Логика и множества

Волченко Ю.М.

Содержание лекции

Высказывания. Операции над высказываниями. Тавтологии. Теоремы. Предикаты. Множества, операции над ними, свойства операций. Отношения между множествами. Свойства отношений.

Анимация переключательной схемы.
Анимация работает только в программе Acrobat Reader!

Знакомство с системой *Mathematica*. Применение системы *Mathematica* для проверки истинности высказываний и выполнения операций над ними, для решения простых задач теории множеств.

27 декабря 2011 г.

1 Элементы математической логики

В курсе высшей математики нам во многих случаях будет удобно пользоваться короткими записями утверждений на языке математической логики. На естественном, в частности, русском языке такие утверждения выглядели бы длинно и запутанно. Это будет относиться только к утверждениям, о которых можно точно сказать, истинны они или ложны. В математической логике их называют ***высказываниями***.

Из высказываний можно составлять другие высказывания с помощью специальных логических операций. Пусть α, β, ... – некоторые высказывания.

Высказывание $\overline{\alpha}$ означает ***отрицание*** высказывания α. Например, если $\alpha =$ «Число делится на три», то $\overline{\alpha} =$ «Число не делится на три».

Высказывание $\alpha \vee \beta$ означает «α или β» и называется ***дизъюнкцией***. В русском языке союз «или» бывает исключающим, как в предложении: «Я пойду в кино или в театр», в котором имеется в виду только одна из двух возможностей, а бывает неисключающим, как в предложении: «Число x либо четно, либо делится на 3» (одно не исключает другое). В высказывании $\alpha \vee \beta$ всегда используется неисключающий союз «или».

Высказывание $\alpha \wedge \beta$ означает «α и β» и называется ***конъюнкцией***. Если $\alpha =$ «Число делится на два», а $\beta =$ «Число делится на три», то $\alpha \wedge \beta =$ «Число делится на шесть».

Высказывание $\alpha \Longrightarrow \beta$ означает: «Из утверждения α следует утверждение β», «Если α, то β» и называется ***следованием***, или ***импликацией***. Высказывание α называется ***достаточным*** условием для β, или ***посылкой***. Высказывание β называется ***необходимым*** условием для α, или ***следствием***. Например, рассмотрим утверждения $\alpha =$ «Число делится на 49», $\beta =$ «Число делится на 7». Тогда $\alpha \Longrightarrow \beta$ означает утверждение, что, если число делится на 49, то оно делится и на 7.

Высказывание $\alpha \Longleftrightarrow \beta$ означает: «Утверждение α равносильно утверждению β» и называется ***эквивалентностью***. В теоремах эта операция описывается словами «тогда и только тогда», «если и только если», «необходимо и достаточно». Например, теорема Пифагора может быть записана как $\alpha \Longleftrightarrow \beta$, где $\alpha =$ «Треугольник – прямоугольный», $\beta =$ «Квадрат бóльшей стороны треугольника равен сумме квадратов меньших сторон».

Каждое сложное высказывание можно интерпретировать как своеобразную формулу, которая может иметь одно из двух значений: 1, если формула истинна, и 0, если формула ложна, в зависимости от значений простых высказываний. Истинность или ложность высказывания можно доказать с помощью таблиц истинности. Для рассмотренных логических операций таблицы истинности имеют аксиоматический характер (см. табл. 1).

Таблица 1.

α	β	$\overline{\alpha}$	$\alpha \vee \beta$	$\alpha \wedge \beta$	$\alpha \Longrightarrow \beta$	$\alpha \Longleftrightarrow \beta$
1	1	0	1	1	1	1
1	0	0	1	0	0	0
0	1	1	1	0	1	0
0	0	1	0	0	1	1

Таблица показывает, что, например, высказывание $\alpha \wedge \beta$ принимает значение 1, когда высказывания α и β равны 1; то есть, высказывание $\alpha \wedge \beta$ истинно, если истинны оба высказывания α и β.

