**МТУ**

**ИКБиСП**

# 

# КАФЕДРА КБ-4

#### ДИСЦИПЛИНА «Программирование»

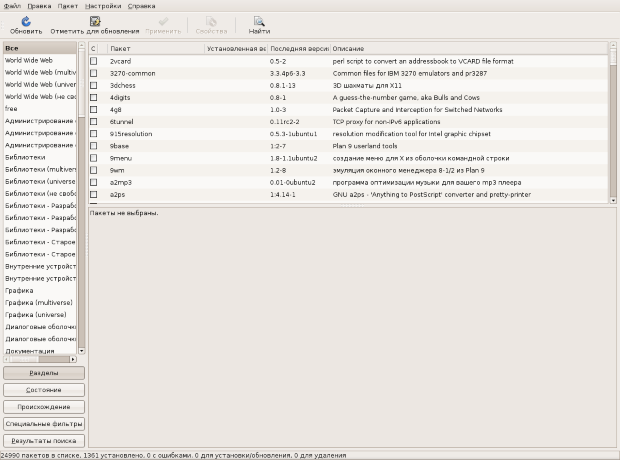
# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к проведению лабораторных работ

#### Москва 2016

1. **Среда программирования Free Pascal**

Рассмотрим процесс установки компилятора Free Pascal в ОС Linux. Для установки программ в операционной системе Linux служит менеджер пакетов. В разных дистрибутивах Linux используются различные менеджеры пакетов, например, в ALT Linux можно воспользоваться программой Synaptic. Окно Synaptic представлено на [рис. 1.1](http://www.intuit.ru/studies/courses/3488/730/lecture/25786?page=2#image.1.1). В школьной линейке дистрибутивов ALT Linux Free Pascal и Lazarus присутствуют сразу после установки операционной системы.

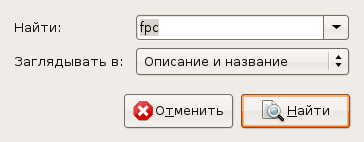
[](http://www.intuit.ru/EDI/26_04_15_9/1430000323-31595/tutorial/708/objects/1/files/1_1.png)

*Рис. 1.1. Менеджер пакетов Synaptic*

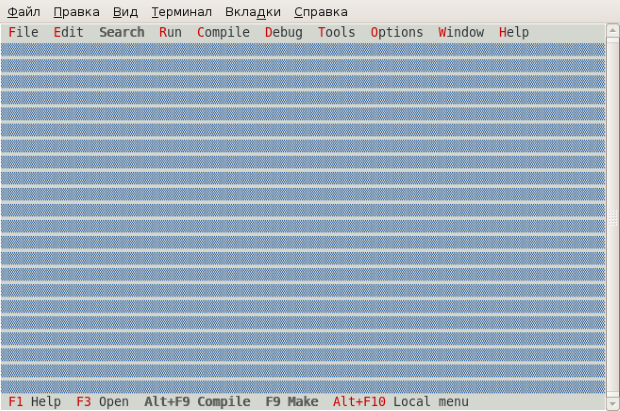
Обратите внимание, что для установки программ необходимо установить список источников программ (список репозиториев4).

Для установки Free Pascal в окне Synaptic ([см. рис. 1.1](http://www.intuit.ru/studies/courses/3488/730/lecture/25786?page=2#image.1.1)) необходимо щёлкнуть по кнопке Найти и в открывшемся окне ввести fpc ([см. рис. 1.2](http://www.intuit.ru/studies/courses/3488/730/lecture/25786?page=2#image.1.2)). Менеджер программ находит программу FreePascal, после чего в окне Synaptic необходимо отметить программы fpc (Free Pascal Compiler Meta Package) для установки (с помощью контекстного меню или с помощью кнопки Отметить для обновления) и начать установку, щёлкнув по кнопке Применить. После этого начнётся процесс загрузки пакетов из Интернета и их установки.

В состав метапакета fpс входит компилятор языка Free Pascal fpc и среда разработки fp-ide. Для запуска среды разработки в Linux необходимо просто в терминале набрать fp. На [рис. 1.3](http://www.intuit.ru/studies/courses/3488/730/lecture/25786?page=2#image.1.3) представлено окно среды разработки программ на языке Free Pascal в ОС Linux.

