

Материалы презентации предназначены для размещения только для использования студентами кафедры «Компьютерные системы и технологии» НИЯУ МИФИ дневного отделения, изучающими курс «Программирование (Алгоритмы и структуры данных)».

Публикация (размещение) данных материалов полностью или частично в электронном или печатном виде в любых других открытых или закрытых изданиях (ресурсах), а также использование их для целей, не связанных с учебным процессом в рамках курса «Программирование (Алгоритмы и структуры данных)» кафедры «КСиТ» НИЯУ МИФИ, без письменного разрешения автора запрещена.

С2. Массивы и указатели

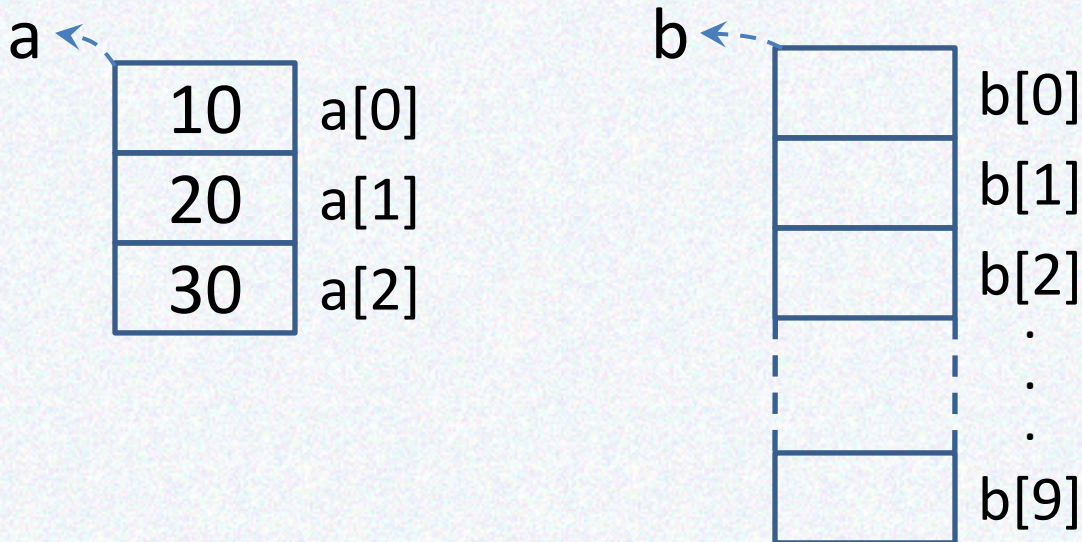
Массивы

Определение массива

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

Пример:

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

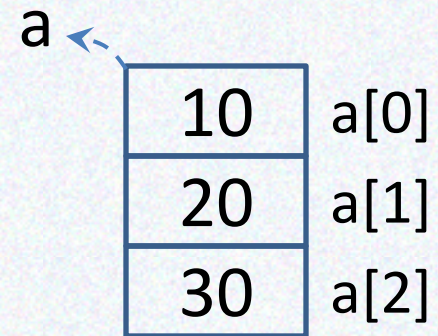


Массивы: использование

Доступ к элементу массива: *имя*[*выражение*]

Пример:

```
int a[3] = {10, 20, 30};  
int i;  
for(i = 0; i < 3; ++i)  
    printf("a[%d] = %d\n", i, a[i]);  
printf("a = %d (%x)\n", a, a);
```

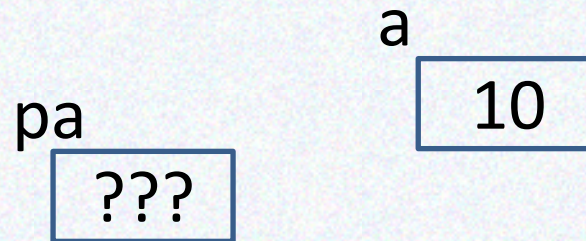


```
a[0] = 10  
a[1] = 20  
a[2] = 30  
a = 1245016 (12ff58)
```


