 2

Создайте проект консольного типа в MS Visual Studio для решения задачи по своему варианту с использованием условного оператора If.

If1. Дано целое число. Если оно является положительным, то прибавить к нему 1, в противном случае не изменять его. Вывести полученное число.

If2. Дано целое число. Если оно является положительным, то прибавить к нему 1, в противном случае вычесть из него 2. Вывести полученное число.

If3. Дано целое число. Если оно является положительным, то прибавить к нему 1, если отрицательным, то вычесть из него 2, если нулевым, то заменить его на 10. Вывести полученное число.

If4. Даны три целых числа. Найти количество положительных чисел в исходном наборе.

If5. Даны три целых числа. Найти количество положительных и количество отрицательных чисел в исходном наборе.

If6. Даны два числа. Вывести большее из них.

If7. Даны два числа. Вывести порядковый номер меньшего из них.

If8. Даны два числа. Вывести вначале большее, а затем меньшее из них.

If9. Даны две переменные вещественного типа: A, B. Перераспределить значения данных переменных так, чтобы в A оказалось меньшее из значений, а в B − большее. Вывести новые значения переменных A и B.

If10. Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной сумму этих значений, а если равны, то при- своить переменным нулевые значения. Вывести новые значения переменных A и B.

If11. Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной большее из этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.

If12. Даны три числа. Найти наименьшее из них.

If13. Даны три числа. Найти среднее из них (то есть число, расположенное между наименьшим и наибольшим).

If14. Даны три числа. Вывести вначале наименьшее, а затем наибольшее из данных чисел.

If15. Даны три числа. Найти сумму двух наибольших из них.

If16. Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию, то удвоить их, в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных A, B, C. 20

If17. Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию или убыванию, то удвоить их, в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных A, B, C.

If18. Даны три целых числа, одно из которых отлично от двух других, равных между собой. Определить порядковый номер числа, отличного от остальных.

If19. Даны четыре целых числа, одно из которых отлично от трех других, равных между собой. Определить порядковый номер числа, отличного от остальных.

If20. На числовой оси расположены три точки: A, B, C. Определить, какая из двух последних точек (B или C) расположена ближе к A, и вывести эту точку и ее расстояние от точки A.

Задание 3

Создайте проект консольного типа в MS Visual Studio для решения задачи по своему варианту с использованием условного оператора выбора Case.

Case1. Дано целое число в диапазоне 1–7. Вывести строку − название дня недели, соответствующее данному числу (1 − «понедельник», 2 − «вторник» и т. д.).

Case2. Дано целое число K. Вывести строку-описание оценки, соответствующей числу K (1 − «плохо», 2 – «неудовлетворительно», 3 − «удовлетворительно», 4 − «хорошо», 5 − «отлично»). Если K не лежит в диапазоне 1–5, то вывести строку «ошибка».

Case3. Дан номер месяца − целое число в диапазоне 1–12 (1 − январь, 2 − февраль и т. д.). Вывести название соответствующего времени года («зима», «весна», «лето», «осень»).

Case4°. Дан номер месяца − целое число в диапазоне 1–12 (1 − январь, 2 − февраль и т. д.). Определить количество дней в этом месяце для невисокосного года.

Case5. Арифметические действия над числами пронумерованы следующим образом: 1 – сложение; 2 – вычитание; 3 – умножение; 4 − деление. Дан номер

действия N (целое число в диапазоне 1–4) и вещественные числа A и B (B не равно 0). Выполнить над числами указанное действие и вывести результат.

Case6. Единицы длины пронумерованы следующим образом: 1 – дециметр; 2 – километр; 3 – метр; 4 – миллиметр; 5 − сантиметр. Дан номер единицы длины (целое число в диапазоне 1–5) и длина отрезка в этих единицах (вещественное число). Найти длину отрезка в метрах.

Case7. Единицы массы пронумерованы следующим образом: 1 – килограмм; 2 – миллиграмм; 3 – грамм; 4 – тонна; 5 − центнер. Дан номер единицы массы (целое число в диапазоне 1–5) и масса тела в этих единицах (вещественное число). Найти массу тела в килограммах.

Case8. Даны два целых числа D (день) и M (месяц), определяющие правильную дату невисокосного года. Вывести значения D и M для даты, предшествующей указанной.

Case9. Даны два целых числа D (день) и M (месяц), определяющие правильную дату невисокосного года. Вывести значения D и M для даты, следующей за указанной.

Case10. Робот может перемещаться в четырех направлениях («С» − север, «З» − запад, «Ю» − юг, «В» − восток) и принимать три цифровые команды: 0 − продолжать движение; 1 − поворот налево; 1 − поворот направо. Дан символ C − исходное направление робота и целое число N − посланная ему команда. Вывести направление робота после выполнения полученной команды.

Case11. Локатор ориентирован на одну из сторон света («С» − север, «З» − запад, «Ю» − юг, «В» − восток) и может принимать три цифровые команды поворота: 1 − поворот налево; -1 − поворот направо; 2 − поворот на 180°. Дан символ C − исходная ориентация локатора и целые числа N1 и N2 − две посланные команды. Вывести ориентацию локатора после выполнения этих команд.

Case12. Элементы окружности пронумерованы следующим образом: 1 − радиус R; 2 − диаметр D = 2·R; 3 − длина L = 2·π·R; 4 − площадь круга $S = \pi \cdot R^2$. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данной окружности (в том же порядке). В качестве значения π использовать 3,14.

Case13. Элементы равнобедренного прямоугольного треугольника пронумерованы следующим образом: 1 − катет a; 2 − гипотенуза $c = a^2$; 3 − высота h; опущенная на гипотенузу (h = c / 2); 4 − площадь S = c · h / 2. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данного треугольника (в том же порядке).

Case14. Элементы равностороннего треугольника пронумерованы следующим образом: 1 − сторона a; 2 − радиус R1 вписанной окружности ($R1 = a^3 / 6$); 3 − радиус R2 описанной окружности (R2 = 2 · R1); 4 − площадь $S = 2\ a^3 / 4$. Дан номер одного из этих элементов и его значение. Вывести значения остальных элементов данного треугольника (в том же порядке).

Case15. Мастям игральных карт присвоены порядковые номера: 1 − пики, 2 − трефы, 3 − бубны, 4 − червы. Достоинству карт, старших десятки, присвоены номера: 11 − валет, 12 − дама, 13 − король, 14 − туз. Даны два целых числа: N − достоинство ($6 \le N \le 14$) и M − масть карты ($1 \le M \le 4$). Вывести название соответствующей карты вида «шестерка бубен», «дама червей», «туз треф» и т. п.

Case16. Дано целое число в диапазоне 20–69, определяющее возраст (в годах). Вывести строку-описание указанного возраста, обеспечив правильное согласование числа со словом «год», например: 20 − «двадцать лет», 32 − «тридцать два года», 41 − «сорок один год».

