Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«МАТИ — Российский государственный технологический университет
имени К. Э. Циолковского» (МАТИ)

Кафедра «Моделирование систем и информационные технологии»

ИСПОЛЬЗОВАНИЕ СОСТАВНЫХ ТИПОВ ДАННЫХ ПАСКАЛЯ. ЧАСТЬ 2. МАССИВЫ, ЗАПИСИ И ФАЙЛЫ

Методические указания к лабораторной работе по курсу
«Алгоритмические языки и программирование»

Составитель В. В. Лидовский

Москва 2013

Л Использование составных типов данных паскаля. Часть 2. Массивы, записи и файлы: методические указания к лабораторной работе по курсу «Алгоритмические языки и программирование». / сост. В. В. Лидовский. — М.: МАТИ, 2013. — 12 с.

ВВЕДЕНИЕ

Настоящие методические указания предназначены для обеспечения учебного процесса студентов дневной формы обучения по направлению подготовки 230100.62 “Информатика и вычислительная техника” по профилю подготовки 230102.62 “Автоматизированные системы обработки информации и управления” при выполнении лабораторной работы по предмету “Алгоритмические языки и программирование”.

Цель лабораторной работы: изучить методы программирования на языке паскаль для работы со регулярным, комбинированным и файловым типами.

1. СИСТЕМНЫЕ ТРЕБОВАНИЯ

Для выполнения лабораторной работы необходимы следующие системные компоненты:

а) компьютер, работающий под управлением операционной системы Linux;

б) компилятор Free Pascal.

2. МАССИВЫ

2.1. Определение и инициализация

Для выбранного типа можно построить основанный на нем регулярный тип, считая последний занумерованным набором из элементов исходного типа. Регулярный тип описывается при помощи служебного слова array.

Примеры:

```
array[0..10] of array[0..10] of char     (* массив из 11 элементов типа
                                 массив из 11 символов - матрица 11x11 *)
array[0..10,0..10]of char          (* это описание того же самого типа *)
array[char]of integer     (* массив из 256 целого числа с индексами от
                                                          #0 до #255 *)
```

Диапазоны в квадратных скобках задают возможные значения индексов элементов массива. Количество индексов — это размерность массива. В первых двух примерах массивы двумерные, а в третьем — одномерный.

Следующие две функции предназначены для работы с данными типа массив:

low(a) — возвращает наименьшее значение первого индекса массива, где a — это имя типа или переменная, например, если matrix — это тип array[1..10,2..12]of byte, то low(matrix)=1, low(matrix[1])=2;

high(a) — отличается от low только тем, что возвращает наибольшее значение, например, high(matrix)=10, high(matrix[5])=12.

Хотя паскаль не поддерживает констант-литералов типа массив, для инициализации данных можно использовать изображение массива. Например, массив-матрицу 3×4 из неотрицательных чисел можно инициализировать следующей конструкцией.

```
const
  matrix: array[1..3, 1..4]of word = ((2, 5, 7, 4),
                                      (7, 8, 5, 2),
                                      (0, 4, 4, 7));
```

2.2. Операции с матрицами

Матрицу можно умножить на число. В результате такого умножения все все элементы исходной матрицы умножатся на это число.

Матрицы равного размера можно складывать. В матрице-сумме все элементы — это суммы элементов исходных матриц.

Если у одной матрицы число строк совпадает с числом столбцов другой, то первую матрицу можно умножить на вторую по формуле.

$$\begin{bmatrix} a_{11} & \dots & a_{1m} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nm} \end{bmatrix} \times \begin{bmatrix} b_{11} & \dots & b_{1s} \\ \dots & \dots & \dots \\ b_{m1} & \dots & a_{mk} \end{bmatrix} = \begin{bmatrix} c_{11} & \dots & c_{1s} \\ \dots & \dots & \dots \\ c_{n1} & \dots & c_{ns} \end{bmatrix},$$

где каждый элемент $c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$. Например,