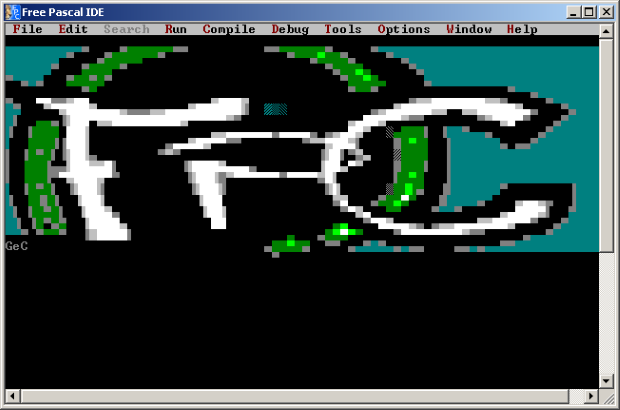


*Рис. 1.2. Окно поиска компилятора Free Pascal в Synaptic*



*Рис. 1.3. Среда программирования Free Pascal в ОС Linux*

Для установки Free Pascal в ОС Windows необходимо запустить скачанный со страницы [загрузки](http://www.freepascal.org/down/i386/var) инсталяционный файл. Первое диалоговое окно сообщит о начале процесса установки Free Pascal на компьютер. Для продолжения установки во всех следующих окнах нужно выбирать кнопку Next, для возврата к предыдущему шагу — кнопку Back, а для прерывания процесса установки — кнопку Cancel. В следующем окне нужно определить путь для установки Free Pascal. По умолчанию установка происходит в корневой каталог диска С. Для выбора другого пути установки можно воспользоваться кнопкой Browse... Кроме того, в этом окне выводится информация о количестве свободного места на диске. В следующих четырёх окнах пользователь сможет выбрать из списка тип установки: Full Installation (полная), Minimum Installation (минимальная), Custom Installation (выбор компонентов), указать название устанавливаемого приложения в главном меню, выбрать типы файлов, поддерживаемых средой, и начать процесс установки Free Pascal, нажав кнопку Install. Контролировать процесс установки можно с помощью линейного индикатора.



*Рис. 1.4. Окно компилятора Free Pascal*

Запуск среды программирования Free Pascal в Windows можно осуществить из главного меню: Пуск — Программы — Free Pascal — Free Pascal. На экране появится окно, представленное на [рис. 1.4](http://www.intuit.ru/studies/courses/3488/730/lecture/25786?page=2#image.1.4).

Установив пакет Free Pascal, мы получили компилятор и среду программирования.

Компилятор Free Pascal работает в командной строке. Для того чтобы создать исполняемый файл из текста программы, написанного на языке Pascal, необходимо выполнить команду

fpc name.pas

Здесь fpc — имя исполняемого файла компилятора командной строки Free Pascal, name.pas — имя файла с текстом программы. В результате в L inux будет создан исполняемый файл с именем name (в Windows имя исполняемого файла — name.exe).

При использовании компилятора fpc после компиляции автоматически происходит компоновка программы (запуск компоновщика make).

Технология работы с компилятором Free Pascal может быть такой: набираем текст программы в стандартном текстовом редакторе, затем в терминале запускаем компилятор, после исправления синтаксических ошибок запускаем исполняемый файл. При такой технологии работы с компилятором, необходимо не забывать сохранять текст программы, иначе при запуске компилятора будет компилироваться старая версия текста программы.

Однако среда программирования позволяет значительно упростить процесс разработки программ. В состав среды программирования Free Pascal входит текстовый редактор, транслятор и отладчик. Рассмотрим их работу подробнее.

* + 1. **Работа в текстовом редакторе Free Pascal**

С помощью редактора Free Pascal можно создавать и редактировать тексты программ. После открытия пустого окна (File — New) или загрузки текста программы (File — Open) мы находимся в режиме редактирования, что подтверждается наличием в окне курсора (небольшого мигающего прямоугольника). Для перехода из режима редактирования к главному меню нужно нажать клавишу F10, обратно — Esc. Кроме того, этот переход можно осуществить щелчком мыши либо по строке главного меню, либо по полю редактора.

