

**УКАЗАТЕЛЬ** – это переменная, значением которой является адрес какого-то объекта в памяти компьютера (обычно, другой переменной).

Для каждой описанной в программе переменной выделяется область памяти, когда мы объявляем в программе, например, `float der`, то выделяется **постоянная (статическая) область памяти, которая будет занята этой переменной до конца работы программы.** **Указатели**, в частности, позволяют организовать **динамическое выделение и освобождение памяти для переменных в процессе выполнения программы.**

Но указатели используют и для работы с переменными в статической памяти.

С помощью указателей например можно:

- обрабатывать многомерные и одномерные массивы, строки, символы, структуры и массивы структур, в том числе, большого размера,
- динамически создавать новые переменные и структуры в процессе выполнения программы,
- обрабатывать связанные структуры: стеки, очереди, списки, деревья, сети,
- передавать функциям адреса фактических параметров,
- передавать функциям адреса функций в качестве параметров.

Разберемся с организацией памяти в ПЭВМ и в программах на Pascal и C.

# Адресация и распределение памяти ИНТЕРТ

## для программ Для IBM совместимых ПЭВМ

Оперативная память (ОП) ПЭВМ состоит из ячеек размером в 1 байт, каждая ячейка имеет собственный уникальный номер – адрес. Память разделяется на сегменты размером  $2^{16} = 65\,536$  байт или 64 Кб, что определяется архитектурой микропроцессоров 80x86 (16-разрядных). [32-х и 64-х-разрядная адресация поддерживается в операционных системах Windows и UNIX / Linux.]

**Адрес** каждого байта ОП формируется из двух слов (каждое размером по 16 бит – по 2 байта) – **адреса сегмента** и **смещения** (смещение указывает на сколько байт от начала сегмента сдвинут адрес конкретного байта).

**Адрес байта** формируется следующим образом: адрес сегмента смещается на 4-е двоичных разряда (бита) влево и к нему прибавляется смещение, таким образом физический адрес байта составляет 20 бит и с его помощью можно адресовать  $2^{20}$  байт (1 Мбайт):



Указатель занимает в памяти 4 байта (32 бита) и адресует первый байт данных, например, первый байт первого элемента массива.

Указатель занимает в памяти 4 байта (32 бита) хотя собственная длина указателя – 20 бит и он помещается в 3 байта, **НО** обычно память выделяется словами по 2 байта (16 бит) и выравнивается на границу слова – т.е под указатель выделяется 2 слова (а не 1,5).

# Адресация и распределение памяти И+ПРГ

Pascal

## для программ

C / C++

Для IBM совместимых ПЭВМ и MS DOS

Для кода программы, данных и переменных, тип которых описан в программе, **компилятор** выделяет фиксированную память, не изменяющуюся в процессе выполнения программы – **статическую память**. Если невозможно заранее определить объем данных, то надо перейти к использованию **динамической памяти**. Её выделение под данные и освобождение осуществляется **программистом**. Расположение **динамической памяти (heap, кучи)** в компьютере показано на схемах ниже. Обращение к динамическим переменным осуществляется через **указатели**.