Наибольший интерес вызывают высказывания, истинные при любых значениях высказываний, из которых они состоят (их называют ***тождественно истинными***, или ***тавтологиями***). Например, высказывание

$$(\alpha \Longleftrightarrow \beta) \Longleftrightarrow ((\alpha \Longrightarrow \beta) \wedge (\beta \Longrightarrow \alpha)) \qquad (1)$$

истинно при любых значениях α и β. Высказывание означает, что α равносильно β тогда и только тогда, когда из α следует β и из β следует α. Обратимся к таблице:

α	β	$\alpha \Rightarrow \beta$	$\beta \Rightarrow \alpha$	$(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$	$\alpha \Leftrightarrow \beta$	$(\alpha \Leftrightarrow \beta) \Leftrightarrow ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$
1	1	1	1	1	1	1
1	0	0	1	0	0	1
0	1	1	0	0	0	1
0	0	1	1	1	1	1

Последний столбец демонстрирует, что высказывание (1) истинно при любых α и β и потому тождественно истинно.

Если $\alpha \Longleftrightarrow \beta$ представляет собой теорему, то формула (1) показывает, что доказательство такой теоремы может быть реализовано следующим образом: сначала надо доказать необходимость ($\alpha \Longrightarrow \beta$), а потом – достаточность ($\beta \Longrightarrow \alpha$). Например, для доказательства рассмотренной выше теоремы Пифагора надо сначала доказать необходимость: «Если треугольник – прямоугольный, то квадрат его бóльшей стороны равен сумме квадратов меньших сторон», а потом достаточность: «Если квадрат бóльшей стороны треугольника равен сумме квадратов меньших сторон, то такой треугольник – прямоугольный».

Если утверждение $\alpha \Longrightarrow \beta$ является теоремой, то ее называют ***прямой теоремой***, утверждение $\beta \Longrightarrow \alpha$ – ***обратной*** теоремой, а утверждение $\overline{\alpha} \Longrightarrow \overline{\beta}$ – ***противоположной*** теоремой.

Обратная теорема не обязательно справедлива, если справедлива прямая теорема, т.е. $(\alpha \Longrightarrow \beta) \Leftrightarrow (\beta \Longrightarrow \alpha)$ не является тавтологией. Например это так для $\alpha =$ «Число делится на 49» и $\beta =$ «Число делится на 7». Из α следует β, но из β не следует α.

Точно так же противоположная теорема не обязательно справедлива, если справедлива прямая теорема, т.е. $(\alpha \Longrightarrow \beta) \Longleftrightarrow \left(\overline{\alpha} \Longrightarrow \overline{\beta}\right)$ не является тавтологией. Предыдущий пример это иллюстрирует.

Как показывает таблица истинности

α	β	$\overline{\beta}$	$\overline{\alpha}$	$\alpha \Longrightarrow \beta$	$\overline{\beta} \Longrightarrow \overline{\alpha}$	$(\alpha \Longrightarrow \beta) \Longleftrightarrow \left(\overline{\beta} \Longrightarrow \overline{\alpha}\right)$
1	1	0	0	1	1	1
1	0	1	0	0	0	1
0	1	0	1	1	1	1
0	0	1	1	1	1	1

тождественно истинной является высказывание

$$(\alpha \Longrightarrow \beta) \Longleftrightarrow \left(\overline{\beta} \Longrightarrow \overline{\alpha}\right),$$

т.е. прямая теорема и теорема, противоположная обратной, равносильны. Эту формулу называют также доказательством от противного: вместо доказательства теоремы «Если число делится на 49, то оно делится на 7» можно доказывать теорему «Если число не делится на 7, то оно не делится на 49».

НЕКОТОРЫЕ ТОЖДЕСТВЕННО ИСТИННЫЕ ФОРМУЛЫ

1° $\overline{\overline{\alpha}} = \alpha$.

2° $(\alpha \vee \alpha) \Longleftrightarrow \alpha \Longleftrightarrow (\alpha \wedge \alpha)$.

