Массивы и указатели

Т.И. Комаров

НИЯУ МИФИ

2023

Уровни представления информации

- Интуитивный
- Логический (абстрактный)
- Конкретный (физический)

Логические структуры данных

- Массивы
- Строки
- Стеки, деки, очереди
- Таблицы
- Деревья
- Графы

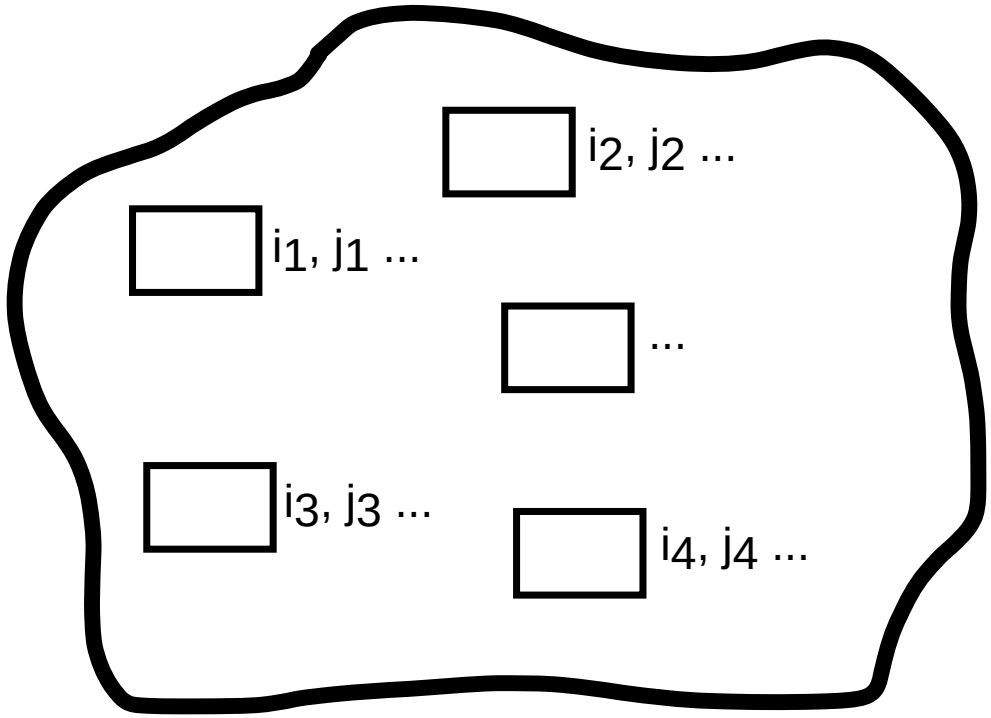
Важно

Для каждой логической структуры данных определён свой набор операций

Массив (логическая структура данных)

Определение

Массив — множество элементов, для которых упорядоченное множество целых чисел однозначно определяет позицию каждого элемента