Указатели: определение

```
int a = 10;
```

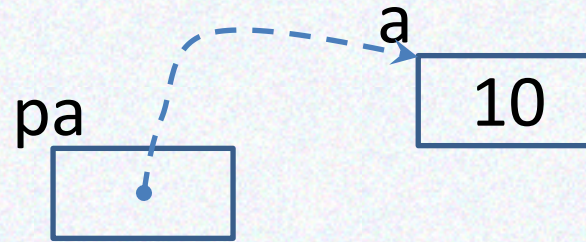
```
int *pa;
```



Операции с указателями

```
int a = 10;
```

```
int *pa = &a;
```

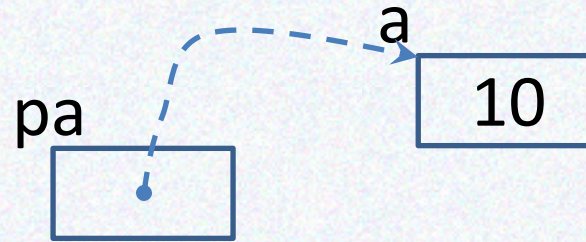


а) инициализация

Операции с указателями

```
int a = 10;
```

```
int *pa;
```



а) инициализация

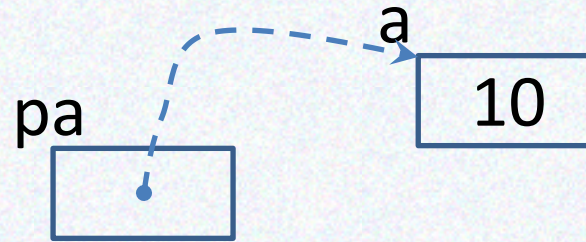
б) присваивание

```
pa = &a;
```

Операции с указателями

```
int a = 10;
```

```
int *pa;
```



а) инициализация

б) присваивание

```
pa = &a;
```

в) доступ по указателю

```
*pa
```

```
printf("*pa = %d\n", *pa);
```

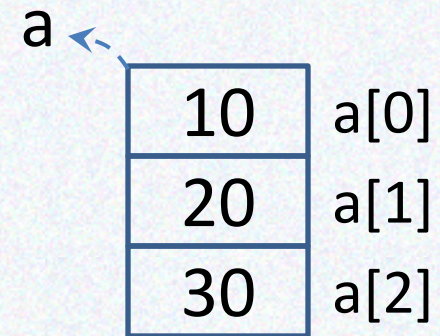
```
*pa = 10
```


Указатели и массивы

```
int a[3] = {10, 20, 30};
```

```
int *p;
```

p ???

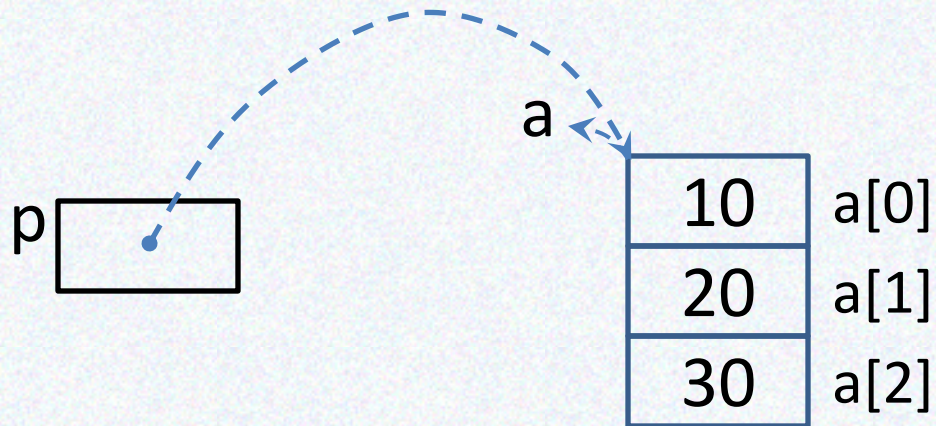


Указатели и массивы

```
int a[3] = {10, 20, 30};
```

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

Инициализация:

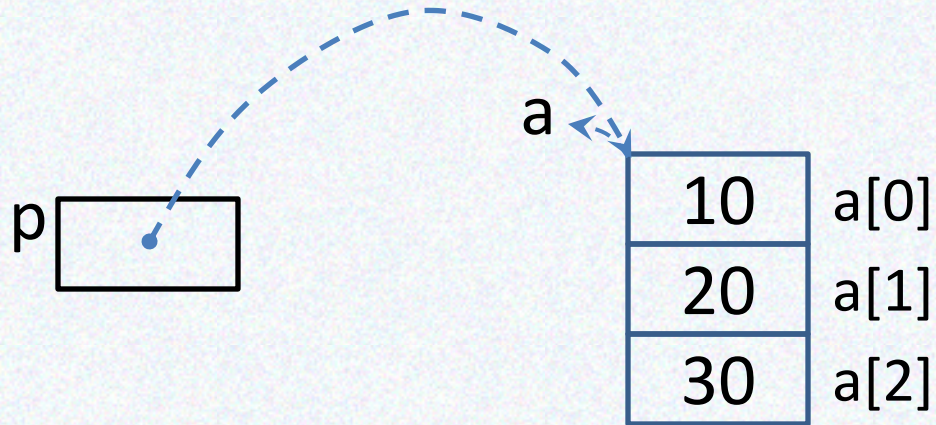


Указатели и массивы

```
int a[3] = {10, 20, 30};
```

```
int *p;
```

Инициализация:



Присваивание:

```
p = a;
```

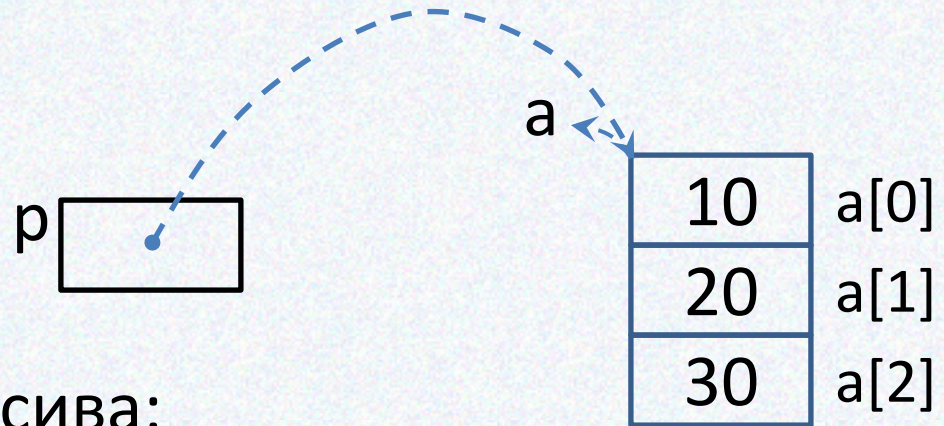
или

```
p = &a[0];
```

С2.10 Указатели и массивы: использование

```
int a[3] = {10, 20, 30};
```

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



Доступ к элементу массива:

a[0]

p[0]

*p

*a

p = a

~~a = ...~~

C2.11 Указатели и массивы: примеры

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

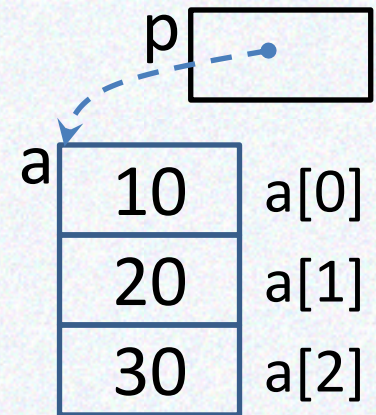
```
a[0] = 10 ---> p[0] = 10
```

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

```
*p = 10 ---> *a = 10
```

```
printf("a = %x ---> p = %x\n", a, p);
```

```
a = 12ff58 ---> p = 12ff58
```