Контрольные вопросы

1 Для чего необходимо ветвление в алгоритмах?
2 Какие формы ветвления различают?
3 Какие операторы языка C# используются в разветвляющихся алгоритмах?
4 Приведите синтаксис и пример использования условного оператора If.
5 Приведите синтаксис и пример использования оператора выбора Switch.

4 Лабораторная работа № 25. Оператор цикла for. Операторы цикла while и do while

Цель работы: изучить операторы циклов и научиться использовать их в программах.

4.1 Краткие теоретические сведения

Циклы применяются в программах C# для выполнения каких-либо повторяющихся действий.

Цикл for.

Оператор for предназначен для повторного выполнения оператора или группы операторов заданное количество раз. Пример этого оператора в общем виде:

```
for([Инициализация];[Условие];[Приращение]) <Оператор>
```

Оператор [Инициализация] выполняется один раз перед началом цикла. Перед каждой итерацией (то есть перед каждым выполнением тела цикла <Оператор>) проверяется [Условие]. И, наконец, после каждой итерации выполняется оператор [Приращение].

Как правило, в цикле имеется переменная, играющая роль так называемой переменной цикла. При каждой итерации переменная цикла изменяет свое значение в заданных пределах.

Начальное значение переменной цикла задается в программе до оператора for или в операторе [Инициализация]. Предельное значение переменной цикла определяется оператором приращения, а проверка ее текущего значения − в блоке [Условие].

Пример

```
int i;
for (i = 0; i < 10; i++)
{
    System.Console.Write("{0}", i);
}
```

Здесь переменная i используется в качестве переменной цикла. Перед началом цикла ей присваивается нулевое значение. Перед каждой итерацией содержимое переменной i сравнивается с числом 10. Если i меньше 10, тело цикла выполняется один раз. В тело цикла помещен вызов метода Write, отображающий текущее значение переменной цикла на консоли.

После выполнения тела цикла значение i увеличивается на единицу в блоке приращения. Далее переменная цикла вновь сравнивается с числом 10. Когда значение i превысит 10, цикл завершится.

Таким образом, параметр цикла анализируется перед выполнением тела цикла, а модифицируется после его выполнения.

Вывод на консоль приведенного выше фрагмента программы: 0123456789

Прерывание цикла.

С помощью оператора break можно в любой момент прервать выполнение цикла. Например, в следующем фрагменте программы прерывается работа цикла, когда значение переменной i становится больше пяти:

```
for (i = 0; i < 10; i++)
{
    if (i > 5)
    {
        break;
    }
    System.Console.Write(" {0} ", i);
}
```

В результате на консоль будут выведены цифры от 0 до 5: 0 1 2 3 4 5

Допускается опускать реализацию любого блока оператора for:

```
int i = 0;
for (; ; )
{
    if (i > 10)
    {
        break;
    }
    System.Console.Write(" {0} ", i);
    i++;
}
```

Это позволяет реализовывать любую необходимую логику циклической обработки.

При создании цикла обязательно нужно предусмотреть условие его завершения. Если же этого не сделать, цикл будет выполняться бесконечно.

Программа при этом будет работать вхолостую на одном месте, или, как еще говорят, «зациклится».

Пример цикла, из которого нет выхода:

```
for (i = 0; ; )   //   Зацикливание!
{
    i++;
    System.Console.Write("{0} ", i);
  }
```

Здесь не предусмотрена проверка значения переменной цикла, поэтому программа будет постоянно выводить на консоль возрастающие значения, пока не прервется ее работа. Это можно сделать, нажав комбинацию клавиш Control-C или закрыв консольное окно.

Возобновление цикла.

В отличие от оператора break, прерывающего цикл, оператор continue позволяет возобновить выполнение цикла с самого начала.

Пример его использования:

```
for (i = 0; ; i++)
{
    System.Console.Write("{0} ", i);
    if (i < 9)
    {
        continue;
    }
    else
    {
        break;
    }
}
```

Если в ходе выполнения цикла значение переменной i не достигло девяти, цикл возобновляет свою работу с самого начала (то есть вывода значения переменной цикла на консоль). Когда указанное значение будет достигнуто, выполнение цикла прервется оператором break.

Цикл while.

Оператор while проверяет условие завершения цикла перед выполнением тела цикла:

```
i = 0;
while (i < 10)
{
    Systern.Console.Write("{0} ", i);
    i++;
}
```

В отличие от оператора for оператор while никак не изменяет значения переменной цикла.

Перед тем как приступить к выполнению цикла, устанавливают начальное значение параметра цикла i, равное нулю. После выполнения тела цикла необходимо самостоятельно изменять значение параметра цикла, увеличивая его на единицу. Цикл будет прерван, как только значение переменной i превысит 10.

В цикле while можно использовать описанные ранее операторы прерывания цикла break и возобновления цикла continue.

Следующий цикл будет выполняться бесконечно:

```
while (true)
  {
      System.Console.Write("{0}", i);
    i++;
}
```

Цикл do.

Оператор do используется вместе с ключевым словом while. При этом условие завершения цикла проверяется после выполнения его тела:

```
  i = 0;
do
{
    System.Console.Write("{0}", i);
    i++;
} while (i < 10);
```

Как только это значение достигнет 10, цикл будет прерван.

Аналогично циклу while цикл do допускает прерывание оператором break и возобновление оператором continue.

Задание 1

Составьте программу по алгоритму циклической структуры из лабораторной работы № 22 и создайте проект консольного типа в MS Visual Studio для ее решения. Шаг и значение функции выводить на консоль на каждой итерации.

Задание 2

Создайте проект консольного типа в MS Visual Studio для решения задачи по своему варианту с использованием оператора циклов For .

For1. Даны целые числа K и N (N > 0). Вывести N раз число K.

For2. Даны два целых числа A и B (A < B). Вывести в порядке возрастания все целые числа, расположенные между A и B (включая сами числа A и B), а также количество N этих чисел.

For3. Даны два целых числа A и B (A < B). Вывести в порядке убывания все целые числа, расположенные между A и B (не включая числа A и B), а также количество N этих чисел.

For4. Дано вещественное число − цена 1 кг конфет. Вывести стоимость 1, 2, …, 10 кг конфет.

For5. Дано вещественное число − цена 1 кг конфет. Вывести стоимость 0,1; 0,2; …; 1 кг конфет.

For6. Дано вещественное число − цена 1 кг конфет. Вывести стоимость 1,2; 1,4; …; 2 кг конфет.

For7. Даны два целых числа A и B (A < B). Найти сумму всех целых чисел от A до B включительно.

For8. Даны два целых числа A и B (A < B). Найти произведение всех целых чисел от A до B включительно.