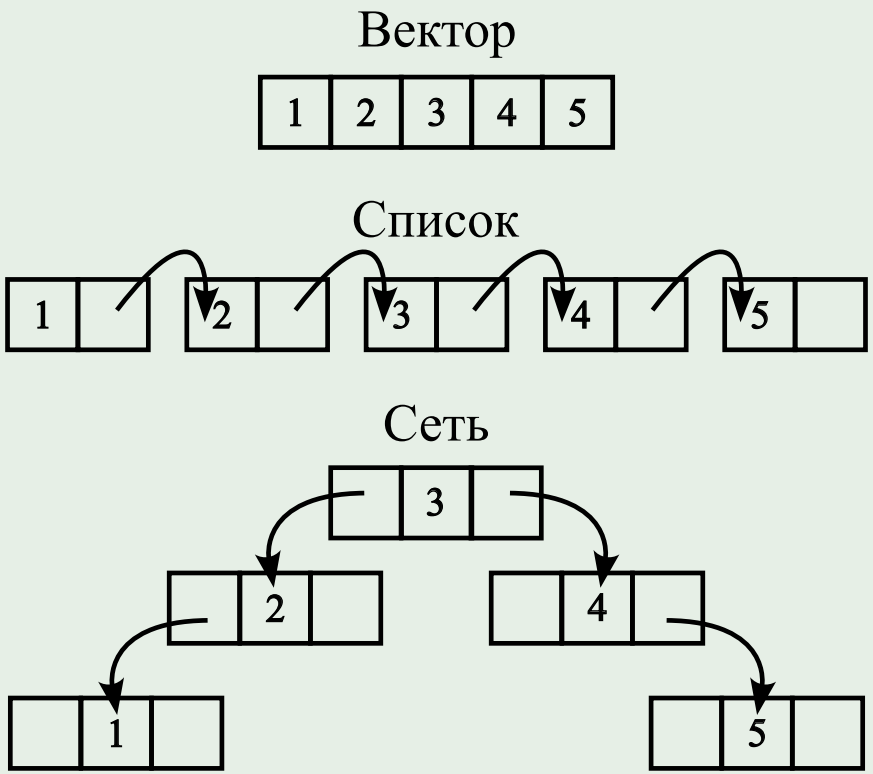


Физические структуры данных

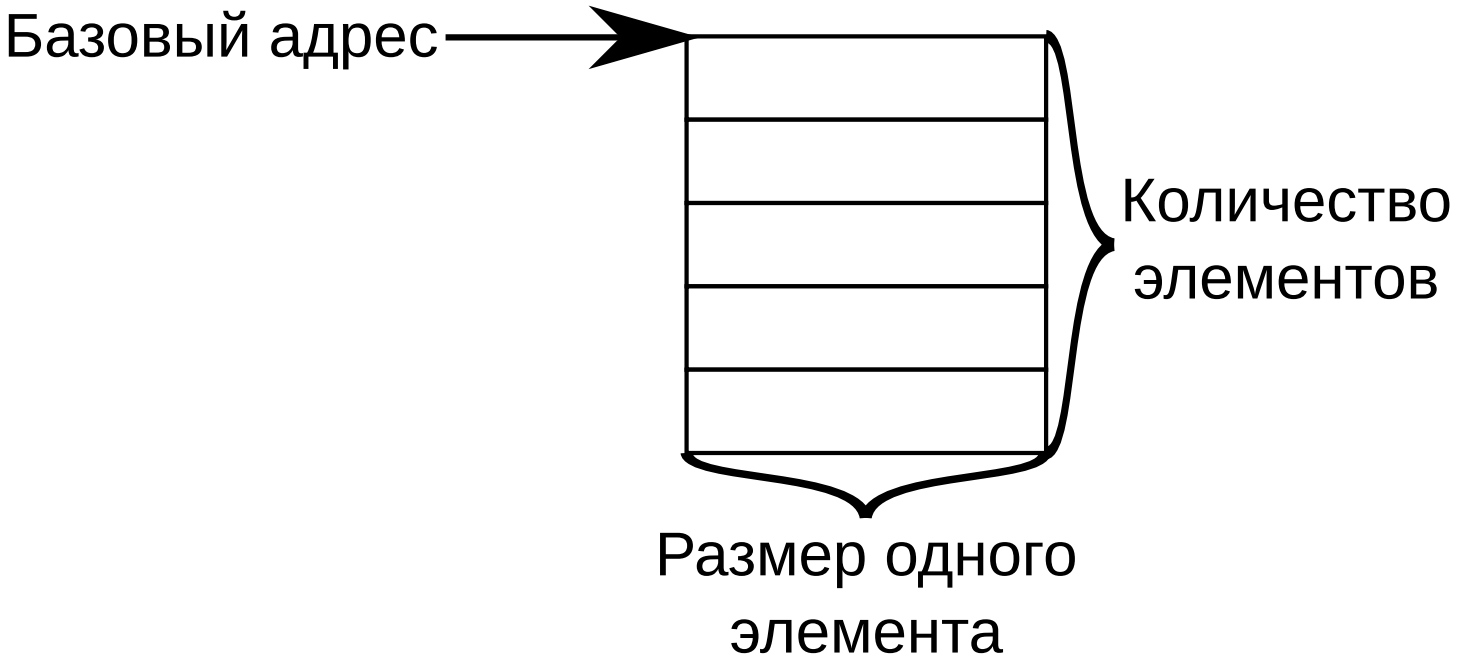
Представление данных в памяти компьютера:

- Вектор
- Список
- Сеть (не путать с компьютерными сетями!)

