**Логические основы алгоритмизации**

*Логика* – это наука о формах и способах мышления (первые учения – Древний Восток).

Начало исследований в области формальной логики было положено Аристотелем в IV в. до н.э. Однако математические подходы к этим вопросам были впервые указаны Джорджем Булем, который положил в основу математической логики алгебру логики (булеву, а логические значения называют булевыми). Алгебра логики используется при построении основных узлов ЭВМ, например, таких как шифратор, сумматор и др.

Основными формами мышления являются *понятие*, *высказывание* и *умозаключение*.

**Понятие** – это форма мышления, фиксирующая основные, существенные признаки объекта (например, прямоугольник, компьютер).

**Умозаключение** – это форма мышления, с помощью которой из одного или нескольких суждений может быть получено новое суждение – заключение (например, все углы равнобедренного треугольника равны → это треугольник равносторонний).

Составляющей алгоритмов являются логические условия, вычисление значений которых происходит в соответствии с аксиомами алгебры логики.

Основу математической логики составляет алгебра высказываний.

Объектами, с которыми работает алгебра высказываний, являются повествовательные предложения, относительно которых можно сказать, истинны они или ложны.

**Высказыванием** называется утверждение, о котором можно определенно сказать, истинно оно или ложно. Высказываний одновременно истинных и ложных не бывает.

Приведем примеры высказываний:

1. Москва - столица России
2. число 27 является простым
3. Волга впадает в Каспийское море

Высказывания 1 и 3 являются истинными. Высказывание 2 - ложным, потому что число 27 составное 27=3*3*3.

Следующие предложения высказываниями не являются:

1. давай пойдем гулять
2. 2\*x>8
3. a*x2+b*x+c=0
4. который час?

Подчеркнем еще раз, что отличительным признаком высказывания является свойство быть истинным или ложным, последние четыре предложения этим свойством не обладают. Невозможно отнести неравенство 2 или уравнение 3 к высказываниям пока не определено значение x. При x=3 высказывание "2\**3>8" ложно, а при x=5 "2\**5>8" - истинно.

Условимся обозначать высказывания большими буквами и, следуя Джорджу Булю, истинное (true) высказывание A обозначим так, A=1. В том случае, когда A - ложное (false) высказывание, будем писать: A=0.

Функция ƒ(x1, x2, ..., xn) называется логической или булевой, если она, так же как и ее аргументы xi, может принимать только два значения: "истина" или "ложь".  
По числу переменных различают логические функции одной, двух и многих переменных.

Логические функции могут быть описаны различными способами:

* в виде таблицы истинности;
* совершенными нормальными формами;
* в виде формулы.

Чаще всего встречаются табличная форма представления логической функции (в виде таблицы истинности) и ее аналитическое представление (например, в виде дизъюнктивных или конънктивных форм).

Из простых высказываний можно строить сложные, называемые составными высказывания, соединяя простые логическими операциями. Над простыми высказываниями определены следующие операции:

1. логическое отрицание
2. логическое умножение
3. логическое сложение
4. логическое следование или импликация
5. эквивалентность

Рассмотрим некоторые из этих операций более подробно.

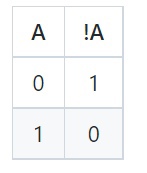
**Логические операции**

**Инверсия (логическое отрицание - НЕ) - обозначение !.**

Логическое сложение, умножение, следование и эквивалентность являются бинарными операциями ("би" - два), потому что соединяют два операнда (два высказывания). В отличие от них, логическое отрицание является унарной операцией, потому что применяется лишь к одному высказыванию.

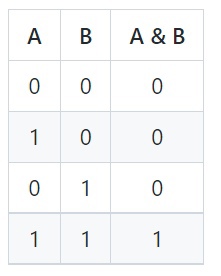
Присоединение частицы НЕ к сказуемому простого высказывания A называется операцией логического отрицания.

Результат будет истинным, если исходное высказывание ложно, и наоборот, ложным - если исходное высказывание истинно.



**Конъюнкция (логическое умножение - И) – обозначение & или &&**

Результат будет истинным тогда и только тогда, когда оба исходных высказывания истинны.



Для более легкого запоминания этой функции следует придерживаться правила: функция конъюнкции ложна, если ложен хотя бы один из ее операндов. Это правило действует для функции, содержащей произвольное число операндов. Если обозначать значение "истина" как "1", а значение "ложь" как "0", то эта функция будет похожа на математическую функцию умножения. Поэтому ее зачастую называют функцией логического умножения.

Отметим, что для конъюнкции, так же как и для следующей рассматриваемой функции – дизъюнкции – действуют ассоциативный (сочетательный) и коммуникативный (переместительный) законы.

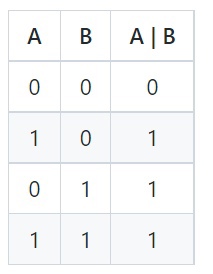
В то же время для некоторых других логических функций тот или иной закон не действует. Некоторые из примеров таких функций мы рассмотрим ниже.

