

Логические и побитовые операции

№ урока: 5 **Курс:** Java Starter

Средства обучения: Компьютер с установленной IntelliJ IDEA

Обзор, цель и назначение урока

Рассмотрение логических функций: конъюнкция, дизъюнкция, отрицание, исключающее ИЛИ.
Использование побитовых логических операций.
Рассмотрение логических операций.
Рассмотрение операторов сдвига.
Короткозамкнутые вычисления.
Теоремы Де Моргана.

Изучив материал данного занятия, учащийся сможет:

- Понимать работу логических операторов.
- Понимать работу битовых логических операторов.
- Понимать работу операторов сдвига.
- Использовать короткозамкнутые вычисления и теоремы Де Моргана.

Содержание урока

1. Рассмотрение логических функций: конъюнкция, дизъюнкция, отрицание, исключающее ИЛИ.
2. Рассмотрение побитовых логических операций.
3. Рассмотрение примера: Побитовые логические операции.
4. Рассмотрение примера: Использование побитовых логических операций, для установки и сброса флагов.
5. Рассмотрение примера: Использование XOR для шифрования данных.
6. Рассмотрение логических операций (пропозициональная логика).
7. Рассмотрение примера: Комбинация логических операций с операциями сравнения.
8. Рассмотрение примера: Логический сдвиг влево, вправо, а также беззнаковый сдвиг вправо.
9. Рассмотрение примеров: Короткозамкнутые вычисления.
10. Теоремы Де Моргана (Laws of dualization).
11. Рассмотрение примера: Теоремы Де Моргана.

Резюме

- Конъюнкция (от лат. conjunctio союз, связь) – логическая операция, по своему применению максимально приближённая к союзу "и". Синонимы: логическое "И", логическое умножение, иногда просто "И".
- Дизъюнкция – (лат. disjunctio – разобщение) логическая операция, по своему применению максимально приближённая к союзу «или» в смысле «или то, или это, или оба сразу». Синонимы: логическое «ИЛИ», включающее «ИЛИ», логическое сложение, иногда просто «ИЛИ».
- Исключающее ИЛИ (логическое сложение, строгая дизъюнкция) – булева функция и логическая операция. Результат выполнения операции является истинным только при условии, если является истинным в точности один из аргументов.
- Отрицание в логике – унарная операция над суждениями, результатом которой является суждение (в известном смысле) «противоположное» исходному. Обозначается знаком ~ перед или чертой над суждением. Синоним: логическое "НЕ".
- Побитовое отрицание (или побитовое НЕ, или дополнение) – это унарная операция, действие которой эквивалентно применению логического отрицания к каждому биту двоичного представления операнда. Другими словами, на той позиции, где в двоичном представлении операнда был 0, в результате будет 1, и, наоборот, где была 1, там будет 0.

- 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.
- Битовые сдвиги относят к битовым операциям. При сдвиге значения битов копируются в соседние по направлению сдвига. Различают несколько видов сдвигов – логический, арифметический и циклический, в зависимости от обработки крайних битов. Также 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 `boolean` оператор & выполняет операцию логического умножения операндов, то есть, если оба оператора – `true`, результатом будет являться значение `true` иначе `false`.
- Оператор & вычисляет оба оператора независимо от значения первого из них.
- Для целочисленных типов | вычисляет результат битовой операции ИЛИ для своих операндов. Для операндов `boolean` | выполняет операцию логического ИЛИ для своих операндов, то есть результатом будет значение `false` тогда и только тогда, когда оба операнда имеют значение `false`.
- Оператор ^ выполняет побитовую операцию исключающего OR его операндов. Для операндов `boolean` оператор ^ выполняет операцию логического исключающего OR операндов, то есть результатом будет являться значение `true` только в том случае, если ровно один из его операндов имеет значение `true`.
- 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`.
- Старшие разряды, которые находятся не в диапазоне тип 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 число отрицательное (со знаком -), то старшие(слева) разряды заполняются единицами, а младшие соответственно теряются. Если же число положительное, то старшие разряды заполняются нулями, младшие – теряются.
- Оператор сдвига вправо (>>>) сдвигает первый операнд вправо в соответствии с количеством бит, заданным вторым операндом. При использовании беззнакового сдвига, **ВСЕГДА** старшие разряды заполняются нулями вне зависимости от знака числа (+ или же -)
- Если тип первого операнда – `int` или `long`, сдвиг вправо является арифметическим сдвигом (пустым старшим разрядам задан знаковый бит). Если тип первого операнда – `long`, начало сдвига задается шестью младшими разрядами второго операнда (второй операнд & 0x3f).
- Условный оператор AND (&&) выполняет логическое AND своих операндов типа `boolean`, но вычисляет только второй операнд при необходимости.
- Условный оператор OR (||) выполняет логическое OR своих операндов типа `boolean`, но вычисляет только второй операнд при необходимости.

- Короткозамкнутое вычисление – техника работающая по следующему принципу: Если значение первого операнда в операции AND (&&) ложно, то второй операнд не вычисляется, потому что полное выражение в любом случае будет ложным.
- Или если значение первого операнда в операции OR (||) истинно, то второй операнд не вычисляется, потому что полное выражение в любом случае будет истинным.
- Для применения теорем Де Моргана к логическому оператору AND или OR и паре операндов, требуется инвертировать оба операнда, заменить (AND на OR) или (OR на AND) и инвертировать все выражение полностью.

Закрепление материала

- Назовите основные логические функции.
- Расскажите таблицу истинности конъюнкции.
- Расскажите таблицу истинности дизъюнкции.
- Расскажите таблицу истинности исключающего ИЛИ.
- Где и для чего используются логические операции?
- Где и для чего используются побитовые логические операции?
- Где и для чего используются сдвиги?
- Что такое короткозамкнутые вычисления?
- Какие вы знаете короткозамкнутые вычисления?
- Дайте определение теорем Де Моргана.

Дополнительное задание

Задание 1

Известно, что у чисел, которые являются степенью двойки, только один бит имеет значение 1.

Используя IntelliJ IDEA, создайте класс **PowerOfTwo**.

Напишите программу, которая будет выполнять проверку – является ли указанное число степенью двойки или нет.

Задание 2

Используя IntelliJ IDEA, создайте класс **DeMorganComparison**.

Используя теорему Де Моргана, преобразуйте исходное выражение $A \mid B$, в эквивалентное выражение.

Самостоятельная деятельность учащегося

Задание 1

Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2

Известно, что у четных чисел младший бит имеет значение 0

Используя IntelliJ IDEA, создайте класс **Parity**.

Напишите программу, которая будет выполнять проверку чисел на четность.

Предложите два варианта решения поставленной задачи.

Задание 3

Используя IntelliJ IDEA, создайте класс **LogicOperations**.

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 4

Используя IntelliJ IDEA, создайте класс **Premium**.

Напишите программу расчета начисления премий сотрудникам. Премии рассчитываются согласно выслуге лет.

Если выслуга до 5 лет, премия составляет 10% от заработной платы.

Если выслуга от 5 лет (включительно) до 10 лет, премия составляет 15% от заработной платы.

Если выслуга от 10 лет (включительно) до 15 лет, премия составляет 25% от заработной платы.

Если выслуга от 15 лет (включительно) до 20 лет, премия составляет 35% от заработной платы.

Если выслуга от 20 лет (включительно) до 25 лет, премия составляет 45% от заработной платы.

Если выслуга от 25 лет (включительно) и более, премия составляет 50% от заработной платы.

Результаты расчета, выведите на экран.

Рекомендуемые ресурсы

Логические операторы

<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/op3.html>

Все операторы Java

<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/opsummary.html>