

C++

Отличия

- Регистро-зависимый
- Контроль типов
- Перегрузка операторов
- Классы, объекты, наследование
- Конструкторы, Деструкторы
- Inline, Темплейты
- Исключения

Типы данных

- 1) `bool`
- 2) `char`, `signed char`, `unsigned char`
- 3) `short`, `signed short`, `unsigned short`
- 4) `int`, `signed int`, `unsigned int`
- 5) `long`, `signed long`, `unsigned long`
- 6) `float`, `double`, `long double`
- 7) `void`

НОВЫЕ ТИПЫ

```
typedef ТИП НОВЫЙ_ТИП;
```

Примеры:

```
typedef unsigned int size_t;
```

```
typedef size_t count_t;
```

```
size_t n = 100;
```

```
count_t c = n;
```

Диапазоны значений типов

- `bool` true, false
- `signed char`, `char` $-2^4 \dots 2^4$
- `unsigned char` $0 \dots 2^8$
- `signed short` $-2^8 \dots +2^8$
- `unsigned short` $0 \dots +2^{16}$
- `signed long` $-2^{16} \dots +2^{16}$
- `unsigned long` $0 \dots +2^{32}$
- `long = int`

Объявление типов

- 1) `int i;`
- 2) `unsigned int k, K;`
- 3) `char ch; double d;`
- 4) `int m = 10;`
- 5) `char e, c = 'b';`
- 6) `bool b = false;`
- 7) `int* p1, p2; (int* p1; int p2)`
- 8) `const int b = 100;`

Операции

- ++ увеличение на 1
- -- уменьшение на 1
- -= вычитание с присваиванием
- += сложение ...
- *= умножение ...
- /= деление ...
- %= остаток от деления ...
- ?: при условии (a > b ? 1 : 0)

Приоритеты операций

1) `int b = 5;`

`b += 6; (b = b + 6;)`

2) `int i = 1;`

`int b = i++; (b = 1, i = 2);`

`int c = ++i; (b = 2, i = 2);`

3) `int i = 0;`

Не делать: `++i++;`

`(i+=2);`

Условный оператор if

```
if ( выражение1 )  
    операция1;  
else if(выражение2)  
    операция2;  
....  
else  
    операцияN;
```

Примеры

1) `if(a<b) b = 1;`

2) `if(a<b) {a=2; b=5;}`

3) `if(a<b) b = 1; else {b = 0;}`

4) `if(a<b) b=2; else if (a>b) b=10; else b=0;`

`int a = 1; int b = 1;`

5) `if(a++) b++; (a = 2, b = 2)`

6) `if(++a) b++; (a = 2, b = 2)`

Условный оператор switch

```
switch( выражение)
{
    case константа1: операция; break;
    case константа2: {операции;} break;
    ...
    default: операция; break;
}
```

Примеры

```
int a = 1; int b = 3; int c = 5; int d = 10;  
int x = a;  
switch(x) {  
case 1: x++; break;  
case 3: x++;  
case 5: x++; break;  
default: x--; break;  
}
```

Оператор цикла while

`while` (выражение) оператор;

```
int a = 10;
```

1) `while(0 != a) a--;` (a = 0, [10 ... 0])

2) `while(0 != a) {--a;}` (a = 0, [10 ... 0])

3) `while(0 != a--);` (a = -1, [10 ... 0])

4) `while(0 != --a);` (a = 0, [9 ... 0])

Оператор цикла do while

do оператор; while (выражение);

```
int a = 10
```

```
do
```

```
{
```

```
    a--;
```

```
} while(0 != a); ( a = 0, [9 ... 0] )
```

Оператор цикла for

for(инициализация; выражение; операции)
оператор;

1) **for** (int i = 0; i < 10; i++) ; (i = 9, [0 ... 9])

2) **int** b = 2;

for (int i = 0; i < 100 && 0 != b; i++) b--;
(b = 0, l = 2);

Операторы передачи управления

- goto
- break
- continue
- return

Массивы

```
int buf[128]; int bbuf[100][128];
```

```
char str[] = "Строка";
```

Неправильно: `int n = 10; int buf[n];`

Пример:

```
int m[4] = {10, 15, 20, 30};
```

```
int sum = 0;
```

```
for(int i = 0; i < 4; i++) sum += m[i];
```