$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \\ 4 & 5 & 6 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 \cdot 2 + 1 \cdot 3 + 2 \cdot 4 & 0 \cdot 3 + 1 \cdot 4 + 2 \cdot 5 & 0 \cdot 4 + 1 \cdot 5 + 2 \cdot 6 \\ 3 \cdot 2 + 4 \cdot 3 + 5 \cdot 4 & 3 \cdot 3 + 4 \cdot 4 + 5 \cdot 5 & 3 \cdot 4 + 4 \cdot 5 + 5 \cdot 6 \\ 6 \cdot 2 + 7 \cdot 3 + 8 \cdot 4 & 6 \cdot 3 + 7 \cdot 4 + 8 \cdot 5 & 6 \cdot 4 + 7 \cdot 5 + 8 \cdot 6 \end{bmatrix} =$$

$$= \begin{bmatrix} 11 & 14 & 17 \\ 38 & 50 & 62 \\ 65 & 86 & 107 \end{bmatrix}.$$

Другой пример —

$$\begin{bmatrix} 0 & 1 & 2 & 4 \\ 2 & 3 & 4 & 5 \end{bmatrix} \times \begin{bmatrix} 2 & 3 \\ 3 & 4 \\ 4 & 5 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 28 & 30 \\ 39 & 43 \end{bmatrix}.$$

Один из эффективных методов расчета определителя квадратной матрицы основывается на ее преобразовании к треугольному виду по специальным правилам. Применение этих правил для матрицы $A = (a_{ij})$ из n строк приводит к следующему алгоритму.

Шаг 1. Номер строки $i = 1$;

Шаг 2. Если $a_{ii} = 0$, то если $\forall\ j > i\ a_{ji} = 0$, то определитель равен 0, иначе если $\exists\ j > i\ a_{ji} \neq 0$, то меняем строки i и j местами и, кроме того, меняем знак определителя;

Шаг 3. Меняем компоненты каждой j-й строки ($j > i$) по следующей формуле $a_{jk} = a_{jk} - a_{ik} * a_{ji}/a_{ii}$, меняя k от n до i;

Шаг 4. Увеличиваем i на 1;

Шаг 5. Если $i < n$, то переход к шагу 2, иначе завершение работы — матрица приведена к треугольному виду.

Рассмотрим применение этого алгоритма. Пусть

$$A = \begin{bmatrix} 0 & 11 & 5 \\ 1 & 2 & 4 \\ 7 & 8 & 5 \end{bmatrix}.$$

Начинаем работать с первой строкой ($i = 1$). Элемент $a_{11} = 0$, а $a_{21} = 1 \neq 0$ — поэтому меняем первые две строки местами. Меняем знак определителя. Получаем матрицу

$$\begin{bmatrix} 1 & 2 & 4 \\ 0 & 11 & 5 \\ 7 & 8 & 5 \end{bmatrix}.$$

Далее, согласно шагу 3, нужно установить под 1 в первом столбце нули. Вторая строка уже имеет требуемый вид, а от третьей нужно отнять первую, умноженную на 7. Получаем

$$\begin{bmatrix} 1 & 2 & 4 \\ 0 & 11 & 5 \\ 0 & -6 & -23 \end{bmatrix}.$$

Переходим к работе со второй строкой, устанавливаем $i = 2$ согласно шагу 4. Элемент $a_{22} \neq 0$, поэтому переходим к шагу 3 и отнимаем от последней строки вторую, умноженную на $-\frac{6}{11}$. Матрица приведена к треугольному виду —

$$\begin{bmatrix} 1 & 2 & 4 \\ 0 & 11 & 5 \\ 0 & 0 & -\frac{223}{11} \end{bmatrix}.$$

Определитель равен произведению элементов на главной диагонали преобразованной матрицы. С учетом однократного изменения знака $det(A) = 223$.

Определитель целочисленной матрицы — целое число.