3° $(\alpha \vee \beta) \Longleftrightarrow (\beta \vee \alpha)$, $(\alpha \wedge \beta) \Longleftrightarrow (\beta \wedge \alpha)$.

4° $((\alpha \vee \beta) \vee \gamma) \Longleftrightarrow (\alpha \vee (\beta \vee \gamma))$, $((\alpha \wedge \beta) \wedge \gamma) \Longleftrightarrow (\alpha \wedge (\beta \wedge \gamma))$.

5° $\overline{\alpha \vee \beta} \Longleftrightarrow \overline{\alpha} \wedge \overline{\beta}$, $\overline{\alpha \wedge \beta} \Longleftrightarrow \overline{\alpha} \vee \overline{\beta}$.

6° $\alpha \vee \overline{\alpha} = 1$, $\alpha \wedge \overline{\alpha} = 0$.

7° $(\alpha \vee \beta) \wedge \gamma = (\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$, $(\alpha \wedge \beta) \vee \gamma = (\alpha \vee \gamma) \wedge (\beta \vee \gamma)$.

8° $(\alpha \Longrightarrow \beta) \Longleftrightarrow (\overline{\alpha} \vee \beta)$.

Высказывание может содержать переменные, которые называются ***предметными***. Такое высказывание называют ***предикатом*** и обозначают $R(x)$, $Q(x, y)$,... , где x, y – предметные переменные. По поводу предиката можно сказать, будет ли он истинным для любых значений предметных переменных, или только для некоторых их значений. Это можно записать с помощью специальных символов, которые называются ***кванторами***:

$\forall$ – «для всех», «для любого»,

$\exists$ – «существует», «найдется»,

$\exists!$ – «существует единственный», «найдется единственный».

Квантор $\forall$ называют квантором ***всеобщности***, а кванторы $\exists$ и $\exists!$ – кванторами ***существования***. Мы будем всегда писать их *перед* высказываниями.

Например, предикат

$$\forall(x)\exists(y)\ \ y = x^2$$

означает, что для любого числа x найдется число y, которое будет квадратом числа x. Это – истинный предикат.

Часто возникает необходимость найти отрицание некоторого предиката. Для этого надо поменять кванторы всеобщности на кванторы существования и наоборот и взять отрицание от утверждения, которое стоит после последнего

квантора. Например, для предыдущего предиката отрицанием буде такой предикат

$$\overline{\forall(x)\exists\,(y)\ y = x^2} \Longleftrightarrow \exists\,(x)\,\forall\,(y)\ \ y \neq x^2.$$

Он означает, что существует такое число x, квадрат которого нельзя выразить никаким числом y. Этот предикат не является истинным.

В Приложении[1)]показано применение системы *Mathematica* для решения некоторых задач математической логики.

Отметим, что методы математической логики используются, например, в теории переключательных схем[2)].

2 Множества

Понятие множества относится к первичным понятиям в математике. Поэтому, множеству нельзя дать какого-либо определения, так же, как, например, понятию точки или прямой в геометрии. Чтобы как-то описать, о чем все же идет речь, говорят, что множество — это совокупность некоторых объектов, которые называются элементами множества. Однако такое описание не может считаться определением, так как совокупность — это просто другое название множества.

Запись $a \in A$ означает, что элемент a принадлежит множеству A, в противном случае пишут $a \notin A$. Например, начало координат принадлежит координатной оси, а точка $(1;1)$ ей не принадлежит. Множество, которому не принадлежит ни один элемент, называется ***пустым*** и обозначается символом $\emptyset$. Например, пустым является множество треугольников на плоскости с суммой внутренних углов, равной 190°. Обычно в конкретной теории, или конкретной задаче любой элемент любого множества всегда принадлежит еще одному множеству, которое называют ***универсальным*** и обозначают U. Например, в планиметрии универсальным множеством является плоскость, на которой рассматриваются все геометрические фигуры.

