Системное ПО

Системное ПО

Логические выражения и побитовые операции в C++

```
if(1)
{
    printf("1\n");//всегда выполнится
}
if(0)
{
    printf("0\n");//никогда не выполнится
}
if(-1024)
{
    printf("-1024\n");//всегда выполнится
}
```

True и False

- Булевский тип занимает один байт (Visual C++ 5,0 и выше) и имеет целочисленный тип (1 - true, 0 - false)
- При приведении к булевскому типу 0 интерпретируется как false, все отличное от нуля как true

Системное ПО

Логические выражения и побитовые операции в C++

Примеры логических выражений

Первые два выражения истинны, последнее ложно

```
int i=-1024;
if(true||false) printf("Логическое ИЛИ \n");
if(100&&500) printf("Логическое И \n");
if(!i) printf("Логическое НЕ \n");
```

Системное ПО

Логические выражения и побитовые операции в С++

Примеры побитовых операций

```
unsigned char i=13,j=11,k;
// 13=00001101
// 11=00001011
//  9=00001001
// 15=00001111
//240=11110000
k=i&j;
printf("k=%u\n",k);//k=9
k=i|j;
printf("k=%u\n",k);//k=15
k=~(i|j);
printf("k=%u\n",k);//k=240
```

&	Побитовое И
|	Побитовое ИЛИ
~	Побитовое НЕ

Системное ПО

Логические выражения и побитовые операции в С++

Примеры побитовых операций

```
unsigned char k=13;
//Задача - выяснить,
//установлен ли 3-й (справа) бит в единицу
//13=00001101
// 4=00000100
if(k&4) printf("Установлен! \n");
```

Системное ПО

Логические выражения и побитовые операции в С++

Примеры побитовых операций

```
unsigned char k=23;
//Задача - установить 4-й (справа)
//бит в единицу
//23=00010111
// 8=00001000
//31=00011111
k=k|8;
printf("k=%d \n",k);//31
```

Системное ПО

Массивы и арифметика указателей

C++ не отличает указатель и массив. Можно использовать массив как указатель на нулевой элемент.

```
int A[10];//переменная A как массив
A[0]=19;
//Используем массив как указатель
//Указывает на нулевой элемент
printf("k=%d \n",*A);//19
```

Системное ПО

Массивы и арифметика указателей

C++ не отличает указатель и массив. Для использования указателя как массива необходимо только выделить память. Приведение типов использовать не нужно.

```
int * A;//переменная A как указатель
//Выделяем память
A=(int*)malloc(sizeof(int)*10);
//То же самое как объявление int A[10];
//Можно работать с указателем как с массивом
A[5]=18;
printf("k=%d \n",A[5]);//18
```

Системное ПО

Массивы и арифметика указателей

Прибавление к указателю целого числа n аналогично увеличению адреса на

n*(размер типа, на который указывает указатель)

Системное ПО

Массивы и арифметика указателей

Пример использования арифметики указателей

```
int * A;//переменная A как указатель
//Выделяем память
A=(int*)malloc(sizeof(int)*10);
//То же самое как объявление int A[10];
//Можно работать с указателем как с массивом
A[5]=18;
//Используем арифметику указателей
printf("k=%d \n",*(A+5));//18
```

Системное ПО

Массивы и арифметика указателей

```
//Отменяем выравнивание
#pragma pack (push)
#pragma pack (1)
//Объявляем структуру
typedef struct _MyStruct
{
    int a;
    char b;
    double c;
} MyStruct;

int _tmain(int argc, _TCHAR* argv[])
{
//  setlocale( LC_ALL,"Russian" );
    double temp;
    MyStruct Record;
    Record.a=10;
    Record.b=15;
    Record.c=100.43;
    //Обращаемся к полю "c" используя приведение типов
    //и арифметику указателей
    temp=*(double*)(((char*)&Record)+sizeof(int)+sizeof(char));
    printf("temp=%f \n",temp);//100.430000
```

В C++ можно приводить типы, превращая любые указатели в указатели на нужный тип

& - взятие адреса

* - разыменование