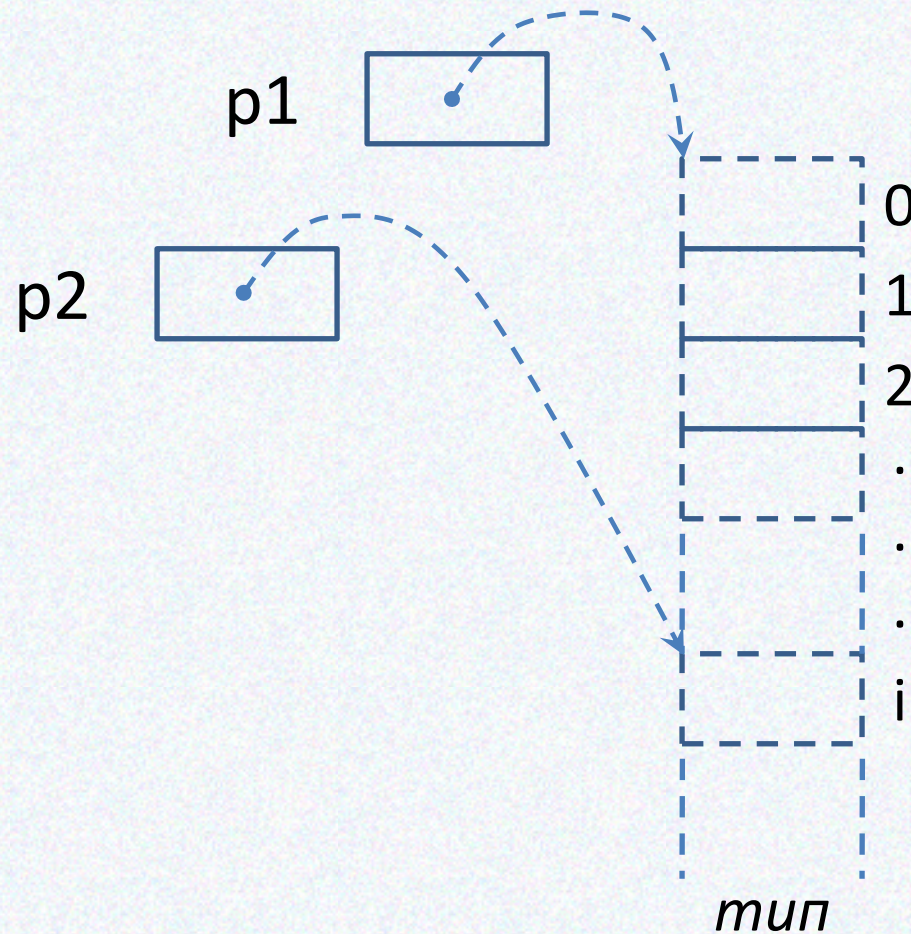
Адресная арифметика

тип *p1, *p2;

int i;

p1 = ...;

p2 = p1 + i;



Адресная арифметика

тип *p1, *p2;

int i;

p1 = ...;

p2 = p1 + i;

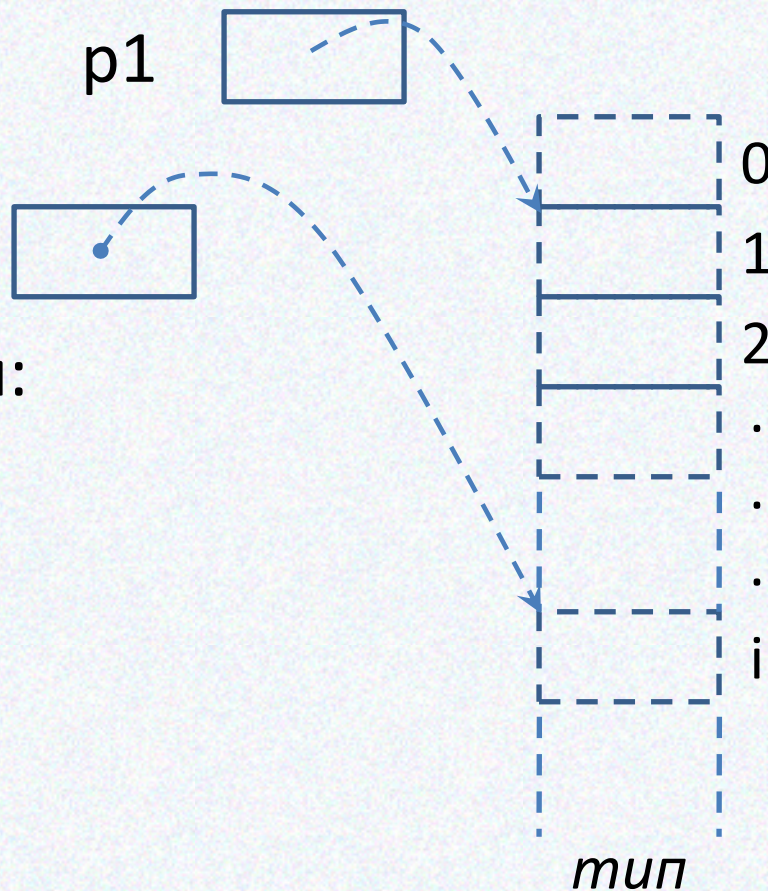
Изменение указателя:

++ p1;

Можно: $p1 \pm i$

Допускается: $p2 - p1$

Нельзя! ~~$p1 + p2$~~



Сравнение указателей

тип *p1, *p2;

Определены всегда:

p1 == p2 p1 != p2

Допускается:

p1 < p2 p1 > p2

Передача аргументов


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

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

x=2, y=3

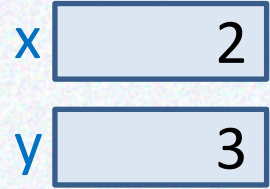
Передача аргументов

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



The diagram shows two empty rectangular boxes, one for parameter 'a' and one for parameter 'b', positioned to the right of the function signature and the first line of the function body.

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



The diagram shows two light blue rectangular boxes. The top box is labeled 'x' and contains the number '2'. The bottom box is labeled 'y' and contains the number '3'.

Передача аргументов

```
void swap(int a, int b)
```


```
{
```


```
    int t = a;
```

```
    a = b;
```

```
    b = t;
```

```
}
```

a 

b 

```
int main()
```

```
{
```

```
    int x = 2, y = 3;
```

```
    swap(x, y);
```

```
    printf("x=%d, y=%d\n", x, y);
```

```
    return 0;
```


```
}
```

x 

y 

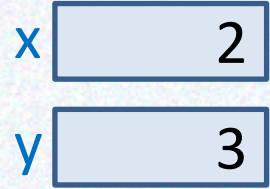
Передача аргументов

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



The diagram shows two variables, 'a' and 'b', each represented by a light blue box with a dark blue border. Variable 'a' contains the value 3, and variable 'b' contains the value 2. The boxes are positioned to the right of the 'swap' function code, corresponding to the parameters 'a' and 'b'.

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



The diagram shows two variables, 'x' and 'y', each represented by a light blue box with a dark blue border. Variable 'x' contains the value 2, and variable 'y' contains the value 3. The boxes are positioned to the right of the 'main' function code, corresponding to the local variables 'x' and 'y'.

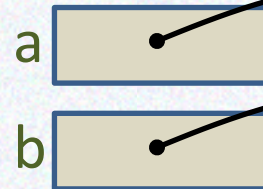
Передача аргументов

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

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

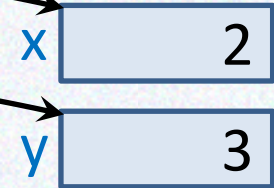
Передача аргументов

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



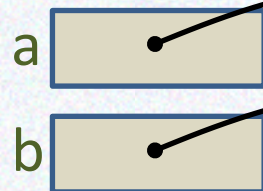
```
int main()
```

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



Передача аргументов

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



```
int main()
```

```
{
```

```
    int x = 2, y = 3;
```

```
    swap(&x, &y);
```

```
    printf("x=%d, y=%d\n", x, y);
```

```
    return 0;
```

```
}
```



x=3, y=2

Передача массива

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

```
... f(тип p[], int n)
{
    ...
}
или
... f(тип *p, int n)
{
    ...
}
```


Задача 4

Дана нуль ограниченная строка символов.

Подсчитать количество символов в строке (длину строки).

Задача 4

```
int slength(char *ptr)
{
    char *p = ptr;
    for(; *p; ++p)
        ;
    return p - ptr;
}
```

Задача 4

```
int main()
{
    char buf[80], c;
    int len;
    while(puts("enter sting..."),
        (len = scanf("%79[^\n]%c", buf, &c)) >= 0){
        if(len == 0){
            *buf = '\0';
            scanf("%*c");
        }
        len = slength(buf);
        printf("string \"%s\" has %d chars\n", buf, len);
    }
    return 0;
}
```