Редактор Free Pascal обладает возможностями, характерными для большинства текстовых редакторов. Остановимся на некоторых особенностях.

Работа с фрагментами текста (блоками) в редакторе Free Pascal может осуществляться с помощью главного меню и функциональных клавиш. Выделить фрагмент текста можно с помощью клавиши Shift и клавиш перемещения курсора (стрелок).

В главном меню для работы с фрагментами текста предназначены команды пункта редактирования Edit:

Copy (Ctrl+C) — копировать фрагмент в буфер;

Cut (Ctrl+X) — вырезать фрагмент в буфер;

Paste (Ctrl+V) — вставить фрагмент из буфера;

Clear (Ctrl+Del) — очистить буфер;

Select All — выделить весь текст в окне;

Unselect — отменить выделение.

Команды Copy и Cut применяют только к выделенным фрагментам текста. Кроме того, пункт меню Edit содержит команды Undo и Redo, с помощью которых можно отменять и возвращать выполненные действия.

Комбинации клавиш, предназначенные для работы с блоком, таковы:

Ctrl+K+B — пометить начало блока;

Ctrl+K+K — пометить конец блока;

Ctrl+К+Т — пометить в качестве блока слово слева от курсора;

Ctrl+K+Y — стереть блок;

Ctrl+K+C — копировать блок в позицию, где находится курсор;

Ctrl+K+V — переместить блок в позицию, где находится курсор;

Ctrl+K+W — записать блок в файл;

Ctrl+K+R — прочитать блок из файла;

Ctrl+K+P — напечатать блок;

Ctrl+К+Н — снять пометку блока; повторное использование Ctrl+К+Н

вновь выделит блок.

Работа с файлами в среде Free Pascal осуществляется с помощью команд File главного меню и функциональных клавиш:

New — открыть окно для создания новой программы;

Open (F3) — открыть ранее созданный файл;

Save (F2) — сохранить созданный файл;

Save As — сохранить файл под другим именем;

Exit (Alt+X) — выйти из среды программирования.

При создании новой программы ей по умолчанию присваивается стандартное имя NONAMEOO.PAS (NO NAME — нет имени).

При первом сохранении файла пользователю будет предложено ввести его имя. При повторном сохранении файл сохраняется под тем же именем. Команда Save As аналогична первому сохранению. Если файл не был сохранён, то при попытке завершить работу со средой, появится запрос о необходимости сохранить изменения в файле. При открытии ранее созданного файла его имя выбирают из списка существующих файлов.

В редакторе Free Pascal допускается работа с несколькими окнами. Переключаться между окнами можно двумя способами:

для переключения в окно с номером от первого до девятого нажать комбинацию клавиш Alt+i, где i — номер окна (например Alt+5 — вызов пятого окна);

для вывода списка окон на экран нажать комбинацию клавиш Alt+0, появится список активных окон, в котором будет необходимо выбрать нужное и нажать Enter.

* + 1. **Запуск программы в среде Free Pascal и просмотр результатов**

После того как текст программы набран, его следует перевести в машинный код. Для этого необходимо вызвать транслятор с помощью команды Compile — Compile (комбинация клавиш Alt+F9). На первом этапе транслятор проверяет наличие синтаксических ошибок. Если в программе нет синтаксических ошибок, то на экране сообщается о количестве строк транслированной программы и объёме доступной оперативной памяти.

Если на каком-либо этапе транслятор обнаружит ошибку, то в окне редактора курсор укажет ту строку программы, в которой ошибка была обнаружена. При этом в верхней строке редактора появится краткое диагностическое сообщение о причине ошибки.

Для запуска транслированной программы необходимо выполнить команду Run — Run (комбинация клавиш Ctrl+F9), после чего на экране появляется окно командной строки, в котором пользователь и осуществляет диалог с программой. После завершения работы программы вновь появляется экран среды Free Pascal.

**2. Лабораторная работа № 1**

**Условный оператор и программирование разветвляющихся алгоритмов.**

Цель работы: Изучение приемов программирования разветвляющихся алгоритмов с использованием оператора сравнения и логических операций над данными типа boolean.