3. ЗАПИСИ

3.1. Определение и инициализация

Этот тип позволяет совмещать в одном элементе данных разнотипные величины. Он описывается при помощи служебного слова record. Рассмотрим, например, тип, определяющий прямоугольники. Для определения уникального прямоугольника можно дать ему имя и координаты двух точек: верхнего левого угла и нижнего правого. Такой тип можно описать следующей записью.

```
type
   rectangle = record
      id: string;
      x_ul, y_ul, x_lr, y_lr: integer
                                (* ul - upper left, lr - lower right *)
   end;
```

Данные типа запись можно инициализировать значениями в скобках после имени соответствующего поля. Например, переменную rect введенного типа прямоугольник можно инициализировать следующей конструкцией.

```
const
   rect: rectangle = (id: 'rect1';
      x_ul: 10; y_ul: 12; x_lr: 110; y_lr: 220);
```

С записями может быть полезен оператор присоединения with. Например, площадь приведенного прямоугольника естественно посчитать следующим оператором.

```
with rect do
   square := (x_lr - x_ul)*(y_lr - y_ul)
```

3.2. Структуры данных, списки

Записи в соединении с указателями позволяют естественным образом создавать произвольные структуры данных: списки, очереди, деревья, стеки, сети, ... Например, следующие типы

```
PointerToListElement = ^ListElementType;
ListElementType = record
   data: word;
   next, previous: PointerToListElement
end;
```

можно использовать для работы с двунаправленным списком. Тип ListElementType — это тип элементов списка, а указатели типа PointerToListElement позволяют выбирать элементы списка.

В циклическом однонаправленном списке последний элемент должен указывать на первый как на следующий. При работе с таким списком необходимо наличие двух указателей: на первый и на текущий элементы.

4. ФАЙЛЫ

Файл — это последовательность элементов одного типа. Данные файла существуют независимо от программы. Файловый тип описывается при помощи служебного слова file. Например, переменная для работы с файлом из целых чисел может быть описана следующей декларацией.

```
var
   numbers: file of word;
```

Работу с файлом начинают со связывания файловой переменной с элементом файловой системы. Для этого используется процедура assign. Если нужно работать с существующим файлом, то используют операцию reset. Если нужно создать новый файл, то — rewrite. Для записи в файл предназначена процедура write, а для чтения данных из файла — read. Для выбора заданной позиции в файле используют операцию seek. Работа с файлом завершается командой close.

Работа с текстовым файлом имеет особенности. Тип текстового файла описывается идентификатором text. При считывании данных обеспечивается автоматическое преобразование строк текста в данные заданного типа. Например, если n — переменная целого типа, то при считывании процедурой read(f, n) из текстового файла f строки '123' она будет преобразована в число 123, которое присвоится n.

Процедура readln применима только к текстовым файлам. Она считывает данные построчно и игнорирует данные за пределами сопоставимых со списком своих аргументов.

5. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

Необходимо программно решить три задачи согласно установленному варианту задания.

Первая задача состоит в подсчете значения заданных матричных выражений. Вторая — в вычислении определителя заданной матрицы. Третья — в считывании числовых данных из заданного текстового файла, записи этих данных в заданный бинарный файл, затем образования из этих данных циклического однонаправленного списка и обеспечение последующего интерактивного поиска заданных чисел среди введенных из файла числовых данных.

Программа для решения первой задачи должна содержать процедуры для умножения матриц, сложения матриц, умножения матрицы на число,

распечатки матрицы. Все арифметические операции следует проводить в поле классов вычетов по модулю 17.

Программа для решения второй задачи также должна содержать и функцию для вычисления определителя.

Программа для решения третьей задачи должна содержать функцию поиска числа в списке, которая должна возвращать номер найденного элемента или 0, если искомого числа в списке нет. Кроме того, в программе должна быть процедура, создающая список.

6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какой самый глубокий уровень вложения в циклы при расчете определителя по приведенному алгоритму?
2. Может ли для расчета определителя целочисленной матрицы по приведенному алгоритму использоваться массив целых чисел?
3. Может ли в результате произведения матриц получиться целое число?
4. Может ли файл содержать данные типа файл?
5. Какой разделитель используется между полями в изображении записи в синтаксической конструкции инициализации данных?
6. Всегда ли в паскале объявление типа предшествует его использованию?
7. Является ли использование записей лучшим решением при работе со сложными структурами данных? Если нет, то какой тип предпочтительнее?
8. Как, используя текстовый файл, реализовать процедуры, заменяющие стандартные процедуры str и val?
9. Чем тип file of char похож на тип text? В чем они различаются?
10. Что общего у типов массив и файл? В чем их отличия?
11. При реализации бинарных матричных операций процедурой можно ли возвращать результат через параметр, передаваемый по значению?
12. Какие типы подходят для данных файла numbers.bin?
13. Насколько файл numbers.txt больше файла numbers.bin?
14. Какие операции существуют для данных комбинированного типа?

