Уважаемые студенты групп!

Вашему вниманию представлена лекция на тему «РАБОТА С ЗАПИСЯМИ»

Лекция рассчитана на 8 часов

Задание

1. Прочитать внимательно лекцию.
2. Законспектировать лекцию в рабочую тетрадь не менее 3-5 страницы рукописного текста. В конспекте лекции обязательно должно быть приведены примеры.
3. Решить приведенные в лекции в контрольных вопросах задачи.
4. Дата предоставления фотоотчета лекции до 22.05.2023.

С уважением Ганзенко Ирина Владимировна

!!! Если возникнут вопросы обращаться по телефону 0506324562(вайбер), 0721134803, +79591134803 (вацап, телеграм)

disobuch.ganzenko2020@mail.ru

РАБОТА С ЗАПИСЯМИ

План

1. Тип «Запись» (Record) и оператор присоединения With
2. Записи с вариантами
3. Контрольные вопросы:

В информационно-поисковых системах приходится хранить и обрабатывать большие объемы данных. При решении научно-технических и экономических задач обработки совокупностей большого количества значений используются массивы. Но при работе с массивами основное ограничение заключается в том, что все элементы массива должны иметь один и тот же тип данных. Иногда для решения задач, в которых возникает необходимость хранить и обрабатывать совокупности данных различного типа, используются отдельные массивы для каждого типа данных, а для установления соответствия между ними вводятся соответствующие индексы.

Реальные данные об объектах часто описываются величинами разных типов. Например, товар на складе описывается следующими величинами: наименование, количество, цена, наличие сертификата качества и т. д. В этом примере наименование - величина типа string количество - integer, цена - real, наличие сертификата качества можно описать величиной типа Boolean. Для

записи комбинации объектов разных типов в Pascal применяется комбинированный тип данных - запись.

Запись представляет собой наиболее общий и гибкий структурированный тип данных, так как она может быть образована из разнотипных компонентов и в ней явным образом выражена связь между элементами данных, характеризующими реальный объект.

1. Тип «Запись» (Record) и оператор присоединения With

Запись - это структурированный тип данных, состоящий из фиксированного числа компонентов одного или нескольких типов. Определение типа записи начинается идентификатором record и заканчивается зарезервированным словом end. Между ними располагается список компонентов, называемых полями, с указанием идентификаторов полей и типа каждого поля. Тип «запись» определяется конструкцией

```
RECORD
Поле1: ТипПоля1;
Поле2: ТипПоля2;
ПолеМ : ТипПоляМ
END;
```

Если тип нескольких полей совпадает, то имена полей могут быть просто перечислены, например:

```
Type
PointRecType = record x, y: Integer end;
Пример.
Type
Car=Record
    Number:     integer;      {Номер}
    Marka:      string[20];   {Марка автомобиля}
    FIO:        string[40];   {Фамилия, инициалы владельца}
    Address:    string[60];   {Адрес владельца}
end;
Var
M, V: Char;
```

В данном примере запись Car содержит четыре компонента: номер, название марки машины, фамилию владельца и его адрес. Доступ к полям записи осуществляется через переменную типа «запись». В нашем случае это переменные M и V типа Car.

Идентификатор поля должен быть уникален только в пределах записи, однако во избежание ошибок лучше делать его уникальным в пределах всей программы. Порядок описания полей в определении записи задает их

порядок хранения в памяти. Объем памяти, необходимый для записи, складывается из длин полей.

Значения полей записи могут использоваться в выражениях. Имена отдельных полей не применяются по аналогии с идентификаторами переменных, поскольку может быть несколько записей одинакового типа. Обращение к значению поля осуществляется с помощью идентификатора переменной и идентификатора поля, которые разделены точкой. Такая комбинация называется составным именем.

Например, чтобы получить доступ к полям записи Car надо написать:

M.Number, M.Marka, M.FIO, M.Address.

Составное имя можно использовать везде, где допустимо применение типа поля. Для присваивания полям значений используется оператор присваивания.

Пример.

```
M.Number:=1678;
M.Marka:= ‘Газ_24’;
M.FIO:=’Иванов И. И.’
M.Address:=’ул. Пушкина 12-31’;
```

Составные имена можно использовать, в частности, в операторах ввода-вывода:

```
Read(M.Number, M.Marka, M.FIO, M.Address);
Write(M.Number:4, M.Marka:7, M.FIO:12, M.Address:25);
```

Допускается применение оператора присваивания, и к записям в целом, если они имеют одинаковый тип. Например, V := M;

После выполнения этого оператора значения полей записи V станут равными значениям соответствующих полей записи M.