Множество можно задать перечислением всех входящих в него элементов: $A = \{-1, \pi, e, j\}$. А если элементов очень много или даже бесконечно много? Тогда множество задают либо незавершенным перечислением его элементов: $B = \{2, 4, 8, 16, \ldots\}$, из которого легко догадаться, какие еще элементы принадлежат множеству; либо описанием структуры его элементов и их свойств: $D = \{(x, y) : x^2 + y^2 = 1, y \geq 0\}$ (полукруг, расположенный в верхней полуплоскости координатной плоскости).

2.1 Операции над множествами

Далее будем использовать обозначение $\stackrel{\triangle}{=}$, которое называется «равно по определению» и поясняет, что величина слева от этого знака определяет величину, которая стоит справа от него. Например, можно написать

$$a^n \stackrel{\triangle}{=} \underbrace{a \cdot \ldots \cdot a}_{n}, \; n \geq 2.$$

Множество $\overline{A}$ называется ***дополнением*** множества A, если $\overline{A}$ состоит из элементов, которые не принадлежат A: $\overline{A} \stackrel{\triangle}{=} \{x : x \notin A\} = U \setminus A$. Например, на множестве натуральных чисел дополнением четных чисел является множество чисел нечетных. Операции над множествами обычно иллюстрируются с помощью так называемых диаграмм Венна (см. рис. 1, на котором результат выполнения операций над множествами показан красным цветом). Дополнение множества представлено на рис. 1, а).

Множество $A \cup B$ называется ***объединением*** множеств A и B, если оно состоит из элементов, которые принадлежат или множеству A, или множеству B (операция «или»): $A \cup B \stackrel{\triangle}{=} \{x : x \in A \vee x \in B\}$ (рис. 1, б)). Примером объединения двух множеств может служить отрезок $[1; 4]$, который является объединением отрезков $[1; 3]$ и $[2; 4]$.

Множество $A \cap B$ называется ***пересечением*** множеств A и B, если оно состоит из элементов, которые принадлежат и множеству A, и множеству B (операция «и»): $A \cap B \stackrel{\triangle}{=} \{x : x \in A \wedge x \in B\}$ (рис. 1, в)). Например, множество натуральных чисел, которые делятся на 6, является пересечением множеств натуральных чисел, делящихся и на 2, и на 3.

Множество $A - B = A \setminus B \stackrel{\triangle}{=} A \cap \overline{B}$ называется ***разностью*** множеств A и B, оно состоит из элементов, которые принадлежат множеству A, но не принадлежат множеству B (рис. 1, г)). Например, показатель n корня $\sqrt[n]{}$ принадлежит множеству $\mathbb{N} \setminus \{1\}$.

СВОЙСТВА ОПЕРАЦИЙ НАД МНОЖЕСТВАМИ

1° $\overline{\emptyset} = U, \overline{U} = \emptyset, \overline{\overline{A}} = A.$

2° $\overline{A \cup B} = \overline{A} \cap \overline{B}, \overline{A \cap B} = \overline{A} \cup \overline{B}.$

3° $A \cup \emptyset = A, A \cap \emptyset = \emptyset.$

4° $A \cup U = U, A \cap U = A.$

5° $A \cup \overline{A} = U, A \cap \overline{A} = \emptyset.$

6° $A \cup B = B \cup A$, $A \cap B = B \cap A$.

7° $(A \cup B) \cup C = A \cup (B \cup C)$, $(A \cap B) \cap C =. A \cap (B \cap C)$.

8° $(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$, $(A \cap B) \cup C = (A \cup C) \cap (B \cup C)$.