For9. Даны два целых числа A и B (A < B). Найти сумму квадратов всех целых чисел от A до B включительно.

For10. Дано целое число N (> 0). Найти сумму 1 + 1/2 + 1/3 + … + 1/N (вещественное число).

For11. Дано целое число N (> 0). Найти сумму N2 + (N + 1) 2 + (N + 2) 2 + +… + (2 · N) 2 (целое число).

For12. Дано целое число N (> 0). Найти произведение 1,1·1,2·1,3… (N сомножителей).

For13. Дано целое число N (> 0). Найти значение выражения 1,1 – 1,2 + 1,3… (N слагаемых, знаки чередуются). Условный оператор не использовать.

For14. Дано целое число N (> 0). Найти квадрат данного числа, используя для его вычисления следующую формулу: N2 = 1 + 3 + 5 + … + (2·N – 1). После добавления к сумме каждого слагаемого выводить текущее значение суммы (в результате будут выведены квадраты всех целых чисел от 1 до N).

For15. Дано вещественное число A и целое число N (> 0). Найти A в степени N: AN = A·A· … ·A (числа A перемножаются N раз).

Задание 3

Создайте проект консольного типа в MS Visual Studio для решения задачи по своему варианту с использованием оператора циклов While.

While1. Даны положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Не используя операции умножения и деления, найти длину незанятой части отрезка A.

While2. Даны положительные числа A и B (A > B). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Не используя операции умножения и деления, найти количество отрезков B, размещенных на отрезке A.

While3. Даны целые положительные числа N и K. Используя только операции сложения и вычитания, найти частное от деления нацело N на K, а также остаток от этого деления.

While4. Дано целое число N (> 0). Если оно является степенью числа 3, то вывести True, если не является − вывести False.

While5. Дано целое число N (> 0), являющееся некоторой степенью числа 2 : N = 2K . Найти целое число K − показатель этой степени.

While6. Дано целое число N (> 0). Найти двойной факториал N: N!! = N·(N–2)·(N–4) … (последний сомножитель равен 2, если N четное, и 1, если N нечетное). Чтобы избежать целочисленного переполнения, вычислять это произведение с помощью вещественной переменной и вывести его как вещест- венное число.

While7. Дано целое число N (> 0). Найти наименьшее целое положительное число K, квадрат которого превосходит N: K2 > N. Функцию извлечения квадратного корня не использовать.

While8. Дано целое число N (> 0). Найти наибольшее целое число K, квадрат которого не превосходит N: K2 ≤ N. Функцию извлечения квадратного кор- ня не использовать.

While9. Дано целое число N (> 1). Найти наименьшее целое число K, при котором выполняется неравенство 3K > N.

While10. Дано целое число N (> 1). Найти наибольшее целое число K, при котором выполняется неравенство 3K < N.

While11. Дано целое число N (> 1). Вывести наименьшее из целых чисел K, для которых сумма 1 + 2 + … + K будет больше или равна N, и саму эту сумму.

While12. Дано целое число N (> 1). Вывести наибольшее из целых чисел K, для которых сумма 1 + 2 + … + K будет меньше или равна N, и саму эту сумму.

While13. Дано число A (> 1). Вывести наименьшее из целых чисел K, для которых сумма 1 + 1/2 + … + 1/K будет больше A, и саму эту сумму.

While14. Дано число A (> 1). Вывести наибольшее из целых чисел K, для которых сумма 1 + 1/2 + … + 1/K будет меньше A, и саму эту сумму.

While15. Начальный вклад в банке равен 1000 р. Через каждый месяц размер вклада увеличивается на P процентов от имеющейся суммы (P – вещественное число, 0 < P < 25). По данному P определить, через сколько месяцев размер вклада превысит 1100 р., и вывести найденное количество месяцев K (целое число) и итоговый размер вклада S (вещественное число).

Контрольные вопросы

1 Для чего используют структуру «цикл»?

2 Какие управляющие инструкции C# используются в цикличес- ких алгоритмах?

3 Приведите синтаксис и пример использования оператора For.

4 Приведите синтаксис и пример использования оператора While.

5 Приведите синтаксис и пример использования оператора Do..While. В чем его отличие от While?

6 Какие операторы управляют прерываниями циклов?

5 Лабораторная работа № 26. Обработка одномерных массивов. Сортировка массивов

Цель работы: изучение массивов; закрепление навыков структурного программирования при работе с одномерными массивами.

Массив задает способ организации данных. ***Массивом*** называют упорядоченную совокупность элементов одного типа. Каждый элемент массива имеет индексы, определяющие порядок элементов. Число индексов характеризует ***размерность массива***. Каждый индекс изменяется в некотором диапазоне [a,b]. В языке C#, как и во многих других языках, индексы задаются целочисленным типом. В других языках, например, в языке Паскаль, индексы могут принадлежать счетному конечному множеству, на котором определены функции, задающие следующий и предыдущий элемент. Диапазон [a,b] называется ***граничной парой***, a − ***нижней границей***, b − ***верхней границей*** индекса. При объявлении массива границы задаются выражениями. Если все границы заданы константными выражениями, то число элементов массива известно в момент его объявления и ему может быть выделена память еще на этапе трансляции. Такие массивы называются ***статическими***. Если же выражения, задающие границы, зависят от переменных, то такие массивы называются ***динамическими***, поскольку память им может быть отведена только динамически в процессе выполнения программы, когда становятся известными значения соответствующих переменных. Массиву, как правило, выделяется непрерывная область памяти.

В языке C# снято существенное ограничение языка C++ на статичность массивов. Массивы в языке C# являются настоящими динамическими массивами. Как следствие этого, массивы относятся к ссылочным типам, память им отводится динамически в «куче». К сожалению, не снято ограничение 0-базируемости.

В языке C#, с соблюдением преемственности сохранены одномерные массивы и массивы массивов. В дополнение к ним в язык добавлены многомерные массивы. Динамические многомерные массивы языка C# являются весьма мощной, надежной, понятной и удобной структурой данных, которую смело можно рекомендовать к применению не только профессионалам, но и новичкам, программирующим на C#.

Объявление массивов

Рассмотрим, как объявляются одномерные массивы.

Объявление одномерных массивов. Общая структура объявления:

[<атрибуты>] [<модификаторы>] <тип> <объявители>;

Объявление одномерного массива выглядит следующим образом:

<тип>[] <объявители>;

Следует отметить, что в отличие от языка C++ квадратные скобки приписаны не к имени переменной, а к типу. Они являются неотъемлемой частью определения класса, так что запись T[] следует понимать как класс «одномерный массив» с элементами типа T.

Что же касается границ изменения индексов, то эта характеристика к классу не относится, она является характеристикой переменных-экземпляров, каждый из которых является одномерным массивом со своим числом элементов, задаваемых в объявителе переменной.