Чем отличается оператор **&** от **&&**:

* Оператор **&** всегда вычисляет оба операнда
* Оператор **&&** вычисляет второй операнд, только если это необходимо.

**Дизъюнкция (логическое сложение - ИЛИ) – обозначение | или ||.**

Результат будет истинным тогда, когда истинно хотя бы одно из высказываний.



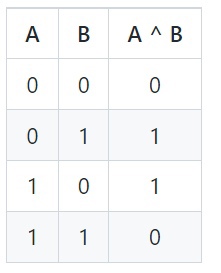
Для более легкого запоминания этой функции следует придерживаться правила: функция дизъюнкции истинна, если истенен хотя бы один из ее операндов. Это правило действует для функции, содержащей произвольное число операндов. Если обозначать значение "истина" как "1", а значение "ложь" как "0", то эта функция для двух операндов будет похожа на математическую функцию поразрядного сложения. Поэтому ее зачастую называют функцией логического сложения. Хотя здесь следует иметь в виду, что в логическом сложении сигнал переноса в более старший разряд не вырабатывается.

Чем отличается оператор **|** от **||**:

* Оператор **|** всегда вычисляет оба операнда
* Оператор **||** вычисляет второй операнд, только если это необходимо.

**Сложе́ние по мо́дулю 2 (исключа́ющее «ИЛИ», логи́ческая неравнозна́чность) - обозначение ^.**

Таблица истинности

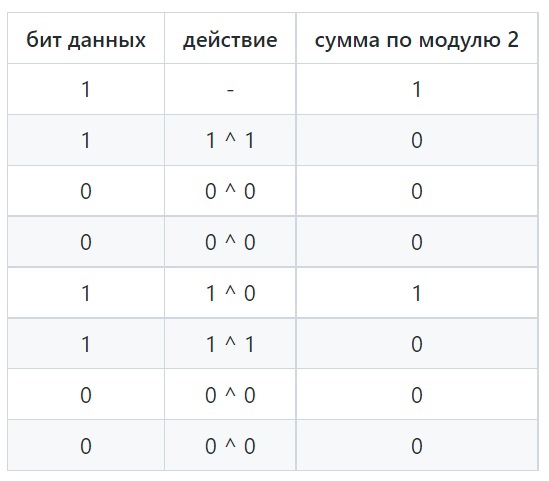


Для операндов целочисленных типов оператор ***^*** вычисляет побитовое исключающее ИЛИ своих операндов.

Про функцию *Сумма по модулю 2* поговорим здесь поподробнее, так она является основой для реализации устройств контроля и защиты информации.

**Пример контроля**

Пусть мы хотим передать некоторые данные, например, код 11001100. Перед началом передачи код дополняется контрольным разрядом.



В нашем примере контрольный разряд будет равен 0 (сложим 4 единицы и разделим по модулю 2, результат 0). Контрольный разряд передается вместе с основным кодом (11001100+0).

Если в процессе передачи произойдет искажение одного разряда (например, последний разряд кода примет значение 1), то приёмное устройство примет код 11001101 и контрольный разряд "0", равный исходному (там искажения нет).



Приемное устройство вычисляет контрольный разряд от принятого кода (теперь он будет равен "1", 5 mod 2 = 1), сравнивает его с принятым контрольным разрядом (они не совпадают) и сообщает об ошибке в передаче данных. Обычно после этого передача повторяется.

Такой алгоритм только демонстрирует принцип формирования контрольных сумм. В современной технике никто, разумеется, по байтам данные не передает. Контроль обычно осуществляется на уровне пакета данных добавлением циклического кода целостности (CRC16, CRC32).

**Пример защиты**

*A ^ B = C, C ^ B = A !!!  
Почему то нигде не написано про самое интересное свойство этой функции: если к результату повторно применить аналогичное преобразование, то получим исходные данные.*

Эту логическую операцию можно применять для элементарной защиты передаваемой информации. Можно использовать так называемый ключ защиты, который должен быть у передающей и принимающей стороны. Пусть в нашем случае таким ключом защиты будет код 01101101.



Суммируя по модулю 2 поразрядно передаваемый код и код защиты, получим, что передаваться будет код 10100001, который отличается от исходного кода. Даже если в случае несанкционированного доступа эта информация будет перехвачена, она ничего не даст злоумышленнику. В то же время, принимающая сторона, имея тот же самый ключ защиты, восстановит исходную информацию.

Если длина передаваемых данных больше длины ключа, то ключ используется повторно для следующего блока данных.

**Свойства функции *сложения по модулю 2***

1. Коммутативность.A ^ B = B ^ A
2. Ассоциативность.A ^ (B ^ C) = (A ^ B) ^ C
3. Дистрибутивность относительно конъюнкции.A & (B ^ C) = A & B ^ A & C
4. ИдемпотентностьA ^ 0 = A
5. ОтрицаниеA ^ 1 = !A

Имеют место следующие очевидные соотношения:

A ^ A = 0

A ^ !A = 1

Википедия знает еще несколько логических операций *Стре́лка Пи́рса* (ИЛИ-НЕ), *Штрих Ше́ффера* (И-НЕ), *Логическое следование* (импликация, "если то"), *Логическая равнозначность или эквивале́нция* (эквивале́нтность).  
В программировании они не используются, поэтому останавливаться на них мы не будем.