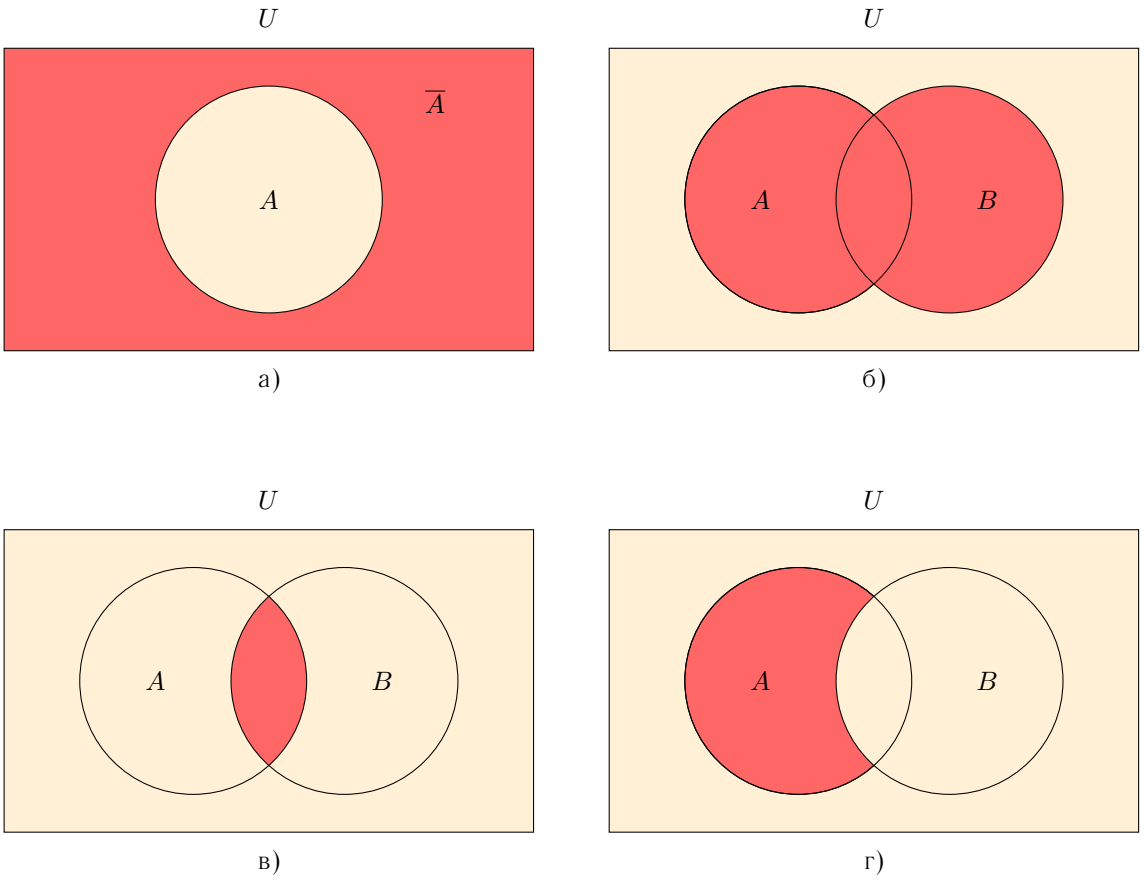


Рис. 1. Операции над множествами: а) дополнение, б) объединение, в) пересечение, г) разность.

2.2 Отношения между множествами

Множество A является ***подмножеством*** множества B, или множество A ***включено*** в множество B, и пишут $A \subset B$, если элемент A является элементом множества B:

$$A \subset B \iff \forall\,(x)\ \ x \in A \Longrightarrow x \in B. \tag{2}$$

Пример 1. *Что означает* $\overline{A \subset B}$ *?*

Решение. Надо вспомнить, что $(\alpha \Longrightarrow \beta) \iff (\overline{\alpha} \vee \beta)$ и $\overline{\alpha \vee \beta} \iff \overline{\alpha} \wedge \overline{\beta}$, а затем взять отрицание от предиката в правой части эквивалентности (2):

$$\overline{A \subset B} \iff \exists(x)\ x \in A \wedge x \notin B.$$

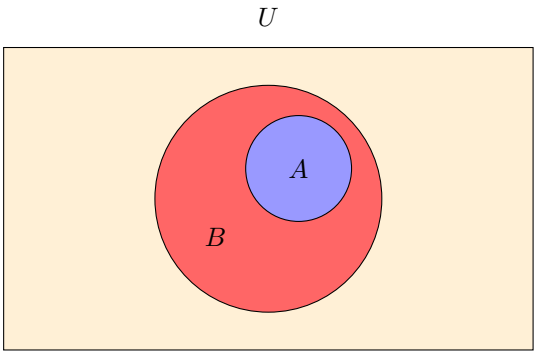


Рис. 2. Подмножество множества.