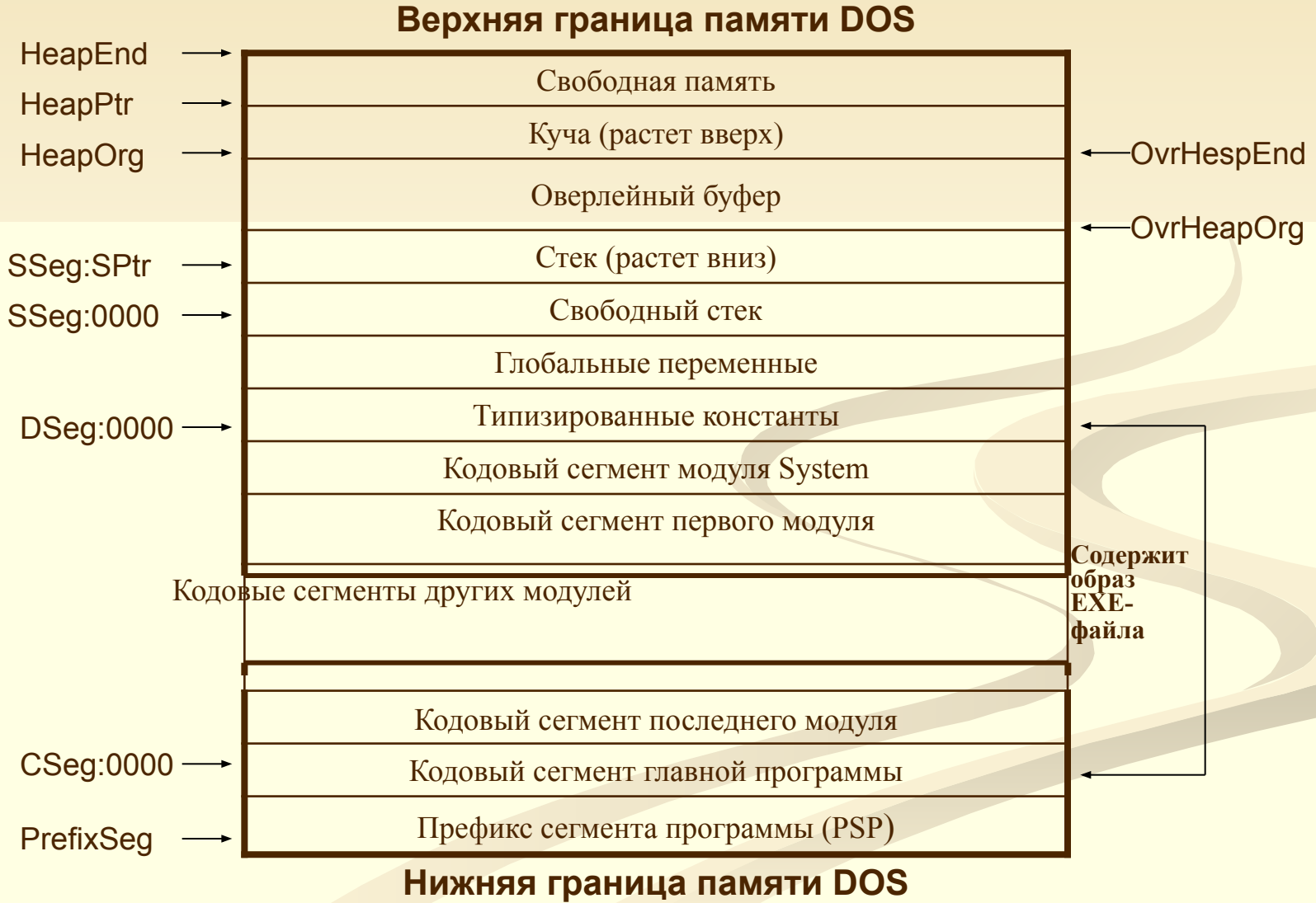
### Распределение оперативной памяти ПЭВМ для программ на Pascal и C.

Старшие адреса	Системная область	
	Область динамической памяти (heap, куча)	
	Стек	Область программы
	Данные	
	Код	
Младшие адреса	Системная область	

Старшие адреса	Системная область	
	Область стека	
	Область динамической памяти (heap, куча)	
	Область глобальных переменных	
	Область программы	
Младшие адреса	Системная область	

# Адресация и распределение памяти для программ

## Карта оперативной памяти программ на Borland Pascal



# Адресация и распределение памяти для программ

## Замечания к карте оперативной памяти для программ на Pascal

**Префикс сегмента программы (Program Segment Prefix - PSP)** -это 256-ти байтовая область, создаваемая DOS при загрузке программы. **Адрес сегмента PSP хранится в переменной PrefixSeg.**

Главная программа, и каждый модуль имеют свой **кодový сегмент**. Главная программа занимает **первый кодový сегмент**; кодóвые сегменты, которые следуют за ним, занимают **модули (в порядке, обратном тому, как они следовали в операторе uses)**, и последний кодový сегмент занимает **библиотека времени выполнения (модуль System)**. Размер одного кодóвого сегмента не может превышать 64К, но общий размер кода ограничен только имеющейся памятью.