Примеры



Вектор (физическая структура данных)



Доступ к элементам вектора осуществляется по индексу:

$$\mathrm{addr_i} = \mathrm{base\_addr} + \mathrm{i} * \mathrm{item\_size} \quad (1)$$

где $\mathrm{addr_i}$ — адрес элемента вектора с индексом i, $\mathrm{base\_addr}$ — базовый адрес вектора, $\mathrm{item\_size}$ — размер одного элемента вектора

Пример представления данных на различных уровнях

Полином (многочлен) может быть представлен в виде функции следующего вида:

$$\mathrm{P(x)} = \mathrm{a_0} + \mathrm{a_1} * \mathrm{x} + \mathrm{a_2} * \mathrm{x^2} + \mathrm{a_3} * \mathrm{x^3} + ... + \mathrm{a_n} * \mathrm{x^n} \quad (2)$$

На логическом уровне полином может быть представлен в виде массива коэффициентов $\mathrm{a_i}$

На физическом уровне полином может быть представлен в виде вектора

Массивы I

Важно

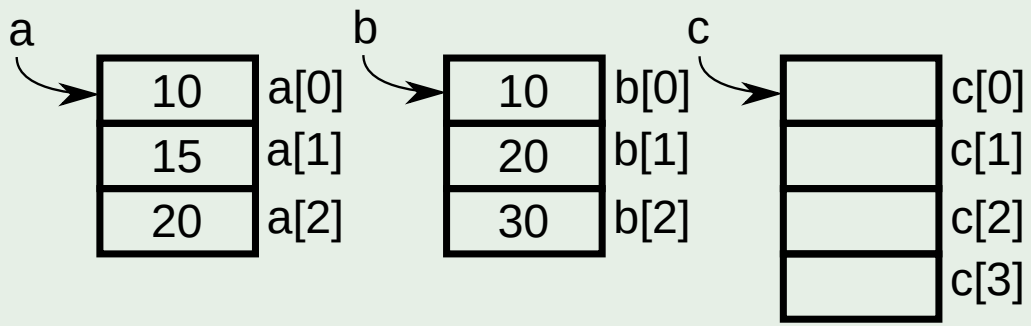
Массив в языке C соответствует конкретной структуре данных «вектор»

Определение массива:

```
тип имя[количество][={значение1, ...}], ...;
```

Примеры

```
int a[3] = {10, 15, 20}, b[] = {10, 20, 30};
int c[4];
```



Массивы II

Доступ к элементам массива: имя[выражение]

Примеры

```
int a[3] = {10, 15, 20};
for (int i = 0; i < 3; ++i) {
    a[i] += i;
    printf("a[%d] = %d\n", i, a[i]);
}
printf("a = %p\n", a);

// a[0] = 10
// a[1] = 16
// a[2] = 22
// a = 0x7ffcb5f7d4ec
```

Указатели I

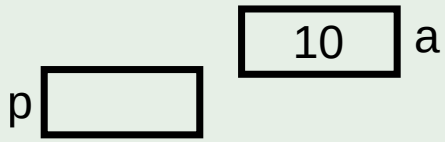
Определение

Указатель — переменная, содержащая адрес другого объекта

Примеры

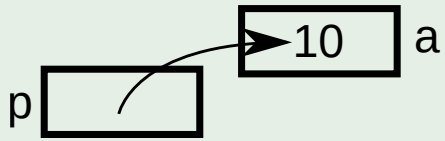
Определение:

```
int a = 10;
int *p;
```



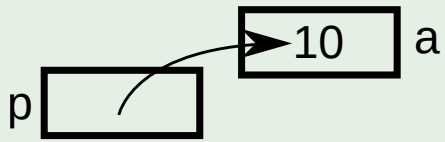
Инициализация:

```
int a = 10;
int *p = &a;
```



Присваивание:

```
int a = 10;
int *p;
p = &a;
```



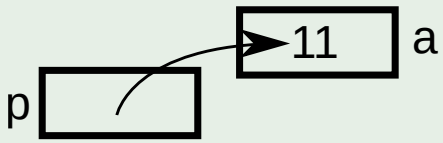
Указатели II

Доступ по указателю:

Примеры

```
int a = 10;
int *p = &a;
...
*p = 11;
printf("*p = %d\n", *p);

// p = 11
```



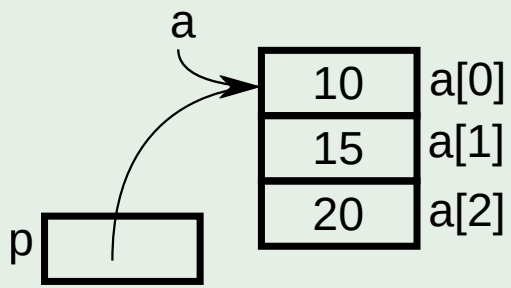
Указатели и массивы I

Примеры

```
int a[] = {10, 15, 20};
int *p = a; // int *p = &a[0];
int v;

// get first element
v = a[0], v = p[0];
v = *a, v = *p;

// assignment
p = a; // ok
a = ...; // error
```



