Битовые операции

№ урока: 17 **Курс:** Процедурное программирование на языке C#

Средства обучения: Visual Studio 2019 Community Edition

Обзор, цель и назначение урока

Задача данного урока состоит в том, чтобы познакомить слушателя с различными операциями, которые могут проводиться над отдельными битами при записи значения переменных в оперативной памяти, иначе говоря - с битовыми операциями.

Изучив материал данного занятия, учащийся сможет:

- Понимать и уметь применять логические побитовые операции над значениями переменных.
- Уметь работать с отдельными битами в переменных.

Содержание урока

1. Оператор побитового дополнения
2. Целочисленные логические операторы
3. Операции сдвига

Резюме

- **Оператор побитового отрицания** (еще называют оператором побитового дополнения или оператором побитового инвертирования) - заменяет в байте каждый ноль на единицу, а единицу на ноль. Другими словами, этот оператор инвертирует значение каждого бита в байте. Инвертировать байт, это и есть произвести замену всех единиц на нули, а нулей на единицы.
- Оператор побитового отрицания, представлен значком **~**, который называется «тильда».
- Если взять переменную знакового типа данных, инвертировать хранящееся в переменной число и прибавить к инвертированному значению единицу, то мы получим число с противоположным знаком - «инверт-плюс-один».
- Формула изменения знака числа: ~ (+N) + 1 = -N и ~ (-N) + 1 = +N -
- Целочисленные битовые операторы воспринимают все двоичные единицы как true (истину), а все двоичные нули, воспринимают как false (ложь).
- **Побитовое отрицание** (или побитовое НЕ, или дополнение) – это унарная операция, действие которой эквивалентно применению логического отрицания к каждому биту числа, представленного в двоичной форме.
- **Побитовое И** – это бинарная операция, действие которой эквивалентно применению логического «И» к каждой паре битов, которые стоят на одинаковых позициях в двоичных представлениях операндов. Другими словами, если оба соответствующих бита операндов равны 1, результирующий двоичный разряд равен 1; если же хотя бы один бит из пары равен 0, результирующий двоичный разряд равен 0.
- **Побитовое ИЛИ** – это бинарная операция, действие которой эквивалентно применению логического «ИЛИ» к каждой паре битов, которые стоят на одинаковых позициях в двоичных представлениях операндов. Другими словами, если оба соответствующих бита операндов равны 0, двоичный разряд результата равен 0; если же хотя бы один бит из пары равен 1, двоичный разряд результата равен 1.
- **Побитовое исключающее ИЛИ** (или побитовое сложение по модулю два) – это бинарная операция, действие которой эквивалентно применению логического исключающего «ИЛИ» к каждой паре битов, которые стоят на одинаковых позициях в двоичных представлениях операндов. Другими словами, если соответствующие биты операндов различны, то двоичный разряд результата равен 1; если же биты совпадают, то двоичный разряд результата равен 0.

CyberBionic systematics

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Оператор побитовой конъюнкции, не возвращает byte. Он возвращает только типы int, uint, long и ulong.
- **Битовая маска** — это определённые данные, которые используются для выбора отдельных битов или полей из нескольких битов из двоичной строки или числа (маскирования).
- **Битовые сдвиги** — это битовые операции, при которых значения битов копируются в соседние по направлению сдвига. Различают несколько видов сдвигов – логический, арифметический и циклический, в зависимости от обработки крайних битов.
- **Логический сдвиг**. При логическом сдвиге значение последнего бита по направлению сдвига теряется (копируясь в бит переноса), а первый приобретает нулевое значение. Логические сдвиги влево и вправо используются для быстрого умножения и деления на 2, соответственно.
- Оператор сдвига влево (**<<**) сдвигает первый операнд влево, в соответствии с количеством бит, заданным вторым операндом. Тип второго операнда должен быть int или тип, имеющий предопределенное неявное числовое преобразование в int.
- Если тип первого операнда – int или uint (32-разрядное число), начало сдвига задается пятью младшими разрядами второго операнда. Фактический сдвиг от 0 до 31 бит.
- Если тип первого операнда – long или ulong (64-разрядное число), начало сдвига задается шестью младшими разрядами второго операнда. Фактический сдвиг от 0 до 63 бит.
- Старшие разряды, которые находятся не в диапазоне типа первого операнда, после смены отбрасываются, а пустые младшие разряды заполняются нулями. Операторы сдвига никогда не вызывают переполнений.
- Оператор сдвига вправо (**>>**) сдвигает первый операнд вправо в соответствии с количеством бит, заданным вторым операндом.
- Если тип первого операнда – int или uint (32-разрядное число), начало сдвига задается пятью младшими разрядами второго операнда (второй операнд & 0x1f).
- Если тип первого операнда – long или ulong (64-разрядное число), начало сдвига задается пятью младшими разрядами второго операнда (второй операнд & 0x3f).
- Если тип первого операнда – int или long, сдвиг вправо является арифметическим сдвигом (пустым старшим разрядам задан знаковый бит). Если тип первого операнда – long или ulong
- Если тип первого операнда – uint или ulong, сдвиг вправо является логическим сдвигом
- **Арифметический сдвиг**. Арифметический сдвиг аналогичен логическому, но значение слева считается знаковым числом, представленным в дополнительном коде. Так, при правом сдвиге старший бит сохраняет свое значение. Левый арифметический сдвиг идентичен логическому. В языке C# арифметический сдвиг отсутствует.
- **Циклический сдвиг**. При циклическом сдвиге, значение последнего бита по направлению сдвига копируется в первый бит (и копируется в бит переноса). Также различают циклический сдвиг через бит переноса – при нём первый бит по направлению сдвига получает значение из бита переноса, а значение последнего бита сдвигается в бит переноса. В языке C# циклический сдвиг отсутствует.
- Операторы сдвига применяются для того, чтобы мы могли узнать значение того или иного бита в байте.
- Различают сдвиг влево (в направлении от младшего бита к старшему) и вправо (в направлении от старшего бита к младшему).
- Бинарные операторы & являются предопределенными для целых типов и bool. Для целых типов оператор & выполняет битовую операцию логического умножения операндов. Для операндов bool оператор & выполняет операцию логического умножения операндов, то есть, если оба оператора – true, результатом будет являться значение true иначе false.

Закрепление материала

- Что такое побитовая конъюнкция?
- Что такое побитовая дизъюнкция?
- Что такое побитовое логическое отрицание?
- Что такое побитовое исключающее «ИЛИ»?

CyberBionic systematics

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- В чем отличия побитовых логических операций от обычных?
- Для чего применяется логический сдвиг?
- Что такое битовая маска и для чего она применяется?

Самостоятельная деятельность учащегося

- Задание 1.

Ознакомьтесь с дополнительными материалами к уроку.

- Задание 2

Имеется 3 переменные типа int x = 5, y = 10, и z = 15;
Выполните и рассчитайте результат следующих операций для этих переменных:
x += y >> x++ * z;
z = ++x & y * 5;
y /= x + 5 | z;
z = x++ & y * 5;
x = y << x++ ^ z;

- Задание 3

Напишите программу, выполняющую «зашифровку» и «расшифровку» введенного пользователем символа. Ключом шифрования считайте число 216. Выведите на экран консоли зашифрованный символ и расшифрованный.

Рекомендуемые ресурсы

Побитовые операторы и операторы сдвига (справочник по C#)
https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/bitwise-and-shift-operators

https://ru.wikipedia.org/wiki/Битовая_маска

https://ru.wikipedia.org/wiki/Битовый_сдвиг

CyberBionic systematics

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com