**Приоритет логических операций**

* Инверсия ( **!** )
* Конъюнкция ( **&** )
* Сложение по модулю 2 ( **^** )
* Дизъюнкция ( **|**)
* Условная конъюнкция ( **&&** )
* Условная дизъюнкция ( **||** )

Порядок вычисления, определяемый приоритетом операторов, можно изменить с помощью скобок (()).

**Законы логических операций**

В алгебре логики доказано, что любую логическую функцию можно выразить только через комбинацию логических операций И, ИЛИ и НЕ. Для приведения логических выражений к эквивалентным, но более простым в записи используют ряд логических законов.

1. *Закон противоречия* говорит о том, что никакое предложение не может быть истинно одновременно со своим отрицанием. "это яблоко спелое" и "это яблоко неспелое".А & !А = 0
2. *Закон исключенного третьего* говорит о том, что для каждого высказывания имеются лишь две возможности: это высказывание либо истинно, либо ложно. Третьего не дано. "Сегодня я либо получу 5, либо не получу". Истинно либо суждение, либо его отрицание.А | !А = 1
3. *Закон двойного отрицания* заключается в том, что отрицать отрицание какого-нибудь высказывания - то же, что утверждать это высказывание.!(!А) = А
4. *закон идемпотентности* говорит о том, что в алгебре логики нет показателей степеней и коэффициентов. Конъюнкция одинаковых "сомножителей" равносильна одному из них. Дизъюнкция одинаковых "слагаемых" равносильна одному из них.А & А = А и А | А = А
5. *Законы коммутативности и ассоциативности* говорят о том, что конъюнкция и дизъюнкция аналогичны одноименным знакам умножения и сложения чисел.коммутативность:А & В = В & А и А | В = В | Аассоциативность:А | (В | С) = (А | В) | С и А & (В & С) = (А & В) & С
6. *Законы дистрибутивности* говорят о том, что логическое сложение и умножение равноправны по отношению к дистрибутивности: не только конъюнкция дистрибутивна относительно дизъюнкции, но и дизъюнкция дистрибутивна относительно конъюнкции.А & (В | С) = (А & В) | (А & С)А | (В & С) = (А | В) & (А | С)
7. *Законы поглощения констант* утверждают, что ложь не влияет на значение логического выражения при дизъюнкции, а истина - при конъюнкции.А & 1 = А и А | 0 = А
8. *Законы поглощения* показывают как упрощать логические выражения при повторе операнда.A | (A & B) = A и A & (A | B) = A
9. *Законы де Мо́ргана* (правила де Мо́ргана) — логические правила, связывающие пары логических операций при помощи логического отрицания. Названы в честь шотландского математика Огастеса де Моргана. В краткой форме звучат так:Отрицание конъюнкции есть не что иное, как дизъюнкция отрицаний.!(А & В) = !А | !ВОтрицание дизъюнкции есть не что иное, как конъюнкция отрицаний.!(А | В) = !А & !В

***Задание:****построить таблицы истинности для каждого закона и проверить эти законы.*

**Решение логических выражений**

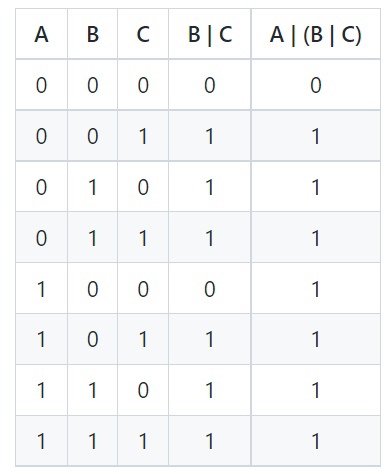
Решение логических выражений записывают в виде таблиц истинности – таблиц, в которых по действиям показано, какие значения принимает логическое выражение при всех возможных наборах его переменных.

Для составления таблиц истинности необходимо:

* определить количество строк в таблице: ***2n***, n–количество переменных
* определить количество столбцов в таблице: количество логических переменных + количество логических операций
* установить последовательность выполнения логических операций
* построить таблицу, указывая названия столбцов и возможные наборы значений исходных логических переменных
* заполнить таблицу истинности по столбцам

Например, построим таблицу истинности для ассоциативного закона дизьюнкции: А | (В | С)

* количество строк: ***2\*3=8***
* количество столбцов: ***3+2=5***



**Вопросы для конспектирования**

1. Основные формы мышления.
2. Что такое логическое высказывание?
3. Способы описания логических функций.
4. Основные логические операции.
5. Приоритет логических операций
6. Законы логических операций
7. Проверка законов построением таблиц истинности