**Пример первого задания.**

Составить программ*у* вычисления функции:

х2+1, x<0

Y= x-2,1, x>п/2

Sinx, 0<=x<=п/2

Для произвольного X и вычислить значения Y(x) в точках:

Х1= -2,5 х2=-0,5 х3=1,5

Program Func;

var

x,y:real;

BEGIN

Write(‘введите х=’); readln(x);

If x<0 then y := sqr(x)+1

else

if x>pi/2 then y := x-2.1

else

y := sin(x);

writeln( ' Y= ' , у: 6: 3)

END.

**Пример второго задания.**

Составить программу определения произведения двух наименьших *целых* чисел А,В,С (А<>В<>С<>0) .

*Program* mult;

var

max,a,b,с : integer;

mul : integer;

BEGIN

writeln('введите 3 целых числа А,В,С'); readln(a,b,c);

max:= a ;

if b>max then max:=b;

if c>max then max:=c;

mul:= (a \*b\*с) div max ;

writeln('произведение двух наименьших=',mul)

END.

**3. Лабораторная работа № 2*.***

Операторы цикла и программирование циклических алгоритмов.

Цель работы:

1. Изучение приемов программирования циклических алгоритмов с применением трех различных операторов цикла.

2. Изучение приемов программирования при работе с перечисляемыми типами.

**Пример первого задания.**

Найти сумму членов знакопеременного ряда.

S= (+3)/2-(+5)/4+…+ +35)/34- (+35)/36

Program sum\_;

var

a,i : integer;

s : геа1 ;

BEGIN

s:=0; {сумма}

i:=2; {начальное значение}

a:=l; {«мигалка»}

while i< = 36 do

begin

s:=s+a\*(sqrt(i)+(i+1))/i;

i:=i+2;

a := -a

end;

writeln(‘s= ‘,s:6:2)

END.

**Пример второго задания.**

Составить программу табулирования функции:

Ex+Sin2x\*cos(x-2), x<=1

Y=

2x+), x>1

на отрезке х [0,2] с шагом h

Program tab\_;

const

h=0.2;

var

x : real;

BEGIN

x**:**=0;

while x < = ( 2 + **h / 2)** do

begin

if x < = 1 then у:=exp(x) +sqr(sin(x))\*соs(x-2)

else

у:=еxр(ln(2)\*x)+sqrt(x+6);

writeln('x = ' , x : 5 : 2, ' у = ' , у : 7 : 2) ;

x:= x + h

end

END.

**Пример третьего задания.**

Определить по заданной дате (число, месяц, год.) дату следующего дня (число, название месяца, год).

Program data\_;

type

mes= (jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec);

day=1..31;

year=1900..2010;

var

d : day;

m : mеs**;**

g : year;

n : 1..12;

BEGIN

Write(‘введите дату D,N,G’); readln(d,n.g);

Case n of

1:m:=jan;

2:m:=feb;

3:m:=mar;

4:m:=apr;

5:m:=may;

6:m:=jun;

7:m:=jul;

8:m:=aug;

9:m:=sep;

10:m:=oct;

11:m:=nov;

12:m:=dec;

end;

Case m of

Jan, mar, may, jul, aug, oct:

If d=31 then

begin

d:=1; m:=succ(m)

end

else

inc(d);

apr, jun, sep, nov:

f d=30 then

begin

d:=1; m:=succ(m)

end

else

inc(d);

dec:

If d=31 then

begin

d:=1; m:=jan; inc(g)

end

else

inc(d);

feb:

if ((d=28) and (not((g mod 400=0) or (g mod 4=0) and (g mod 100<>0)))

or (d=29) then

begin

d:=1; m:=mar;

end

else

inc(d);

end; {case}

writeln(‘дата следующего дня: ‘);

write(d:2);

Case m of

jan : write(‘января’);

feb : write(‘февраля’);

mar : write(‘марта’);

apr : write(‘апреля’);

may : write(‘мая’);

jun :write(‘июня’);

ju1 : wгite( ‘июля‘ ) ;

aug : wгitе('августа' ) ;

sep :write('сентября');

oct : write('октября');

nov : write('ноября');

dec : write('декабря');

end;

write(g:5, ' года')

END.