В ряде задач удобно пользоваться массивами из записей. Их можно описать следующим образом:

```
Type
Person= record
FIO: string[20];
Age : 1..99;
Prof: string[30];
End;
Var
List: array[1..50] of Person;
```

Обращение к полям записи имеет несколько громоздкий вид, что особенно неудобно при использовании мнемонических идентификаторов длиной более пяти символов. Для решения этой проблемы в языке Pascal предназначен оператор with, который имеет следующий формат

```
with ИмяПеременной_Записи DO Оператор;
```

Указав один раз переменную типа запись в операторе with, можно работать с именами полей как с обычными переменными, т. е. без указания перед идентификатором поля имени переменной, определяющей запись.

Пример: Присвоить значения полям записи Car с помощью оператора with

```
with M do
begin
Number:=1678;
Marka:= ‘Газ_24’;
FIO:=’Петров В. И.’
Address:=’ул. Остужева 2-31’;
end;
```

Внутри области действия оператора with могут указываться и переменные, не имеющие отношения к записи. Но в этом случае надо следить, чтобы они не совпадали по написанию с полями записи.

```
    PROGRAM MAIN;
    var
    X, Y: Integer;
    RecXY: record X, Y: Integer end;
Begin
        X: =10; Y: =20;             { значения переменных X и Y }
        with RecXY do begin         { работаем с записью RecXY }
         X: = 3.14 * X;             {Где какой X и Y? }
        Y: = 3.14 * Y
end; {with}
End.
```

Действия внутри оператора with проводятся только над полями записи RecXY. Чтобы сохранить оператор with и «развязать» имена X и Y, надо к переменным X и Y приписать так называемый квалификатор - имя программы или модуля (UNIT), в которой они объявлены (для этого программа должна иметь заголовок). Так, оператор присоединения можно исправить следующим образом:

```
with RecXY do begin
 X: = 3.14 * Main.X;
```

```
Y: = 3.14 * Main.Y
end;
```

и проблема исчезнет.

В Pascal допускает вложение записей (т.е. поле записи само является записью), соответственно оператор with тоже может быть вложенным.

```
with RV1 do
with RV2 do
with RVn do…
```

что эквивалентно конструкции

```
with RV1, RV2, …, RVn do...
```

Уровень вложенности записей не должен превышать 9.

Пример 1. Создать программу, для решения следующей задачи: Пусть дана ведомость абитуриентов, сдавших вступительные экзамены в институт. определить фамилии абитуриентов, начинающиеся с буквы А, и набравших средний балл не менее 4. Вывести на печать фамилию абитуриента и его средний балл.

Программа выглядит следующим образом:

```
Program Abt;
Uses CRT;
const M=15;
type T=record
 Fio :string[10];
 Ocn :array[1..3] of integer;
 end;
var
Spisok :T;
Kol :integer;
N :integer;
Bal :real;
I :integer;
Begin
clrscr;
Kol:=0;
Write('Введите количество абитуриентов :');
readln(N);
For i:=1 to N do
with Spisok do
begin
write('Введите ФИО :');readln(Fio);
```

```
write('Введите Оценку 1 :');readln(Ocn[1]);
write('Введите Оценку 2 :');readln(Ocn[2]);
write('Введите Оценку 3 :');readln(Ocn[3]);
Bal:=(Ocn[1]+Ocn[2]+Ocn[3])/3;
if (Bal>=4.0) and(Fio[1]='A') then begin kol:=kol+1;
Writeln('Абитуриент ',Fio:10,' средний балл ',bal:6:2);end;
writeln;
end;
if kol=0 then writeln('Абитуриентов, отвечающих данным условиям нет');
readln;
end.
```

Результат работы:

```
Введите количество абитуриентов :3
Введите ФИО :Иванов
Введите Оценку 1 :4
Введите Оценку 2 :5
Введите Оценку 3 :4
Введите ФИО :Антонов
Введите Оценку 1 :4
Введите Оценку 2 :3
Введите Оценку 3 :4
Введите ФИО :Акунин
Введите Оценку 1 :4
Введите Оценку 2 :5
Введите Оценку 3 :5
Абитуриент Акунин средний балл 4.67
```

2. Записи с вариантами

Записи, представленные выше, имеют строго определенную структуру. В не которых случаях это серьезно ограничивает возможности их применения. Поэтому в Pascal имеется возможность задать тип записи, содержащий произвольное число вариантов структуры. Такие записи называются записями с вариантами. Записи с вариантами обеспечивают средства объединения записей, которые похожи, но не идентичны по форме. Они состоят из фиксированной и вариантной частей.