Таким образом, $\overline{A \subset B}$ означает, что существует элемент множества A, не принадлежащий множеству B. □

Говорят, что множество A ***равно*** множеству B и пишут $A = B$, если A включено в B и B включено в A:

$$A = B \Longleftrightarrow (A \subset B) \wedge (B \subset A)\,.$$

СВОЙСТВА ОТНОШЕНИЙ МЕЖДУ МНОЖЕСТВАМИ

1° $\emptyset \subset A \subset U$.

2° $((A \subset B) \wedge (B \subset C)) \Longrightarrow A \subset C$.

3° $((A = B) \wedge (B = C)) \Longrightarrow A = C$.

В теории множеств теоремы, в частности, свойства множеств доказываются двумя способами. Рассмотрим эти способы на примере доказательства такой теоремы.

Теорема 1. *Доказать, что для любых двух множеств A и B справедливо равенство $\overline{A \setminus B} = \overline{A} \cup B$.*

Доказательство. Первый способ заключается в использовании определения операции разности множеств и свойств операций над множествами:

$$\overline{A \setminus B} = \overline{A \cap \overline{B}} = \overline{A} \cup \overline{\overline{B}} = \overline{A} \cup B.$$

Второй способ опирается на тот факт, что доказательство теоремы есть доказательством равенства двух множеств, а разность множеств по определению можно доказать как то, что первое множество включено во второе и наоборот.

Возьмем произвольный элемент a из левой части доказываемого равенства. Тогда справедлива цепочка следствий

$$\Big(a \in \overline{A \backslash B}\Big) \Longrightarrow (a \notin A \backslash B) \Longrightarrow \big(a \notin A \cap \overline{B}\big) \Longrightarrow \big(a \notin A \vee a \notin \overline{B}\big) \Longrightarrow$$
$$\Longrightarrow \big(a \in \overline{A} \vee a \in B\big) \Longrightarrow a \in \overline{A} \cup B,$$

из которой получаем, что a принадлежит и правой части формулы. Точно так же оказывается справедливой и обратная цепочка:

$$a \in \overline{A} \cup B \Longrightarrow \big(a \in \overline{A} \vee a \in B\big) \Longrightarrow \big(a \notin A \vee a \notin \overline{B}\big) \Longrightarrow$$
$$\Longrightarrow \big(a \notin A \cap \overline{B}\big) \Longrightarrow (a \notin A \backslash B) \Longrightarrow \Big(a \in \overline{A \backslash B}\Big).$$

Таким образом, левая часть равенства включена в правую, а правая — в левую. По определению отношения равенства двух множеств множества $\overline{A \backslash B}$ и $\overline{A} \cup B$ равны друг другу. □

В Приложении[3)] можно ознакомиться с тем, как система *Mathematica* решает несложные задачи теории множеств.

Приложение

[1)] Свои лекции я решил сопроводить примерами того, как аналитические задания, иногда довольно трудные, решает система *Mathematica*. Во-первых, это дает возможность рассмотреть более сложные, трудоемкие, но и более интересные задачи, а, во-вторых, постоянное использование этой системы приучит студента не бояться сложностей и использовать данную систему и после окончания вуза.

Я не буду подробно останавливаться на том, как устроена система *Mathematica*, в чем состоят особенности ее работы и т.п. Обо всем этом написаны книги, имеются справочник и руководства, встроенные в систему, и сайт http://wolfram.com.

Скажу только, что, вызвав эту систему, вы начинаете работать в ее документе с расширением (`.nb`). Его особенность заключается в том, что он состоит из ячеек, из которых мы будем использовать в основном ячейки ввода, вывода и графического вывода.