**4. Лабораторная работа №3.**

**Программирование типичных задач при работе с массивами.**

Цель работы

1. Приемы программирования циклических алгоритмов для задач с обработкой одномерного и двумерного маcсивов.

2. Вывода элементов одномерного и двумерного маcсивов.

3. Изучение одного из видов сортировки одномерного массива.

**Пример первого задания.**

Дан массив целых чисел А ( 4 , 5). Переставить четвертый и пятый столбец местами.

Program change\_;

Var

a:array[1..4,1..5] of integer;

i,n:byte;

buf:integer;

BEGIN

Writeln(Введите элементы массива А (4\*5)’);

Randomize;

For i:=1 to 4 do

For n:=1 to 5 do

a[I,n]:= integer(random(100))-50;

writeln(‘исходный массив А до перестановки’);

For i:=1 to 4 do

begin

For n:=1 to 5 do

Write(a[I,n]:4);

Writeln

End;

For i:=1 to 4 do

Begin

Buf:=a[I,4];

a[I,4]:= a[I,5];

a[I,5]:= Buf

End;

writeln(‘исходный массив А после перестановки’);

For i:=1 to 4 do

begin

For n:=1 to 5 do

Write(a[I,n]:4);

Writeln

End

END.

**Пример второго задания.**

Ввести n чисел. Отсортировать числа по возрастанию.

Program Sогt \_;

type

size=l..100:

mas=array[size] of' integer;

var

arr\_: mas;

i,n,k: size;

Buf: integer;

BEGIN

Write(‘введите количество сортируемых элементов массива ’);

Readln(n);

Writeln(‘Введите элементы массива АRR\_’);

Randomize;

For i:=1 to n do

Arr\_[I]:= integer(random(100))-50;

writeln(‘исходный массив до перестановки’);

For i:=1 to n do

Write(arr\_[i]:4);

Writeln;

For i:=1 to n-1 do

For k:=i+1 to n do

If arr\_[i]> arr\_[k] then

begin

buf:= arr\_[i];

arr\_[i]:= arr\_[k];

arr\_[k]:=buf

End;

writeln(‘исходный массив после перестановки’);

For i:=1 to n do

Write(a[i]:4);

Writeln;

END.

**5. Лабораторная работа №4.**

**Процедуры и функции – элементы структуризации программ.**

Цель работы

1. Использование подпрограмм (процедур и функций) для ввода/вывода векторов и матриц.

2. Использование подпрограмм (процедур и функций) для вставки и удаления элементов массива.

**Пример первого задания.**

Удалить из массива последний (в том случае, если их несколько) максимальный элемент.

Для решения задачи необходимо:

* найти номер максимального элемента к;
* сдвинуть все элементы, начиная с к+1-го на один элемент влево;
* последнему элементу присвоить значение 0.

{$R+}

Program My;

Const

MaxN=10;la=-10;ha=21; {Размер массива и интервал значений}

Type

MyArray=Array[1..MaxN] Of Integer;

Var

A:MyArray;

k:Integer;

Procedure Init(t,v,w:Integer; Var X:MyArray);{Процедура формирования значений элементов массива}

Var

I:Integer;

Begin

Randomize;

For i:=1 To t Do

X[i]:=v+Integer(Random(w));

End;

Procedure Print(t:Integer;X:MyArray);{Вывод t первых элементов массива X}

Var

i:Integer;

Begin

For i:=l To t Do

Write(X[i]:5);

WriteLn;

End;

Function Max(t:Integer;X:MyArray):Integer;

{Поиск номера последнего максимального элемента}

Var

i,mx:Integer;

Begin

mx:=1;

For i: =2 To t Do

If X[i]>=X[Mx] Then Mx:=i;

Max:=Mx;

End;

Procedure Sz (t,q:Integer;Var X:MyArray);{Сдвиг элементов массива влево на одну позицию, начиная с элемента с номером q+1}

Var

i:Integer;

Begin

For i:=q To t-1 Do

X[i] :=X[i+1] ;

X[t]:=0;

End;

Begin

n:=MaxN;

Init(n,la,ha,A) ;

WriteLn ('Исходный массив ') ;

Print(n,A) ;

k:=Max(n,A);

Sz(n,k,A);

WriteLn(‘Массив после удаления элемента ') ;

Print(n-1,A) ;

ReadLn;

END.