Указатели

1) тип (*имя) (список_аргументов);

`int (*func) (double, double);`

2) `тип*` имя;

`int* a, b, *c;`

3) `void*` имя;

`void* p;`

Ссылки

тип& имя;

Пример:

```
int i;
```

```
i = 5;
```

```
int& r = i;
```

```
r = 10;
```

Примеры

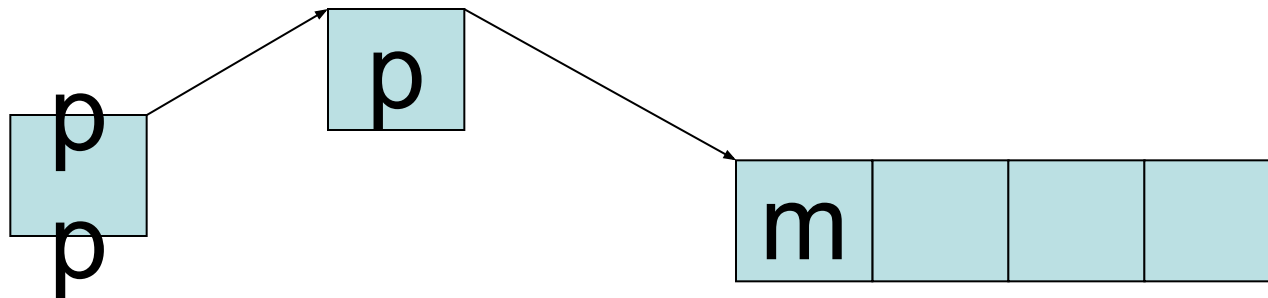
```
int m[4] = {1, 2, 3, 4};
```

```
int* p = m;
```

```
int** pp = &p;
```

1) `int m2 = p[2];` (3 элемент)

2) `int m2 = *(p+2);`



Примеры

```
int i;
```

```
1) int* p = &i;
```

```
2) int* pp = p;
```

```
3) void* vp = (void*)p;
```

```
4) int* const cp = &i;
```

```
5) const int* const ccp = &i;
```

```
6) int* np = NULL;
```

```
7) int* zp = 0;
```

Операции с указателями

```
int m[4] = {1, 2, 3, 4};
```

```
unsigned int* p = m;
```

```
unsigned int* d = p;
```

1) p++;

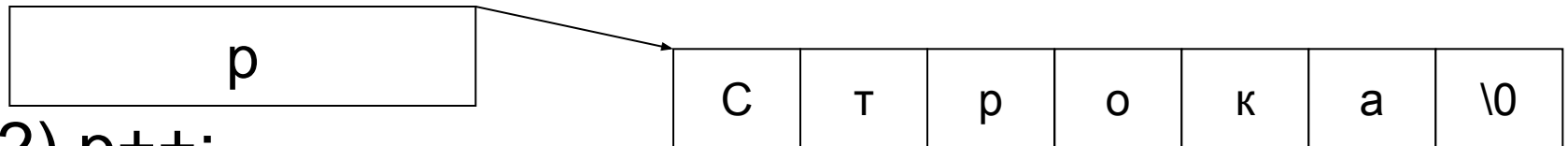
2) int i = p - d; (i = 1, p = m[1])

4) *p = 4; (m {1, 4, 3, 4})

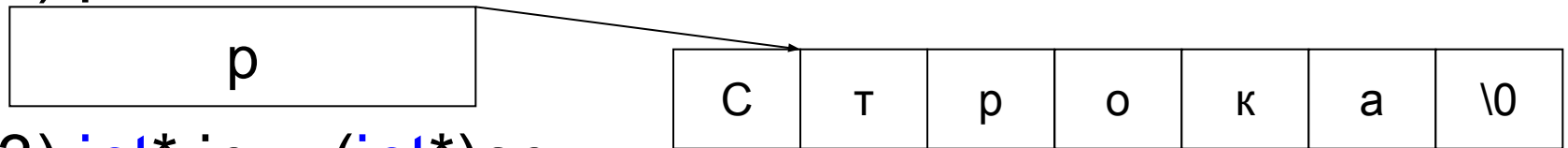
5) p+=2; p--;