7. ВАРИАНТЫ РАБОТ

Лабораторная работа по теме “Использование составных типов данных паскаля. Часть 2. Массивы, записи и файлы” имеет 7 вариантов заданий.

Данные для первых двух задач определяются следующей таблицей.

Вариант	*Выражения*	*Определитель*
1	$A^2 - B$, $2AB - E$	$A - E$
2	$A^3 - E$, $4A(B - E)$	$B - E$
3	$A^2 + B$, $2AB + E$	$2A - E$
4	$A(E - B)$, $(B - 3E)A$	$B + E$
5	$A^2 - B^2$, $2A - B^2$	$A + E$
6	$2AB - E$, $A^2 + B$	$A + B + E$
7	$A(B - A)$, $AB - 2E$	$A + 2E$

Каждый элемент матрицы A равен сумме номеров своих столбца и строки, т.е. $a_{ij} = i + j$. Первая строка матрицы B заполняется числами от 1 до 7, вторая — от 8 до 14, 3-я — от 15 до 4 и т.д., т.е. идет нумерация слева-направо и сверху-вниз по модулю 17. E — это единичная матрица, у которой на главной диагонали единицы, а прочие элементы — нули.

Текстовый файл с данными для третьей задачи должен называться numbers.txt и состоять из не менее чем 10 строк, содержащих произвольные неотрицательные целые числа от 0 до 100. Записываемый файл с бинарными данными должен называться numbers.bin — на каждое число в нем должно выделяться по одному байту.

ЛИТЕРАТУРА

1. *Апатенок Р.Ф., Маркина А.М., Попова Н.В., Хейнман В.Б.* Элементы линейной алгебры и аналитической геометрии — Минск: Вышэйшая школа, 1986. — 272 с.
2. *Боревич З. И.* Определители и матрицы — М.: Наука, 1988. — 194 с.
3. *Бородин Ю.С., Вальвачев А.Н., Кузьмич А.И.* Паскаль для персональных компьютеров — Минск: Вышэйшая школа, БФ ГИТМП “НИКА”, 1991. — 369 с.
4. *Зуев Е.А.* Язык программирования Turbo Pascal 6.0 — М.: Унитех, 1992. — 298 с.
5. *Немнюгин С. А.* Turbo Pascal. Программирование на языке высокого уровня — СПб.: Питер, 2008. — 544 с.
6. *Петерсен Р.* Linux: полное руководство — Киев: “Ирина” BHV, 2000. — 642 с.
7. *Фараонов В. В.* Turbo Pascal: учебное пособие — СПб.: Питер, 2010. — 368 стр.

ПРИЛОЖЕНИЕ

ПРИМЕР РАБОТЫ С ПРОГРАММОЙ ПРИ РЕШЕНИИ 3-Й ЗАДАЧИ

```
12 чисел из файла 'numbers.txt' считаны
Данные сохранены в файле 'numbers.bin'
Список создан

Для выхода введите нечисло, например, строку stop
Введите искомое число: 5
Число 5 найдено в позиции 3

Введите искомое число: 74
Число 74 в списке не найдено

Введите искомое число: 8
Число 8 найдено в позиции 8

Введите искомое число: stop
```

СОДЕРЖАНИЕ

ИСПОЛЬЗОВАНИЕ СОСТАВНЫХ ТИПОВ ДАННЫХ ПАСКАЛЯ. ЧАСТЬ 2. МАССИВЫ, ЗАПИСИ И ФАЙЛЫ

Методические указания к лабораторной работе по курсу
«Алгоритмические языки и программирование»

Составитель:
Владимир Викторович **Лидовский**

Технический редактор *Е. Е. Костылева*

Оригинал-макет подготовлен в пакете Plain-TEX

Подписано в печать _._.2013
Усл. печ. л. 1,86. Формат 60×84 $^1/_{16}$.
Тираж 70 экз. Заказ № _
Издательский-типографский центр МАТИ
109383, Москва, ул. Полбина, 45