**Пример второго задания.**

Даны 3матрицы (5\*4,4\*5,5\*5), содержащие целые компоненты. Требуется в каждой матрице определить минимальный по модулю элемент. Распечатать ту, в которой находится больший из минимальных по модулю элемент. Считать, что такой элемент один.

{$R+}

Const

MaxN=5; МахМ=5;

Type

TMyArray=Array[l..MaxN,l..MaxM] Of Integer;

Var

A,B,C:TMyArray;

Res:array[1..3] of Integer;

M,n,max:Integer;

Procedure Init(t,v:Integer; Var X:TMyArray);{Процедура формирования значений элементов массива}

Var

I,j:Integer;

Begin

Randomize;

For i:=l To t Do

For j:=1 To v Do

X[I,j]:=Integer(Random(50))-20;

End;

For i:=l To t Do

Procedure Print(t,v:Integer;X:TMyArray);{Вывод элементов массива }

Var

I,j:Integer;

Begin

For i:=l To t Do

Begin

For j:=1 To v Do

Write(X[I,j]:5);

WriteLn;

End;

End;

Function Min(t,v:Integer;X:TmyArray):Integer;{ Функция нахождения минимального по модулю элемента}

Var

i,j:Integer;

Begin

Min:=X[1,1]:

For i:=l To t Do

For j:=1 To v Do

If abs(X[I,j])<abs(min) then

min:=X[I,j];

End;

Begin{ 0сновная программа }

Init(5,4,A) ;

Init(4,5,B) ;

Init(5,5,C) ;

Writeln(‘Матрица А’); Print(5,4,A) ;

Writeln(‘Матрица B’); Print(4,5,B) ;

Writeln(‘Матрица C’); Print(5,5,C) ;

Res[1]:=Min(5,4,A);

Res[2]:=Min(4,5,B);

Res[3]:=Min(5,5,C);

Max:=Res[1];

For n:=2 to 3 do

If Res[n]>max then

begin

Max:=res[n];

M:=n

End;

Writeln(‘Матрица содержащая максимальный из минимальных по модулю’);

Case m of

1: Print(5,4,A) ;

2: Print(4,5,B) ;

3: Print(5,5,C) ;

End;

END.

**6. Лабораторная работа №5**

**Комбинированный и множественный типы данных.**

Цель работы

1. Использование записей при решении задач.
2. Использование множеств и операций над множествами.

**Пример первого задания.**

Для заданного количества (MaxN) ЭВМ определяется отношение стоимости к производительности. Производительность задается в миллионах операций в секунду отдельно для скалярных и векторных операций; стоимость - в долларах. Программа выбирает ЭВМ с минимальным отношением стоимости к производительности отдельно для скалярных и векторных операций и печатает характеристики.

Const

MaxN=5;

Type

Computer=record

Firma, namecomp:string[20];

Year:2000..2010;

Proizv:record

Scal, vect:Integer

End;

Price: Integer;

PriceToProizv:array[1..2] of real;

End;

Var

a: array[1..MaxN] of Computer;

I,j1,j2:Integer;

R1,r2:real;

Procedure In\_Data; { Процедура ввода данных }

Begin

Writeln(‘введите данные ЭВМ’);

With A[I],proizv do

Begin

Write(‘введите название фирмы’);

Readln(firma);

Write(‘введите название comp’);

Readln(namecomp);

Writeln(‘введите характеристики’);

Readln(Year, Proizv, Scal, vect, Price);

PriceToProizv[1]:= price/scal;

PriceToProizv[2]:= price/vect;

End;

End;

BEGIN { основная программа}

Writeln;

For i:=1 to MaxN do

In\_Data;

{ поиск ЭВМ с минимальным отношением стоимости к производительности отдельно для скалярных и векторных операций}