Операции с указателями

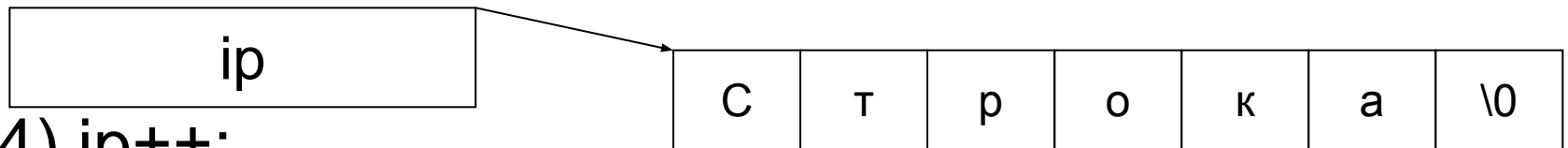
1) `const char*` `cp = "Строка"; char*` `p = (char*)cp;`



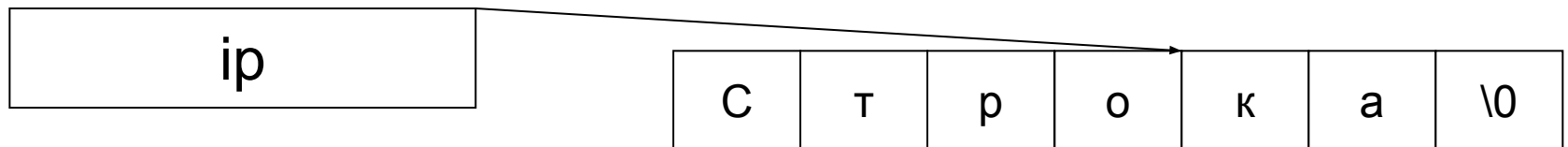
2) `p++;`



3) `int*` `ip = (int*)cp;`



4) `ip++;`



Строки, Символы

- 1) “Строка1” “Строка1\nСтрока2”
- 2) ‘с’, ‘0’, ‘\0’

Примеры:

- 1) `char* str = “Строка1”;` (`const char*`)
- 2) `char ch = ‘с’;`
- 3) `char ch1 = *str;` (`ch1 = ‘C’`);

Операции со строками

```
char* str = "Строка";
```

```
1) int* ip = (int*)str;
```

```
    ip++; (*ip == 'o')
```

```
    Нельзя: *ip = 'м';
```

```
2) char* cp = str;
```

```
    cp++; (*cp == 'т')
```

```
    Нельзя: *cp = 'м';
```

Перебор строки

```
char* p = "Строка";  
while(*p) p++;
```

printf

- 1) `printf("Строка\n");`
- 2) `printf("%s\n", "Строка");`
- 3) `printf("%s %i\n", "Строка", 100);`
- 4) `printf("%s %i%c", "Строка", 100, '\n');`

Структура программы (функции)

```
1) void func()  
{  
}
```

```
2) int func1(void)  
{  
    return 5;  
}
```

```
3)  
bool func2(bool b);  
...  
bool func2(bool b)  
{  
    return !b;  
}
```

Передача объекта

```
void func(string* p, string& r, string s)
{
    *p += 5;
    r +=5;
    s +=5;
}
string s;
func(&s, s, s);
```

Перегрузка функций 1

```
void func(int i)
{
    printf("%i\n", i);
}
```

- 1) func(10);
- 2) func('c');
- 3) func((int)'c');

```
void func(char c)
{
    printf("%c\n", c);
}
```

Перегрузка функций 2

```
void func(const char* p)
{
    printf("1: %s\n", p);
}
```

```
void func(char* p)
{
    printf("2: %s\n", p);
}
```

1) func("Строка");

2) char* p = "Строка";
func(p);

3) func(0);

Структуры

```
struct [имя_типа] {  
    тип1 переменная1;  
    тип2 переменная2;  
    .....  
};
```

```
struct {  
    тип1 переменная1;  
    тип2 переменная2;  
    .....  
} [объекты];
```


Примеры структур

```
struct String {  
    const char* p;  
    size_t len;  
};
```

1) String s;

2) String s = {"Str", 3};

```
struct {  
    const char* p;  
    size_t len;  
} s;
```

```
struct {  
    const char* p;  
    size_t len;  
} s = {"Str", 3};
```