Закончив набор символов в ячейке ввода (*Mathematica* помечает ее как `In` с номером), вы можете нажать `Enter` на дополнительной части клавиатуры или `Shift + Enter` на основной и получить отклик (отсутствие отклика — тоже отклик) в ячейке вывода (помечаемой как `Out` с номером). Выглядит это так:

```
In[1] := 2 + 5 + a a a
```
$$\texttt{Out[1]} := 7 + a^3$$

Как видите, *Mathematica* сама умеет упрощать выражения, в том числе и буквенные (пробелы в «a a a» означают умножение). Собственно, в этом и состоит ее предназначение — выполнять символьные преобразования, включая аналитическое решение уравнений, символьное дифференцирование и интегрирование и многое другое.

Впрочем, вычислять она тоже умеет. Сложим, например, обыкновенные дроби 1/7 и 2/3. В строку ввода можно их ввести и так, как здесь записано, и с помощью двухэтажной конструкции. Шаблон для ввода последней появится, если его выщелкнуть мышкой из палитры инструментов или нажать комбинацию клавиш `Ctrl + /`. Палитры инструментов вызываются из главного меню. Например, при обращении к главному меню вида `Palettes` → `Writing Assistant` появляется мощная палитра, которая в разделе `Typesetting` содержит все необходимые символы. Можно вызвать и палитру попроще: `Palettes` → `Other` → `Basic Math Input`. На первых порах ее вполне достаточно. С использованием шаблона сложение дробей будет выглядеть так:

$$\texttt{In[2]} := \frac{1}{3} + \frac{2}{7}$$
$$\texttt{Out[2]} := \frac{13}{21}$$

Mathematica всегда предпочитает точный ответ приближенному (что следует из ее аналитической предназначенности). А если все же требуется приближенный ответ? Тогда предыдущий ввод следует изменить:

$$\texttt{In[3]} := \frac{1}{3} + \frac{2}{7}\texttt{//N}$$
```
Out[3] := 0.619048
```

Ответ получен со стандартной в этой системе точностью, 6 цифр. Если точность требуется уменьшить или увеличить, применяют оператор `N[expr,n]`, где `expr` — выражение, значение которого требуется вычислить, `n` — количество цифр результата. Вычислим результат сложения дробей приближенно, задав 15 цифр в ответе:

$$\texttt{In[4]} := \texttt{N}\left[\frac{1}{3} + \frac{2}{7}, 15\right]$$

```
Out[4] := 0.619047619047619
```

Далее мы будем опускать обозначения `In` и `Out` входных и выходных ячеек, различая их по цвету.

Поскольку текущая лекция посвящена логике и множествам, посмотрим, что нам может предложить *Mathematica* в этих областях.

Прежде всего необходимо знать, как ввести в поле ввода то или иное отношение между числами и другими математическими объектами. Можно использовать клавиатуру, а можно специальные палитры инструментов. В следующей таблице дана соответствующая информация.

Ввод отношений

Клавиатура	Палитра	Название отношения
==		равно
!=	$\neq$	не равно
>		больше
>=	$\geq$	больше либо равно
<		меньше
<=	$\leq$	меньше либо равно

«Клавиатура» означает, что символы для строки ввода набираются на клавиатуре, а «Палитра» — что символы выщелкиваются мышкой из палитры инструментов. Если пользоваться палитрами, то зачем надо знать комбинации клавиш для набора тех же самых отношений? Дело в том, что *Mathematica* выдавая результаты, может в ответе напечатать не символ из палитры, а соответствующую комбинацию клавиш. Так что для понимания результатов надо знать оба первых столбца таблицы.

Теперь можно задать системе *Mathematica* первые вопросы по поводу отношений между числами:

```
2*2 ≠ 4
False
3 != 2
True
```

`False` означает, что отношение неверно (потому что дважды два все-таки равно четырем), а `True` констатирует, что отношение правильно (три действительно не равно двум).