R1:=A[1].PriceToProizv[1]; j1:=1;{ для скалярных операций }

R2:=A[1].PriceToProizv[2]; j2:=2; { для векторных операций }

For i:=2 to MaxN do

With A[I] do

Begin

If PriceToProizv[1]<r1 then

Begin

R1:=PriceToProizv[1]; j1:=i;

End;

If PriceToProizv[2]<r2 then

Begin

R2:=PriceToProizv[2]; j2:=i;

End;

End;

{ Печать характеристик }

With A[j1],proizv do{ ЭВМ с min отношением ст/пр для скалярных операций}

Writeln(firma, namecomp,Year:5, Scal:8, vect:8, Price:8, PriceToProizv[1]:7:2, PriceToProizv[2]:7:2);

With A[j2],proizv do{ ЭВМ с min отношением ст/пр для векторных операций}

Writeln(firma, namecomp,Year:5, Scal:8, vect:8, Price:8, PriceToProizv[1]:7:2, PriceToProizv[2]:7:2);

END.

**Пример второго задания.**

Известен набор продуктов: хлеб, масло, сыр, молоко, имеющихся в ассортименте магазинов. В 3 магазина поставлены отдельные виды этих продуктов. Требуется построить множества А, В, С, которые содержат соответственно:

* продукты, имеющиеся одновременно во всех магазинах;
* продукты, имеющиеся, по крайней мере, в одном из магазинов;
* продукты, которых нет ни в одном из магазинов.

Const

N=3;

Type

Product=(bread,butter,cheese,milk); {продукты }

Assort=set of product; {ассортимент продуктов }

Shop=array[1..n] of assort; {наличие продуктов в магазинах }

Var

M1:shop;

X: Product;

A,b,c,xm1: assort;

I,j,v,iw:integer;

BEGIN

For I:=1 to n do

Begin

Xm1:=[];

Writeln(‘введите номера продуктов ‘,I:2, ‘-го магазина=‘);

Repeat

Read(iw);

Case iw of

1:x:=bread;

2:x:=butter;

3:x:=cheese ;

4:x:= milk;

else

write(‘номера продуктов 1..4’);

end; {case}

xm1:=xm1+[x]; {наличие продуктов в одном магазине }

until eoln;

m1[1]:=xm1

end; {конец цикла по магазинам }

{получение множеств А,В,С }

A:=m1[1];

B:=[];

For I:=1 to 3 do

Begin

A:=A\*m1[1]; ];{одновременно во всех }

B:=B+ m1[1{хотя бы в одном }

C:=[ bread.. milk]-B; {нет ни в одном }

End;

For I:=1 to 3 do

Begin

Case i of

1: Writeln(‘продукты, имеющиеся во всех магазинах ‘);

2: Writeln(‘ассортимент продуктов ‘);

3: Writeln(‘продукты, которых нет ни в одном магазине ‘);

end;

For x:=bread to milk do

If x in a then

Case x of

Bread: Write(‘хлеб ‘);

Butter: Write(‘масло ‘);

Cheese: Write(‘сыр ‘);

Milk: Write(‘молоко ‘);

end; {case}

if I:=1 then a:=b

else

a:=c;

writeln

end

END.

Рекомендуемая литература:

1. В.В. Попов Паскаль и Дельфи . Учебный курс. - СПб.: Питер, 2005.
2. Г.Г. Рапаков, С.Ю. Ржеуцкая Turbo Pascal для студентов и школьников СПб.: БХВ-Петербург, 2005.
3. А.Б.Ставровский Турбо Паскаль 7.0. Учебник.- К.: Торгово-издательское бюро BHV, 2000.
4. С.М. Окулов Основы программирования – М.: ЮНИМЕДИАСТАЙЛ, 2002

**Основная литература:**

1. Белов В.В. Программирование в DELPHI: процедурное, объектно-ориентированное, визуальное [Электронный ресурс]: учебное пособие / В.В. Белов, В.И. Чистякова. — Электрон. дан. — М.: Горячая линия-Телеком, 2014. — 240 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1\_id=64091

2. Иванова Г.С. Средства процедурного программирования Microsoft Visual C ++ 2008 [Электронный ресурс]: учебное пособие / Г.С. Иванова, Т.Н. Ничушкина, Р.С. Самарев. — Электрон. дан. — М.: МГТУ им. Н.Э. Баумана (Московский государственный технический университет имени Н.Э. Баумана), 2012. — 140 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1\_id=52414

3. Богонин М.Б. Языки и системы программирования. Процедурные языки программирования на примерах Pascal и Си: учебное пособие [Электронный ресурс]: учебное пособие. — Электрон. дан. — Пенза: ПензГТУ (Пензенский государственный технологический университет), 2012. — 212 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1\_id=62777.

