Ответьте на вопросы:

1. **Чем отличаются статические и динамические величины?**

Статические величины — это те, память под которые выделяется сразу при компиляции программы. Размер таких величин фиксируется заранее, и программа точно знает, сколько места потребуется для их хранения. Примером статической величины может служить обычный массив фиксированной длины.

var

arr: array[1..10] of integer;

Динамические величины — это те, которые создаются во время выполнения программы, и память под них выделяется динамически, из специальной области памяти. Динамическое выделение памяти позволяет программе изменять объем используемой памяти прямо во время работы. Примером динамического массива в Pascal может быть использование динамических массивов:

type

TIntArray = array of Integer;

var

dynArr: TIntArray;

begin

SetLength(dynArr, 10);

1. **Что такое указатель?**

Указатель — это переменная, которая хранит адрес памяти. Указатели в Паскале необходимы при работе с динамической памятью. Переменная типа «указатель» в качестве своего значения содержит адрес участка динамической памяти, с которой связан этот указатель.

1. **Какие виды указателей вам известны?**

В Pascal различают следующие виды указателей:

* Простые указатели: указывают на определенный тип данных. Например, ^Integer — это указатель на целое число.
* Типизированные указатели: в отличие от простых указателей, они указывают на конкретные структуры данных. Например, можно создать типизированный указатель на запись:

type

TPoint = record

X, Y: Integer;

end;

var

PtrToPoint: ^TPoint;

1. **Как определяется физический адрес переменной?**

Чтобы определить физический адрес переменной в Pascal, нужно воспользоваться оператором @ (адрес), который возвращает указатель на эту переменную:

var

i: Integer;

pi: ^Integer;

begin

pi := @i; // pi теперь содержит адрес переменной i

end;

1. **Как получить адрес указателя?**

Оператор @ позволяет получить адрес любой переменной, включая указатели. Если у вас есть указатель, вы можете получить его адрес следующим образом:

var

P: ^Integer;

PP: ^^Integer;

begin

New(P); // Выделяем память под целочисленную переменную

PP := @P; // PP теперь содержит адрес указателя P

end;

1. **Приведите примеры объявления указателей.**

Вот несколько примеров объявления указателей в Pascal:

var

IntPtr: ^Integer; // Указатель на целое число

CharPtr: ^Char; // Указатель на символ

StringPtr: ^String; // Указатель на строку

RecordPtr: ^TRecord; // Указатель на запись

1. **Как выделить память под динамическую переменную?**

Для выделения памяти под динамическую переменную в Pascal используется оператор New. Он выделяет блок памяти необходимого размера и возвращает указатель на начало этого блока. Пример:

var

P: ^Integer;

begin

New(P); // Выделяется память под одно целое число

P^ := 123; // Запись значения в выделенный блок памяти

end;

1. **Как освободить память от динамической переменной?**

Освобождение памяти происходит с использованием оператора Dispose. Этот оператор уничтожает указатель и освобождает выделенную ранее память. Пример:

var

P: ^Integer;

begin

New(P);

Dispose(P); // Освобождаем память, выделенную под P

end;

1. **Что в языке Pascal обозначает константа Nil?**

Константа Nil в Pascal обозначает пустой указатель, то есть указатель, который ничего не указывает. Nil часто используется для инициализации указателей перед их использованием или для проверки, был ли успешно выделен блок памяти. Пример:

var

P: ^Integer;

begin

P := nil; // Инициализация указателя пустым значением

end;

1. **В каком случае возможно присваивание указателей?**

Присваивать указатели можно, если они имеют одинаковый тип. Например, если два указателя указывают на одну и ту же структуру данных, то один указатель можно присвоить другому:

var

A, B: ^Integer;

begin

New(A);

B := A; //Теперь B указывает на тот же блок памяти,что и A

end;

Также допустимо присваивание указателям значения nil:

A := nil; // Обнуление указателя

Нетипизированные (Pointer) можно присваивать любым указателям, но требуется приведение типа.