Как и в случае объявления простых переменных, каждый объявитель может быть именем или именем с инициализацией. В первом случае речь идет об отложенной инициализации. Нужно понимать, что при объявлении с отложенной инициализацией сам массив не формируется, а создается только ссылка на массив, имеющая неопределенное значение Null. Поэтому пока массив не будет реально создан и его элементы инициализированы, использовать его в вычислениях нельзя. Пример объявления трех массивов с отложенной инициализацией:

```
int[] a, b, c;
```

Чаще всего при объявлении массива используется имя с инициализацией. И опять-таки, как и в случае простых переменных, могут быть два варианта инициализации. В первом случае инициализация является явной и задается константным массивом.

Пример

```
double[] x= {5.5, 6.6, 7.7};
```

Следуя синтаксису, элементы константного массива следует заключать в фигурные скобки.

Во втором случае создание и инициализация массива выполняется в объектном стиле с вызовом конструктора массива. И это наиболее распространенная практика объявления массивов.

Пример

```
int[] d= new int[5];
```

Итак, если массив объявляется без инициализации, то создается только висячая ссылка со значением Null. Если инициализация выполняется конструктором, то в динамической памяти создается сам массив, элементы которого инициализируются константами соответствующего типа (ноль для арифметики, пустая строка для строковых массивов), и ссылка связывается с этим массивом. Если массив инициализируется константным массивом, то в памяти создается константный массив, с которым и связывается ссылка.

Элементы массива, если они не заданы при инициализации, либо вычисляются, либо вводятся пользователем.

Пример работы с массивами:

```
 public void TestDeclaration()
{
   //объявляются три одномерных массива A,B,C
   int[] A = new int[5], B = new int[5], C = new int[5];

   A[0] = 0;
   A[1] = 1;
   A[2] = 2;
   A[3] = 3;
   A[4] = 4;

   B[0] = 0;
   B[1] = 1;
   B[2] = 2;
   B[3] = 3;
   B[4] = 4;


   for (int i = 0; i < 5; i++)
   {
      C[i] = A[i] + B[i];
   }

   //объявление массива с явной инициализацией
   int[] x ={ 5, 5, 6, 6, 7, 7 };

   //объявление массивов с отложенной инициализацией
   int[] u, v;
   u = new int[3];

   for (int i = 0; i < 3; i++)
   {
      u[i] = i + 1;
   }

   v = new int[4];
   v = u; //допустимое присваивание

}
```

В процедуре показаны разные способы объявления массивов.

Динамические массивы

Во всех вышеприведенных примерах объявлялись статические массивы, поскольку нижняя граница равна нулю по определению, а верхняя всегда задавалась в этих примерах константой. В C# все массивы, независимо от того, каким выражением описывается граница, рассматриваются как динамические, и память для них распределяется в «куче». В действительности реальные

потребности в размере массива скорее всего выясняются в процессе работы в диалоге с пользователем.

Чисто синтаксически нет существенной разницы в объявлении статических и динамических массивов. Выражение, задающее границу изменения индексов, в динамическом случае содержит переменные. Единственное требование − значения переменных должны быть определены в момент объявления. Это ограничение в C# выполняется автоматически, поскольку хорошо известно, сколь требовательно C# контролирует инициализацию переменных.

Пример работы с динамическим массивом:

```
        public void TestDynAr()
       {
           //объявление динамического массива A1
           Console.WriteLine("Введите число элементов массива
A1");

           int size = Convert.ToInt32(Console.ReadLine());
           int[] A1 = new int[size];
        }
```

В особых комментариях эта процедура не нуждается. Здесь верхняя граница массива определяется пользователем.

Задание 1

Вариант 1. Вычислите среднее арифметическое одномерного массива, за полненного целыми числами.

Вариант 2. Замените в одномерном массиве, заполненном целыми числами, все элементы, кратные 3, нулями.

Вариант 3. Одномерный массив заполнен целыми числами. Вычислите сумму его элементов, кратных 5.

Вариант 4. Одномерный массив заполнен любыми числами. Замените отрицательные элементы их модулями.

Вариант 5. Одномерный массив заполнен целыми числами. Возведите в квадрат элементы, стоящие на четных местах.

Вариант 6. Заполните одномерный массив из 100 элементов случайными числами от 1 до 10. Сколько получилось пятерок?

Вариант 7. Вычислите произведение всех отрицательных элементов одномерного массива.

Вариант 8. Найдите максимальный элемент одномерного массива.

Вариант 9. Одномерный массив заполнен целыми числами. Удвойте четные и утройте нечетные элементы.

Вариант 10. Заполните одномерный массив из 50 элементов случайными числами от -5 до 5. Сколько получилось отрицательных чисел?

Вариант 11. Заполните одномерный массив из 10 элементов случайными

числами от 1 до 100. Вычислите максимальный и минимальный элементы массива.

Вариант 12. Одномерный массив заполнен целыми числами. Поменяйте местами первый и максимальный элементы.

Примечание – Работа со случайными числами:

```
Random rnd = new Random();
int a = rnd.Next(10);
```

10 – верхняя граница получаемого случайного числа. В переменной «а» – случайное число.

Задание 2

1 Дан массив размера N и целые числа K и L ($1 < K \le L \le N$). Найти среднее арифметическое всех элементов массива, кроме элементов с номерами от K до L включительно.

2 Дан целочисленный массив размера N. Проверить, чередуются ли в нем четные и нечетные числа. Если чередуются, то вывести 0, если нет, то вывести порядковый номер первого элемента, нарушающего закономерность.

3 Дан массив ненулевых целых чисел размера N. Проверить, чередуются ли в нем положительные и отрицательные числа. Если чередуются, то вывести 0, если нет, то вывести порядковый номер первого элемента, нарушающего закономерность.

4 Дан массив A размера N. Найти минимальный элемент из его элементов с четными номерами: A_2, A_4, A_6, ...

5 Дан массив A размера N. Найти максимальный элемент из его элементов с нечетными номерами: A_1, A_3, A_5, ...

6 Дан массив размера N. Найти номера тех элементов массива, которые больше своего левого соседа, и количество таких элементов. Найденные номера выводить в порядке их убывания.

7 Дан массив размера N. Найти номер его первого локального минимума (локальный минимум – это элемент, который меньше любого из своих соседей).

8 Дан массив размера N. Найти номер его последнего локального максимума (локальный максимум – это элемент, который больше любого из своих соседей).

9 Дан массив размера N. Найти количество участков, на которых его элементы возрастают.

10 Дан массив размера N. Найти количество участков, на которых его элементы убывают.

11 Дан массив размера N. Найти количество его промежутков монотонности (то есть участков, на которых его элементы возрастают или убывают).

12 Дано число R и массив A размера N. Найти элемент массива, который наиболее близок к числу R (то есть такой элемент A_K, для которого величина $|A_K–R|$ является минимальной).