Указатели и массивы II

Примеры

```
printf("a[0] = %d ---> p[0] = %d\n", a[0], p[0]);
// a[0] = 10 ---> p[0] = 10

printf("*a = %d ---> *p = %d\n", *a, *p);
// *a = 10 ---> *p = 10

printf("a = %p ---> p = %p\n", a, p);
// a = 0x7ffcb5f7d4ec ---> p = 0x7ffcb5f7d4ec
```

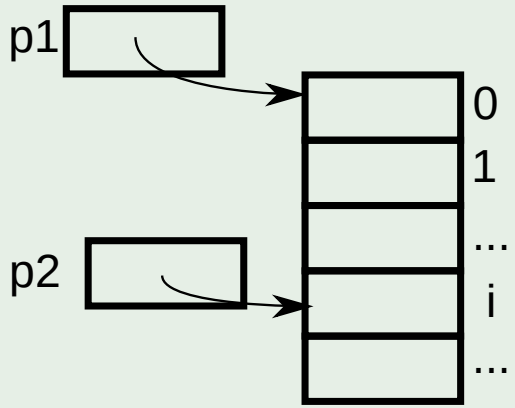
Адресная арифметика I

Примеры

```
<TYPE> *p1, *p2; // <TYPE> - any C type
int i;
p1 = ...;
p2 = p1 + i; // ok

...

++p1; // ok
p1 + i; // ok
p1 - i; // ok
p2 - p1; // ok
p1 + p2; // error
```



Адресная арифметика II

Важно

`a[i]` эквивалентно `*(a + i)`

Передача аргументов в функцию I

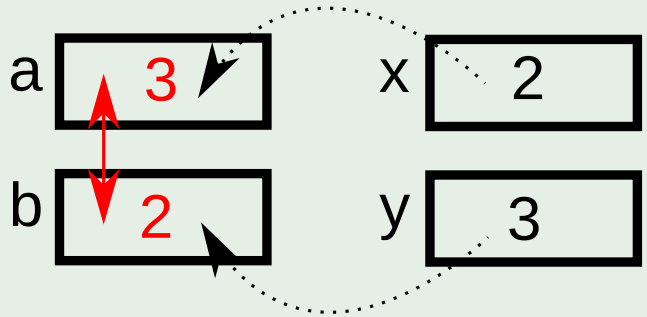
Примеры

```
void swap(int a, int b) {
    int tmp = a;
    a = b;
    b = tmp;
}

int main() {
    int x = 2, y = 3;
    swap(x, y);
    printf("x = %d, y = %d\n", x, y);
    return 0;
}
```

Передача аргументов в функцию II

Примеры




```
// x = 2, y = 3
```

Важно

Параметры в языке C передаются исключительно по значению! Вызываемая функция работает с копиями передаваемых параметров и не может изменять их оригинальные значения

Передача аргументов в функцию III

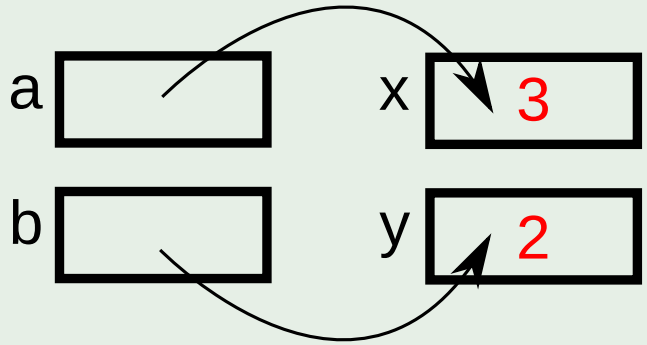
Примеры

```
void swap(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
}

int main() {
    int x = 2, y = 3;
    swap(&x, &y);
    printf("x = %d, y = %d\n", x, y);
    return 0;
}
```

Передача аргументов в функцию IV

Примеры




```
// x = 3, y = 2
```

Вывод

Для возможности изменения значения внешней по отношению к функции переменной необходимо передать в неё указатель на данную переменную

Передача массива в функцию в качестве аргумента

```
int main() {
    <TYPE> a[N];
    ...
    f(a, N);
    ...
    return 0;
}
```

```
... f(<TYPE> p[], int n) {
    ...
}
```

или

```
... f(<TYPE> *p, int n) {
    ...
}
```