|  |  |
| --- | --- |
| Поговорим немного о Булевой алгебре и двоичных вычислениях  Булева алгебра это один из базовых, но вместе с тем один из самых мощных инструментов в программировании. Двоичные вычисления выполняются быстрее десятичных, поскольку являются естественными для цифровой техники. В бинарной алгебре используются операторы И, ИЛИ, НЕ, ИСКЛЮЧАЮЩЕЕ ИЛИ и операции СДВИГА влево и вправо. Работают эти операторы точно также, как и логические, но относительно разрядов двоичного представления чисел, где истина – это единица, а ложь - это ноль. Условия истинности точно такие же как и для логических операторов, повторим:  Оператор И возвращает единицу когда оба операнда единицы  Оператор ИЛИ возвращает единицу когда хотя бы один из операндов единица  Оператор НЕ возвращает единицу когда операнд равен нулю  Оператор ИСКЛЮЧАЮЩЕГО ИЛИ возвращает единицу когда операнды различаются | СЛАЙД С ФОТКОЙ БУЛЯ  СЛАЙД С БИНАРНЫМИ ОПЕРАТОРАМИ  СЛАЙД С ТАБЛИЦАМИ ИСТИННОСТИ ДЛЯ ОПЕРАТОРОВ  /\*  \* 1 & 0 = 0  \* 1 | 0 = 1  \* !1 = 0  \* 0 ^ 1 = 1  \*/ |
| На основе этих знаний мы можем для примера написать программу, меняющую местами значения переменных без использования третьей, вспомогательной и быть уверенными, что переполнения переменных не произойдёт, как это могло бы произойти при использовании сложения и обратного вычитания. *Объявим две переменных a и b, присвоим им значения и выведем их в консоль. Далее подготовим вывод измененных значений a и b в консоль* printf("a = %d, b = %d", a, b);  *и напишем некую конструкцию, которая при детальном изучении не представляет из себя никакой магии*. *В переменную а нужно будет записать a ^ b, в переменную b нужно будет записать b ^ a и в переменную а нужно будет записать a = a ^ b.*  Рассмотрим происходящее пошагово  Число 11 типа char в двоичном представлении это 00001011, число 15 - это 00001111. После выполнения первого оператора в переменную а будет положено промежуточное число 00000100 – это цифра 4. Оператор ксор выполняется следующим образом – результат будет равен 1 если операнды различаются и 0 если они совпадают.  После выполнения второго в переменную b будет положено число 00001011. Здесь b = 15 (00001111), a = 4 (00000100), производим операцию исключающего ИЛИ и получаем 00001011 – т.е. цифру 11.  И после выполнения третьего в переменную а будет положено значение 00001111 – это цифра 15.  Выведем значения переменных в консоль и убедимся в правильности нашего решения. | char a = 11;  char b = 15;  printf("a = %d, b = %d", a, b);  a = a ^ b;  b = b ^ a;  a = a ^ b;  printf("a = %d, b = %d", a, b);  a = a ^ b; //00000100  // a = 11 (00001011)  // b = 15 (00001111)  // a ^ b = 00000100  b = b ^ a; // b = 00001011  // b = 15 (00001111)  // a = 4 (00000100)  // a ^ b = 00001011  a = a ^ b; // a = 00001111  // a = 4 (00000100)  // b = 11 (00001011)  // a ^ b = 00001111  printf("a = %d, b = %d\n", a, b); |
| Операции сдвига бывают логические, арифметические и циклические. В языке С реализован логический сдвиг, то есть недостающие разряды заполняются нулями.  Сдвиг влево на n значений это умножение на 2 в степени n.  Сдвиг вправо на n значений это целочисленное деление на 2 в степени n.  *Итак, нам нужно переменную а сдвинуть влево на 3бита, на самом деле это означает что мы переменную а умножим на 2 в степени 3. А переменную b мы сдвинем вправо на 2бита – это означает, что мы переменную b разделим на 2 в степени 2. Это тоже самое что записать переменная а\*8 и b/4. Просто выполнится это гораздо быстрее.*  Бинарная алгебра это большая и сложная тема. Применять бинарную алгебру можно и в высокоуровневых проектах, поддержка бинарных операций есть в подавляющем числе языков программирования. | a = a << 3; // 15 \* 8  b = b >> 2; // 11 / 4  printf("a = %d, b = %d", a, b); |
| На следующем уроке мы начнём говорить о циклах. До новых встреч коллеги! |  |