4. Богонин М.Б. Языки и системы программирования. Процедурные языки программирования на примерах Pascal и Си: учебно-методическая разработка для лабораторного практикума [Электронный ресурс]: учебно-методическое пособие / М.Б. Богонин, О.Е. Чуфистов, Е.Я. Хижняк. — Электрон. дан. — Пенза: ПензГТУ (Пензенский государственный технологический университет), 2012. — 116 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\_id=62440](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D62440%26ts%3D1475752695%26uid%3D4228515801419147930&sign=abe4336d46853e0f42be7eb5e400af1a&keyno=1)

5. Соколова, Ю.С. Разработка приложений в среде Delphi. В 2 частях. Часть 1. Общие приемы программирования. [Электронный ресурс] : учебное пособие / Ю.С. Соколова, С.Ю. Жулева. — Электрон. дан. — М. : Горячая линия-Телеком, 2013. — 142 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\_id=5196](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D5196%26ts%3D1475752695%26uid%3D4228515801419147930&sign=d9fcb836fe2b8aa42d02cd1e524421ab&keyno=1)

6. Медведик, В.И. Практика программирования на языке Паскаль (задачи и решения) [Электронный ресурс] : учебное пособие. — Электрон. дан. — М. : ДМК Пресс, 2013. — 590 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\_id=58700](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D58700%26ts%3D1475752695%26uid%3D4228515801419147930&sign=45f5d6f2e90610f27edda88519d0d322&keyno=1)

7. Дмитриев, Н.Н. Моделирование инженерных задач на языке программирования Free Pascal в среде Lazarus: учебное пособие для вузов [Электронный ресурс] : учебное пособие / Н.Н. Дмитриев, В.Ю. Сахаров. — Электрон. дан. — СПб. : БГТУ "Военмех" им. Д.Ф. Устинова (Балтийский государственный технический университет «Военмех» имени Д.Ф. Устинова), 2012. — 61 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\_id=63711](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D63711%26ts%3D1475752695%26uid%3D4228515801419147930&sign=7a18209c532cb510f034551c7e50b65a&keyno=1)

**Дополнительная литература:**

1. Москвитина, О.А. Сборник примеров и задач по программированию [Электронный ресурс] : учебное пособие / О.А. Москвитина, В.С. Новичков, А.Н. Пылькин. — Электрон. дан. — М. : Горячая линия-Телеком, 2014. — 245 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\_id=64090](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D64090%26ts%3D1475752695%26uid%3D4228515801419147930&sign=f4c5df67753eed7e93395144b72c761a&keyno=1)

2. Окулов, С.М. Задачи по программированию [Электронный ресурс] : / С.М. Окулов, Т.В. Ашихмина, Н.А. Бушмелева. — Электрон. дан. — М. : "Лаборатория знаний" (ранее "БИНОМ. Лаборатория знаний"), 2014. — 825 с. — Режим доступа:[http://e.lanbook.com/books/element.php?pl1\_id=66115](https://clck.yandex.ru/redir/dv/*data=url%3Dhttp%253A%252F%252Fe.lanbook.com%252Fbooks%252Felement.php%253Fpl1_id%253D66115%26ts%3D1475752695%26uid%3D4228515801419147930&sign=8b7f171dd73652a3a25bc3bcc3cec46b&keyno=1)

3. Окулов, С.М. Основы программирования [Электронный ресурс] : . — Электрон. дан. — М. : "Лаборатория знаний" (ранее "БИНОМ. Лаборатория знаний"), 2015. — 339 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1\_id=66119