Использование фиксированной части аналогично описанному ранее. Вариантная часть формируется с помощью оператора case . Он задает особое поле записи поле признака, которое определяет, какой из вариантов в данный момент будет активизирован. Значением признака в каждый текущий момент выполнения программы должна быть одна из расположенных следом констант. Константа, служащая признаком, задает вариант записи и называется константой выбора. Формат:

```
Type
Rec = record
case <поле признака>: <имя типа> of
<константа выбора1> : (поле.... :тип):
…
<константа выборан > : (поле. . . . :тип)
end;
```

Компоненты каждого варианта (идентификаторы полей и их типы) заключаются в круглые скобки. У части case нет отдельного end, как этого следовало бы ожидать по аналогии с оператором case. Одно слово end заканчивает всю конструкцию записи с вариантами. Необходимо отметить, что количество полей каждого из вариантов не ограничено.

Объем памяти, необходимый для записи с вариантами, складывается из длин полей фиксированной части и максимального по длине поля переменной части.

Пример:

```
Type
Rec = record
Number: byte;
Code: integer;
case Flag: Boolean of
True : (Price1: integer);
False: (Price2: real);
End;
Var PRec: Rec;
```

Поля Number и Code расположены в фиксированной части записи, они доступны в программе в любой момент независимо от значения поля признака. Поле Price1 может использоваться только в том случае, если значение поля признака Flag равно True. Поле Price2 доступно в противоположном случае, т. е. если значение Flag равно False.

При использовании записей с вариантами необходимо придерживаться следующих правил:

• все имена полей должны отличаться друг от друга, по крайней мере, одним символом, даже если они встречаются в разных вариантах;

• запись может иметь только одну вариантную часть, причем она должна размещаться в конце записи;

• если поле, соответствующее какой - либо метке, является пустым, то оно записывается следующим образом: <метка>: ().

Пример 2. Напишем программу, позволяющую вычислять площади различных геометрических фигур с применением одной функции.

Программа выглядит следующим образом:

```
Program SF;
Uses Crt;
Type TF=(rect,circle);
 figure=record
 case art : TF of
 rect :(L1,L2:real);
 circle:(R:real);
 end;
Var x:figure;
p:string[6];
pr:char;
Begin
ClrScr;
repeat
write('Введите вид фигуры (rect, circle) '); readln(p);
with x do begin
if p='rect' then art:=rect;
if p='circle' then art:=circle;
case art of
rect: begin
 write('Введите длину прямоугольника= '); readln(L1);
 write('Введите ширину прямоугольника= '); readln(L2);
 writeln('Площадь равна=', L1*L2:8:3);
 end;
circle: begin
 write('Введите радиус окружности= ');readln(R);
 writeln('Площадь равна=', Pi*R*R:8:3);
 end;
end; {case}
end;
write('Продолжить расчет? (Y-Да,N- Нет) ');readln(pr);
until pr='N';
readln;
End.
```

Результат работы:

```
Введите вид фигуры (rect, circle) rect
Введите длину прямоугольника= 4
Введите ширину прямоугольника= 5
Площадь равна= 20.000
Продолжить расчет? (Y-Да,N- Нет) Y
Введите вид фигуры (rect, circle) circle
Введите радиус окружности= 6
Площадь равна= 113.097
```

Продолжить расчет? (Y-Да,N- Нет) N

3. Контрольные вопросы:

1. К каким типам данных относится запись?
2. Что такое поле записи?
3. Чем запись отличается от массива?
4. могут совпадать имена полей? Объясните ответ.
5. Какого типа могут быть поля записи?
6. Как в общем виде объявляется тип записи?
7. Как задается поле записи?
8. Как осуществляется ввод-вывод записей?
9. Что такое оператор присоединения? Который его формат?
10. Как получить доступ к полю записи в массиве из записей?
11. Как получить доступ к элементу массива, является полем записи?
12. Что такое фиксированные записи? Вариантные записи?
13. Из каких частей состоит запись с вариантами?
14. Когда применяется запись с вариантами?