**Сегмент данных (адресуемый через DSeg)** содержит все глобальные переменные и затем все типизованные константы. Регистр DS никогда не изменяется во время выполнения программы. Размер сегмента данных не может превышать 64К.

При запуске программы регистр **сегмента стека (SSeg)** и **указатель стека (SP)** устанавливаются так, что **SS:SP указывает на первый байт после сегмента стека**. Регистр SS никогда не изменяется во время выполнения программы, а SP может передвигаться вниз пока не достигнет конца сегмента. Размер стекового сегмента не может превышать 64К; размер по умолчанию - 16К, он может быть изменен директивой компилятора \$M.

Буфер оверлеев используется стандартным модулем Overlay для хранения оверлейного кода. Размер оверлейного буфера по умолчанию соответствует размеру наибольшего оверлея в программе; если в программе нет оверлеев, размер буфера оверлеев равен 0. Размер буфера оверлеев может быть увеличен с помощью вызова программы OvrSetBuf модуля Overlay; в этом случае размер кучи соответственно уменьшается, смещением вверх HeapOrg.

## Pascal

**УКАЗАТЕЛЬ** – это переменная, значением которой является **адрес** какого-то объекта в памяти компьютера (обычно, другой переменной).

В Pascal есть два вида указателей – **ТИПИЗИРОВАННЫЕ** – объявляются с использованием символа  $\wedge$  –

**Type** <тип\_указателя> =  
 $\wedge$ <имя\_типа\_переменной>;

**Type** pchar =  $\wedge$ char; **Var** pc : pchar;  
Можно объявить указатель прямо при описании переменной –

**Var** p1 :  $\wedge$ integer; p2 :  $\wedge$ real;  
[В Pascal указатели, как исключение, могут ссылаться на еще не объявленный тип данных],

и **НЕТИПИЗИРОВАННЫЕ** – объявляются с использованием стандартного типа данных **POINTER** –

**Var** <имя\_указателя> : **Pointer**;

**Var** p : pointer;

(их удобно использовать для данных, структура и тип которых изменяется в ходе работы программы).

Указатель в Pascal – переменная ссылочного типа.

## C / C++

Указатель в C не является отдельным типом данных, он всегда связан с каким-либо конкретным типом.

В C / C++ различают три вида указателей:

**на объект, на функцию и на void.**

Объявление указателя на объект:

<Характеристики\_типа\_данных> \*

<Характеристики\_указателя>

Обычно: <тип\_данных> \* <имя\_указателя>;

Например: **int \*a, \*c;**

Объявление указателя на void – применяется когда конкретный тип объекта, адрес которого требуется хранить, не определен:

Например: **extern void \*malloc(int);**

Объявление указателя на функцию:

<тип> (\*<имя>) (<список\_типов\_аргументов>);

Например: **int (\*fun)(double, double);**

Примеры:

**int \*pi;** // Указатель на целую переменную

**const int \*pci;** // Указатель на целую константу

**int \* const cp=&i;** // Указатель-константа на переменную

**const int \*const cpc=&ci;** // Указатель-константа на целую константу

# Элементы ЯПВУ. УКАЗАТЕЛИ.

И+ПРГ

Pascal

Операции с указателями

C / C++

- Адрес переменной –  $@W$  – возвращает адрес переменной  $W$ .
- Разадресация (разыменование) –  $\langle \text{имя\_указателя} \rangle^{\wedge}$  – это обращение к значению переменной, адрес которой хранится в указателе. Указатели стандартного типа (pointer) разыменовывать нельзя.

- Присваивание –  $\langle \text{имя\_указателя} \rangle := \langle \text{ссылочное выражение} \rangle$

Ссылочное выражение – обязательно того же типа, что и указатель, это:

- указатель (ссылочная переменная),
- пустая ссылка  $nil$  – нет конкретного объекта,
- ссылочная функция (результат – ссылка, указатель).