Задание 3

1 Дан массив размера N. Поменять местами его минимальный и максимальный элементы.

2 Дан массив размера N (N – четное число). Поменять местами его первый элемент со вторым, третий с четвертым и т. д.

3 Дан массив размера N (N – четное число). Поменять местами первую и вторую половины массива.

4 Дан массив размера N. Поменять порядок его элементов на обратный.

5 Дан массив A размера N и целые числа K и L (1 < K < L < N). Переставить в обратном порядке элементы массива, расположенные между элементами AK и AL, включая эти элементы.

6 Дан массив A размера N и целые числа K и L (1 < K < L < N). Переставить в обратном порядке элементы массива, расположенные между элементами AK и AL, не включая эти элементы.

7 Дан массив размера N. Обнулить элементы массива, расположенные между его минимальным и максимальным элементами (не включая минимальный и максимальный элементы).

8 Дан массив размера N. Переставить в обратном порядке элементы массива, расположенные между его минимальным и максимальным элементами, включая минимальный и максимальный элементы.

9 Дан массив размера N. Заменить каждый элемент массива на среднее арифметическое этого элемента и его соседей.

10 Дан массив размера N, все элементы которого, кроме первого, упорядочены по возрастанию. Сделать массив упорядоченным, переместив первый элемент на новую позицию.

11 Дан массив размера N, все элементы которого, кроме последнего, упорядочены по возрастанию. Сделать массив упорядоченным, переместив последний элемент на новую позицию.

12 Дан массив размера N, все элементы которого, кроме одного, упорядочены по убыванию. Сделать массив упорядоченным, переместив элемент, нарушающий упорядоченность, на новую позицию.

Контрольные вопросы

1 Что такое массив и какие способы его создания Вы знаете на языке C#?

2 Как объявить одномерный статический массив на языке C#?

3 Что такое динамический массив?

4 Как создать динамический массив на языке C#?

5 Что такое индекс элемента массива?

6 Как нумеруются элементы и пишутся индексы одномерного массива на языке C#?

6 Лабораторная работа № 27. Обработка двухмерных массивов

Цель работы: изучение двумерных массивов на языке C# и алгоритмов их обработки.

Задание 1

1 Дан двумерный массив целых чисел размерностью N×N. Найдите сумму его элементов.

2 Введите целочисленные элементы матрицы N×N. Утройте значение каждого элемента матрицы, который больше 4.

3 Найдите сумму элементов столбца и строки матрицы, на пересечении которых находится максимальный элемент матрицы.

4 На плоскости задано 40 точек. Найдите расстояние до самой удаленной от начала координат точки.

5 В квадратной таблице (любые целые числа) обменяйте местами элементы строки и столбца, на пересечении которых находится первый минимальный элемент из положительных чисел.

6 Дана квадратная матрица. Составьте программу, которая прибавила бы каждому элементу данной строки элемент, принадлежащий этой строке и главной диагонали.

7 Дан двумерный массив. Найдите максимальный элемент и заполните строку и столбец , в которых он расположен, нулями.

8 Дан двумерный массив. Найдите сумму всех элементов главной диагонали.

9 Дан двумерный массив. Переставьте местами элементы на диагоналях (построчно)

10 Дан двумерный массив. Переставьте местами максимальный и минимальный элементы.

Размерность массивов вводите с клавиатуры.

Задание 2

1 Даны целые положительные числа M и N . Сформируйте целочисленную матрицу размера M × N, у которой все элементы i-й строки имеют значение 10 I (I = 1, …, M).

2 Даны целые положительные числа M и N . Сформируйте целочисленную матрицу размера M × N, у которой все элементы j-го столбца имеют значение 5J (J = 1, ..., N).

3 Даны целые положительные числа M, N и набор из M чисел. Сформировать матрицу размера M × N, у которой в каждом столбце содержатся все числа из исходного набора (в том же порядке).

4 Даны целые положительные числа M, N и набор из N чисел. Сформировать матрицу размера M × N, у которой в каждой строке содержатся все числа из исходного набора (в том же порядке).

5 Даны целые положительные числа M, N, число D и набор из M чисел.

Сформировать матрицу размера M × N, у которой первый столбец совпадает с исходным набором чисел, а элементы каждого следующего столбца равны сумме соответствующего элемента предыдущего столбца и числа D (в результате каждая строка матрицы будет содержать элементы арифметической прогрессии).

6 Даны целые положительные числа M, N, число D и набор из N чисел. Сформировать матрицу размера M × N, у которой первая строка совпадает с исходным набором чисел, а элементы каждой следующей строки равны соответствующему элементу предыдущей строки, умноженному на D (в результате каждый столбец матрицы будет содержать элементы геометрической прогрессии).

7 Дана матрица размера M × N и целое число K (1 < K < M). Вывести элементы k-й строки данной матрицы.

8 Дана матрица размера M × N и целое число K (1 < K < M). Вывести элементы k-го столбца данной матрицы.

9 Дана матрица размера M × N. Вывести ее элементы, расположенные в строках с четными номерами (2, 4, ...). Вывод элементов производить по строкам, условный оператор не использовать.

10 Дана матрица размера M × N. Вывести ее элементы, расположенные в столбцах с нечетными номерами (1, 3, ...). Вывод элементов производить по столбцам, условный оператор не использовать.

Задание 3

1 Дана матрица размера M × N. Для каждого столбца матрицы с четным номером (2, 4, ...) найти сумму его элементов. Условный оператор не использовать.

2 Дана матрица размера M × N. В каждой строке матрицы найти минимальный элемент.

3 Дана матрица размера M × N. В каждом столбце матрицы найти максимальный элемент.

4 Дана матрица размера M × N. Найти номер ее строки с наибольшей суммой элементов и вывести данный номер, а также значение наибольшей суммы.

5 Дана матрица размера M × N. Найти номер ее столбца с наименьшим произведением элементов и вывести данный номер, а также значение наименьшего произведения.

6 Дана матрица размера M × N. Найти максимальный среди минимальных элементов ее строк.

7 Дана матрица размера M × N. Найти минимальный среди максимальных элементов ее столбцов.

8 Дана матрица размера M × N. В каждой ее строке найти количество элементов, меньших среднего арифметического всех элементов этой строки.

9 Дана матрица размера M × N. В каждом ее столбце найти количество элементов, больших среднего арифметического всех элементов этого столбца.

10 Дана матрица размера M × N. Найти номера строки и столбца для

элемента матрицы, наиболее близкого к среднему значению всех ее элементов.

Контрольные вопросы

1 Что такое двумерный массив?
2 Как создать двумерный массив на языке C#?
3 Как обратиться к элементу двумерного массива в программе?
4 Как вычислить сумму и произведение элементов двумерного массива, строки, столбца?
5 Каким свойством обладают элементы главной диагонали матрицы?
6 Как вывести двумерный массив на экран (печать)?