Например:  $\text{Var } p1, p2: ^{\wedge}\text{real}; p3: ^{\wedge}\text{integer}; pp : \text{pointer};$

Тогда –  $p1:=p2; p2:=pp;$  - допустимо, а  $p2:=p3;$  - запрещено.

- Сравнение –  $\langle \text{имя\_указателя}_1 \rangle =$  [или  $\langle \rangle$ ]  $\langle \text{имя\_указателя}_2 \rangle$  – сравнивать можно на равенство или неравенство.

- Получение адреса –  $\&\langle \text{имя\_переменной} \rangle$  – значение адреса операнда  $\langle \text{имя} \rangle$  в памяти. Например:  $n = \&p1;$

- Разыменование (разадресация, раскрытие ссылки) –  $*\langle \text{имя\_указателя} \rangle$  – значение объекта, адрес которого  $\langle \text{имя\_указателя} \rangle$ . Например:  $z = *n;$

- Присваивание –  $\langle \text{имя\_указателя}_1 \rangle = \langle \text{имя\_указателя}_2 \rangle$  - если оба операнда имеют один тип – простое присваивание, иначе – присваивание с преобразованием типа указателя – это:

- Приведение типов – два вида – с указателем типа  $\text{void}^*$  и без его использования.

$\text{void}^*$  в C разрешает неявное преобразование (в C++ – только явное, даже для  $\text{void}^*$ ). Преобразования всех остальных типов указателей должны быть явными, т.е. должна быть указана операция приведения типов.

Например: если  $x$  – имеет тип  $\text{int}$ , а  $*p$  –  $\text{double}$ , то надо записывать  $p = (\text{double } *) \&x.$

Но использовать приведение надо осторожно и обязательно к базовому типу указателя, а не объекта.

Pascal

C / C++

## Операции с указателями

Некоторые стандартные функции для работы с указателями:

- **addr (x) : pointer** - аналогично @,
- **seg (x) : word** – адрес сегмента для x,
- **ofs (x) : word** – смещение для x,
- **cseg : word** – значение регистра сегмента кода программы – CS,
- **dseg : word** – значение регистра сегмента данных программы – DS,
- **ptr (seg, ofs : word) : pointer** – по заданному сегменту и смещению формирует адрес типа pointer.

- **Арифметические операции (адресная арифметика)** – суммирование и вычитание – учитывают размер типа величины и применимы только к указателем одного типа:

- **сложение (вычитание) с константой** –  
 $p1 = p1 + 12; p2 = p2 - 5;$
- **увеличение (инкремент)** –  $p1++$ ; – при увеличении на 1 указатель p1 будет смещаться на величину типа данных и указывать на следующее значение. Например, в типе int\* – смещение на 2 байта –  $p1=200, p1++ = 202, p1-- = 198.$
- **уменьшение (декремент)** –  $p1--$ ; – аналогично инкременту.
- **вычитание двух указателей** – разность двух указателей – это разность их значений, деленная на размер типа в байтах. Так можно определять к-во объектов между адресами указателей. Например: в массиве разность указателей на 3-й и 6-й элементы равна 3.
- **Сравнение** - допустимы любые операции сравнения, обычно, они имеют смысл, когда ссылки на общий объект.



Pascal

C / C++

## ПРИМЕР

Вывод на печать адреса (значения указателя) в Pascal не поддерживается.

**Program Pointer;**

**Var**

**pa : ^word;**

**b: integer;**

**Begin**

**pa^ := 100;**

**b := 200;**

**pa := @b;**

**WriteLn('pa^=',pa^,', b=',b,');**

**End.**

Результат:

**pa^=200, b=200**

В языке C/C++ с указателями допустимы практически все операции.

```
#include <iostream.h>
```

```
void main ()
```

```
{
```

```
int *a, b;
```

```
*a=100; b=200'
```

```
cout<<"\na="<<a<<" , *a="<<*a<<" , b="<<b;
```

```
}
```

Результат:

**a=0x034c, \*a=100, b=200**