7 Лабораторная работа № 28. Строковые типы. Обработка текстов и строк

Цель работы: изучение строковых типов, использование строковых типов

В языке C# есть несколько классов для представления символьной информации: char, string и StringBuilder.

Задание 1

1 Для заданной строки символов проверить, является ли она симметричной или нет. (Симметричной считается строка, которая одинаково читается слева направо и справа налево.)
2 Для заданной строки символов определить сумму всех входящих в неё цифр.
3 Для заданной строки определить все входящие в неё символы. Например: строка «abccbbbabba» состоит из символов «a», «b» и «c».
4 Задана строка символов. Определить, какой символ встречается в этой строке подряд наибольшее число раз. В ответе указать символ, образующий самую длинную последовательность, длину последовательности.
5 Для заданной строки символов, состоящей из строчных букв и пробелов, определить слово наибольшей длины, которое начинается и заканчивается на одну и ту же букву.
6 Задана строка символов, содержащая два или более слов, разделенных пробелами. Написать программу, меняющую местами все четные и нечетные слова в строке.
7 Подсчитать, сколько раз в данной строке встречается некоторая буква, вводимая с клавиатуры.
8 Из строки удалить среднюю букву, если длина строки нечетная, если четная − удалить две средние буквы.
9 Заменить все вхождения в текст некоторой буквы на другую букву (их значения вводить с клавиатуры).

10 Заменить все вхождения подстроки Str1 на подстроку Str2 (подстроки вводятся с клавиатуры).

11 Дана последовательность слов. Напечатать все слова в алфавитном порядке.

12 Дана последовательность слов. Напечатать все слова последовательности, которые встречаются в ней по одному разу.

Задание 2

1 Дан символ C. Вывести его код (то есть номер в кодовой таблице).

2 Дано целое число N ($32 \le N \le 126$). Вывести символ с кодом, равным N.

3 Дан символ C. Вывести два символа, первый из которых предшествует символу C в кодовой таблице, а второй следует за символом C.

4 Дано целое число N ($1 \le N \le 26$). Вывести N первых прописных (то есть заглавных) букв латинского алфавита.

5 Дано целое число N ($1 \le N \le 26$). Вывести N последних строчных (то есть маленьких) букв латинского алфавита в обратном порядке (начиная с буквы «z»).

6 Дан символ C, изображающий цифру или букву (латинскую или русскую). Если C изображает цифру, то вывести строку «digit», если латинскую букву − вывести строку «lat», если русскую − вывести строку «rus».

7 Дана непустая строка. Вывести коды ее первого и последнего символа.

8 Дано целое число N > 0 и символ C. Вывести строку длины N, которая состоит из символов C.

9 Дано четное число N > 0 и символы C1 и C2. Вывести строку длины N, которая состоит из чередующихся символов C1 и C2, начиная с C1.

10 Дана строка. Вывести строку, содержащую те же символы, но расположенные в обратном порядке.

11 Дана непустая строка S. Вывести строку, содержащую символы строки S, между которыми вставлено по одному пробелу.

12 Дана непустая строка S и целое число N > 0. Вывести строку, содержащую символы строки S, между которыми вставлено по N символов «*» (звездочка).

13 Дана строка. Подсчитать количество содержащихся в ней цифр.

14 Дана строка. Подсчитать количество содержащихся в ней прописных латинских букв.

15 Дана строка. Подсчитать общее количество содержащихся в ней строчных латинских и русских букв.

16 Дана строка. Преобразовать в ней все прописные латинские буквы в строчные.

17 Дана строка. Преобразовать в ней все строчные буквы (как латинские, так и русские) в прописные.

18 Дана строка. Преобразовать в ней все строчные буквы (как латинские, так и русские) в прописные, а прописные − в строчные.

Контрольные вопросы

1 Какие классы используются для представления символьной информации в C#?
2 Приведите пример описания строковых переменных классов char и string
3 Существуют ли преобразования между классом char и другими классами?
4 Назовите методы и свойства класса Char.
5 Как создать строковую константу?
6 Какие операции над строками используются в C#?
7 Назовите методы и свойства класса String.

8 Лабораторная работа № 29. Понятие класса

Цель работы: получение навыков создания классов на языке C#.

8.1 Краткие теоретические сведения

Класс − это абстрактное понятие, сравнимое с понятием *категория* в его обычном смысле.

Класс в объектно-ориентированном программировании − это абстрактный тип данных, который включает в себя не только **данные**, но и **функции и процедуры**. Функции и процедуры класса называются **методами** и содержат исходный **код**, предназначенный для обработки внутренних **данных** объекта данного класса.

Класс можно также охарактеризовать так: **класс** − описание множества **объектов** и выполняемых над ними **действий**.

Класс описывается (декларируется) следующим образом:

```
Class <Имя_класса>
{
  Секция данных (поля класса)
  Секция методов
}
```

Рассмотрим следующий пример (класс, описывающий базовое поведение прямоугольника):

```
class Rectangle
{
        private int _xPos;
        private int _yPos;
        private double _height;
        private double _weight;

        private static int _angle = 90;

        private static string _className = "Rectangle";
```

```
        public Rectangle(int x1, int y1, int x2, int y2)
        {
                _xPos = x1;
                _xPos = y1;

                _height = y2 - y1;
                _weight = x2 - x1;
        }

        public double GetHeight()
        {
        return _height;
        }

        public double GetWeight()
        {
        return _weight;
        }

        public double GetPerimeter()
        {
        double perimeter = (_height + _weight) * 2;

        return perimeter;
        }

        public void SetRectanglePosition(int xPos, int yPos)
        {
                _xPos = xPos;
                _yPos = yPos;
        }
        public static string GetClassName()
        {
        return _className;
        }

}
```


В качестве полей класса используются следующие переменные: xPos, yPos, _height, _weight – координаты вершины треугольника, высота и ширина. Эти переменные должны быть у каждого объекта (прямоугольника) разные. Кроме этого, есть два поля одинаковые для всех прямоугольников − _angle (угол прямоугольника) и имя класса − _className. Эти переменные объявлены как статические. Все поля закрыты для доступа извне (модификатор доступа **private**).

Далее описывается специальный метод − **public Rectangle(...)** – его имя совпадает с именем класса – это конструктор объекта, специальный метод, который вызывается при создании экземпляра класса (объекта), в него передаются данные для инициализации всех полей объекта, в этом методе

рассчитываются все данные, необходимые объекту (_height и _weight – высота и ширина).

Задание

В классе Rectangle дописать следующие методы:

1) GetSquare – вычислить площадь;

2) GetXPos – вернуть координату X вершины прямоугольника;

3) GetYPos – вернуть координату Y вершины прямоугольника;

4) GetRectangleAngle – вернуть значение угла прямоугольника;

5) SetClassName – задать имя класса;

6) дописать метод Main таким образом, чтобы использовались все методы класса Rectangle.

Контрольные вопросы

1 Что такое класс? Приведите несколько определений.

2 Как создать класс на языке C#?

3 Как создать методы класса?

4 Что такое конструктор класса? Приведите пример.

5 Объясните назначение модификаторов доступа к полям класса private и public.

9 Лабораторная работа № 30. Разработка классов по индивидуальным вариантам

Цель работы: закрепление навыков создания классов на языке C#.

Задание

Реализовать класс (за основу принять класс Rectangle) (количество и содержание методов проработать самостоятельно) согласно варианту в таблице 9.1.

Таблица 9.1 – Варианты заданий

Номер варианта	Название объекта
1	Эллипс
2	Окружность
3	Трапеция
4	Треугольник
5	Параллелограмм
6	Ромб
7	Квадрат
8	Прямая линия
9	Точка

Контрольные вопросы

1 Какие поля, свойства и методы содержит Ваш класс?
2 Как реализован конструктор класса?
3 Как создать новый класс в Visual Studio?
4 В каком методе следует создать экземпляр класса и вызвать его собственные методы?
5 Назовите параметры методов класса, если таковые имеются.

10 Лабораторная работа № 31. Работа с файлами на C#

Цель работы: изучение принципов работы с файлами, изучение классов, связанных с файловыми операциями.

Задание

1 Записать в текстовый файл результат расчета функции f(y). Результат должен быть записан в виде двух столбцов – аргумента и значения функции от данного аргумента. Начало и конец диапазона, имя файла, а также шаг расчета вводить с клавиатуры. Варианты индивидуальных заданий приведены в таблице 10.1.

Таблица 10.1 – Варианты заданий

Вариант	Задание	Вариант	Задание
1	f(y) = y*y	6	f(y) = sin(y)*y
2	f(y) = y*y*y	7	f(y) = cos(y)*y
3	f(y) = cos(y)	8	f(y) = sin(y)*cos(y)*y
4	f(y) = sin(y)	9	f(y) = sin(y)*y*y
5	f(y) = sin(y)*cos(y)	10	f(y) = cos(y)*y*y

2 Считать файл, вывести на экран среднее арифметическое.

Контрольные вопросы

1 Какие классы используются для записи в файл и чтения из файла двоичных данных?
2 Какие классы используются для записи в файл и чтения из файла текстовых данных?
3 Как открыть поток для чтения данных из файла?
4 Как открыть поток для записи данных в файл?
5 Как закрыть поток?
6 Приведите примеры чтения и записи текстовых данных в файл.

11 Лабораторные работы № 32–33. Элементы форм WinForms – основные компоненты. Разработка приложений с формой. WinForms − дополнительные компоненты. Разработка приложений с формой и дополнительными элементами

Цель работы: ознакомление с основой разработки Windows-приложений на языке C# с использованием Windows Forms.

Задание 1

Общая часть задания: написать Windows-приложение, заголовок главного окна которого содержит ФИО, группу и номер варианта. В программе должна быть предусмотрена обработка исключений, возникающих из-за ошибочного ввода пользователя.

Вариант 1

Создать меню с командами Input**,** Calc и Exit.

При выборе команды Input открывается диалоговое окно, содержащее:

− три поля типа TextBox для ввода длин трех сторон треугольника;

− группу из двух флажков (Периметр и Площадь) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность:

− ввода длин трех сторон треугольника;

− выбора режима с помощью флажков: подсчет периметра и/или площади треугольника.

При выборе команды Calc открывается диалоговое окно с результатами. При выборе команды Exit приложение завершается.

Вариант 2

Создать меню с командами Size, Color, Paint, Quit.

Команда Paint недоступна. При выборе команды Quit приложение завершается. При выборе команды Size открывается диалоговое окно, содержащее:

− два поля типа TextBox для ввода длин сторон прямоугольника;

− группу из трех флажков (Red, Green, Blue) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность:

− ввода длин сторон прямоугольника в пикселах в поля ввода;

− выбора его цвета с помощью флажков.

После задания параметров команда Paint становится доступной.

При выборе команды Paint в главном окне приложения выводится прямоугольник заданного размера и сочетания цветов или выдается сообщение, если введенные размеры превышают размер окна.

Вариант 3

Создать меню с командами Input, Work, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Input открывается диалоговое окно, содержащее:

− три поля ввода типа TextBox с метками Radius, Height, Density;

− группу из двух флажков (Volume, Mass) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность:

− ввода радиуса, высоты и плотности конуса;

− выбора режима с помощью флажков: подсчет объема и/или массы конуса.

При выборе команды Work открывается окно сообщений с результатами.

Вариант 4

Создать меню с командами Input, Calc, Draw, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Input открывается диалоговое окно, содержащее:

− поле ввода типа TextBox с меткой Radius;

− группу из двух флажков (Square, Length) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность:

− ввода радиуса окружности;

− выбора режима с помощью флажков: подсчет площади круга (Square) и/или длины окружности (Length).

При выборе команды Calc открывается окно сообщений с результатами. При выборе команды Draw в центре главного окна выводится круг введенного радиуса или выдается сообщение, что рисование невозможно (если диаметр превышает размеры рабочей области).

Вариант 5

Создать меню с командами Input, Calc, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Input открывается диалоговое окно, содержащее:

− три поля ввода типа TextBox с метками Number 1, Number 2, Number 3;

− группу из двух флажков (Summ, Least multiple) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность ввода трех чисел и выбора режима вычислений с помощью флажков: подсчет суммы трех чисел (Summ) и/или наименьшего общего кратного двух первых чисел (Least multiple). При выборе команды Calc открывается диалоговое окно с результатами.

Вариант 6

Создать меню с командами Input, Calc, Quit.

Команда Calc недоступна. При выборе команды Quit приложение завершается. При выборе команды Input открывается диалоговое окно, содержащее:

− два поля ввода типа TextBox с метками Number 1, Number 2;

− группу из трех флажков (Summa, Max divisor, Multiply) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность:

− ввода двух чисел;

− выбора режима вычислений с помощью флажков (можно вычислять в любой комбинации такие величины, как сумма, наибольший общий делитель и произведение двух чисел).

При выборе команды Calc открывается окно сообщений с результатами.

Вариант 7

Создать меню с командами Begin, Help, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Begin открывается диалоговое окно, содержащее:

− поле ввода типа TextBox с меткой input;

− метку типа Label для вывода результата;

− группу из трех переключателей (2, 8, 16) типа RadioButton;

− две кнопки типа Button – Do и OK.

Обеспечить возможность:

− ввода числа в десятичной системе в поле input;

− выбора режима преобразования с помощью переключателей: перевод в двоичную, восьмеричную или шестнадцатеричную систему счисления.

При щелчке на кнопке Do должен появляться результат перевода.

Вариант 8

Создать меню с командами Input color, Change, Exit, Help.

При выборе команды Exit приложение завершает работу. При выборе команды Input color открывается диалоговое окно, содержащее:

− три поля ввода типа TextBox с метками Red, Green, Blue;

− группу из двух флажков (Left, Right) типа CheckBox;

− кнопку типа Button.

Обеспечить возможность ввода RGB-составляющих цвета. При выборе команды Change цвет главного окна изменяется на заданный (левая, правая или обе половины окна в зависимости от установки флажков).

Вариант 9

Создать меню с командами Input size, Choose, Change, Exit.

При выборе команды Exit приложение завершает работу. Команда Change недоступна. При выборе команды Input size открывается диалоговое окно, содержащее:

− два поля ввода типа TextBox с метками Size x, Size y;
− кнопку типа Button.

При выборе команды Choose открывается диалоговое окно, содержащее:
− группу из двух переключателей (Increase, Decrease) типа RadioButton;
− кнопку типа Button.

Обеспечить возможность ввода значений в поля Size x и Size y. Значения интерпретируются как количество пикселов, на которое надо изменить размеры главного окна (увеличить или уменьшить в зависимости от положения переключателей).

После ввода значений команда Change становится доступной. При выборе этой команды размеры главного окна увеличиваются или уменьшаются на введенное количество пикселов.

Вариант 10

Создать меню с командами Begin, Work, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Begin открывается диалоговое окно, содержащее:
− поле ввода типа TextBox с меткой Input word;
− группу из двух переключателей (Upper case, Lower case) типа RadioButton;
− кнопку типа Button.

Обеспечить возможность ввода слова и выбора режима перевода в верхний или нижний регистр в зависимости от положения переключателей. При выборе команды Work открывается диалоговое окно с результатом перевода.

Вариант 11

Создать меню с командами Input color, Change, Clear.

При выборе команды Input color открывается диалоговое окно, содержащее:
− группу из двух флажков (Up, Down) типа CheckBox;
− группу из трех переключателей (Red, Green, Blue) типа RadioButton;
− кнопку типа Button.

Обеспечить возможность:
− выбора цвета с помощью переключателей;
− ввода режима, определяющего, какая область закрашивается: все окно, его верхняя или нижняя половина.

При выборе команды Change цвет главного окна изменяется на заданный (верхняя, нижняя или обе половины в зависимости от введенного режима). При выборе команды Clear восстанавливается первоначальный цвет окна.

Вариант 12

Создать меню с командами Translate, Help, About, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Translate открывается диалоговое окно, содержащее:
− поле ввода типа TextBox с меткой Binary number;
− поле ввода типа TextBox для вывода результата (read-only);

− группу из трех переключателей (8, 10, 16) типа RadioButton;
− кнопку Do типа Button.
Обеспечить возможность:
− ввода числа в двоичной системе в поле Binary number;
− выбора режима преобразования с помощью переключателей: перевод в восьмеричную, десятичную или шестнадцатеричную систему счисления.
При щелчке на кнопке Do должен появляться результат перевода.

Контрольные вопросы

1 Какой тип проекта нужно выбрать в Visual Studio для работы с формой?
2 Назовите известные вам элементы управления на форме и покажите их на панели ToolBox.
3 Для чего используется элемент TextBox и как выполнить ввод данных с проверкой типа методом TryParse? Приведите пример.
4 Для чего используется элемент CheckBox? Как определить его состояние?
5 Для чего используется Вкладки TabControl и переключатели RadioButton?
6 Как создать на форме меню с командами?

12 Лабораторные работы № 34–35. Проектирование таблиц. Создание, связывание и заполнение таблиц базы данных

Цель работы: проектирование и разработка таблиц базы данных по тематике курсового проектирования.

Выполнение работы

Произвести анализ предметной области, проектирование и заполнение таблиц информацией для базы данных в соответствии с заданием на курсовое проектирование. Организовать связи между таблицами базы данных.

Контрольные вопросы

1 Как в СУБД ACCESS создать новую таблицу? Назовите все способы создания новой таблицы.
2 Что такое ключевое поле и как оно задается?
3 Назовите типы связей между таблицами БД.
4 Как установить связи между двумя таблицами в СУБД ACCESS?
5 Как удалить связь между двумя таблицами?

13 Лабораторная работа № 36. Разработка форм для работы с базой данных

Цель работы: проектирование и разработка форм для базы данных по тематике курсового проектирования.

Выполнение работы

Произвести проектирование и создание форм для базы данных в соответствии с заданием на курсовое проектирование.

Контрольные вопросы

1 Что такое форма?
2 Какие способы создания форм Вы знаете?
3 Как создать форму в режиме конструктора в MS ACCESS?
4 Как создать форму на базе связанных таблиц?
5 Как использовать элементы управления на форме?

14 Лабораторная работа № 37. Создание запросов по базе данных

Цель работы: проектирование и разработка запросов для базы данных по тематике курсового проектирования.

Выполнение работы

Произвести создание запросов разных видов для базы данных в соответствии с заданием на курсовое проектирование.

Контрольные вопросы

1 Как создать запрос в конструкторе?
2 Как при создании запроса установить условия отбора?
3 Как используются выражения в запросе?
4 Как сортируются данные в запросе?
5 Как осуществить поиск данных в диапазоне значений?
6 Как выполнить (запустить) запрос?
7 Как создать запрос на базе связанных таблиц?

15 Лабораторная работа № 38. Создание отчетов по базе данных

Цель работы: проектирование и разработка отчетов для базы данных по тематике курсового проектирования.

Выполнение работы

Произвести создание детальных и итоговых отчетов для базы данных в соответствии с заданием на курсовое проектирование.

Контрольные вопросы

1 Назовите разделы отчета.
2 Создание детального отчета без повторов.
3 Создание отчета по сгруппированным данным.
4 Использование выражения для описания сумм по группам.
5 Изменение порядка сортировки данных.
6 Распечатка групп данных без разрывов.

Список литературы

1 **Шилдт, Г.** C# 4.0. Полное руководство / Г. Шилдт. − Москва: Вильямс, 2015. – 1056 с.

2 Язык программирования C# / А. Хейлсберг [и др.] – Санкт-Петербург : БХВ-Петербург, 2011. – 784 с.

3 **Культин, Н. Б.** Microsoft Visual C# в задачах и примерах / Н. Б. Культин. – Санкт-Петербург : БХВ-Петербург, 2009. – 576 с. : ил.

4 **Гуриков, С. Р.** Информатика : учебник / С. Р. Гуриков. – Москва : ФОРУМ ; ИНФРА-М, 214. – 464 с.