Следующая табличка показывает, как надо вводить логические операторы. Значок `Esc` означает нажатие клавиши `Esc`.

Ввод логических операторов

Клавиатура	Палитра	Название оператора
!	$\neg$	отрицание
||	$\vee$	дизъюнкция
&&	$\wedge$	конъюнкция
`Esc` => `Esc`		следование
`Esc` equiv `Esc`		эквивалентность

Попробуем проверить истинность высказывания, в котором будет участвовать логический оператор:

```
7 > 4 ∧ 2 ≠ 3
True
```

Для построения таблиц истинности в системе *Mathematica* имеется специальный оператор `BooleanTable`, который по выражению для высказывания создает список значений этого высказывания для всех возможных наборов входящих в него букв. Отметим, что аргументы операторов обычно берутся в квадратные скобки, а списки представляют собой заключенную в фигурные скобки последовательность некоторых объектов, разделенных между собой запятыми. Пустой список (не имеющий элементов), имеет вид {}. Элементом списка может быть другой список.

Построим таблицу истинности для дизъюнкции, выщелкивая греческие буквы из палитры или набирая их названия между двумя нажатиями клавиши `Esc`, например, `Esc alpha Esc`:

```
BooleanTable[α ∨ β]
{True, True, True, False}
```

Список получен, но хотелось бы, чтобы значения высказывания были бы единицами и нулями, как на лекции. Для этого имеется оператор `Boole`:

```
Boole[BooleanTable[α ∨ β]]
{1, 1, 1, 0}
```

Уже лучше, однако вид совсем не табличный. Подскажем, в какой форме мы хотим увидеть результат:

```
Boole[BooleanTable[α ∨ β]]//TableForm
1
1
1
0
```

Здесь `TableForm` — тоже оператор, но в так называемой постфиксной форме, которая не требует квадратных скобок. Полученный результат можно усовершенствовать, так как надо знать, для каких же значений α и β получены значения дизъюнкции. Для этого добавим аргументы в операторе `BooleanTable`, включив α и β в общий список с дизъюнкцией, и присовокупим список переменных, которые мы считаем аргументами дизъюнкции:

```
Boole[BooleanTable[{α,β, α ∨ β},{α,β}]]//TableForm
1 1 1
1 0 1
0 1 1
0 0 0
```

Далее для придания полной информативности результату, неплохо бы озаглавить столбцы. Используем подходящую форму оператора `TableForm`, не постфиксную:

```
TableForm[Boole[BooleanTable[{α,β, α ∨ β},{α,β}]],
    TableHeadings -> {{}, {α,β,α ∨ β}}, TableAlignments -> Center]
```

α	β	$\alpha\|\|\beta$
1	1	1
1	0	1
0	1	1
0	0	0

В операторе `TableForm` опция `TableHeadings` обеспечивает вывод боковика и шапки таблицы. Значение опции задается списком, состоящим из двух подсписков. В первом подсписке указываются элементы боковика; у нас боковика нет, поэтому подсписок пуст. Во втором

подсписке перечисляются элементы шапки таблицы. Обратите внимание, что *Mathematica* в ответе символ $\vee$ заменила на ||.

Опция `TableAlignments` задает выравнивание элементов в столбцах таблицы. У нас выбрано выравнивание по середине столбца (`Center`).

Теперь мы можем получить таблицу истинности (табл. 1) для основных операций математической логики, рассмотренную на лекции:

```
TableForm[Boole[BooleanTable[{α,β,¬α,α∨β,α∧β,α⇒β,α⇔β},{α,β}]],
    TableHeadings -> {{},{α,β,¬α,α∨β,α∧β,α⇒β,α⇔β }},
    TableAlignments -> Center]
```

α	β	$!\alpha$	$\alpha\|\|\beta$	$\alpha\&\&\beta$	$\alpha \Rightarrow \beta$	$\alpha \Leftrightarrow \beta$
1	1	0	1	1	1	1
1	0	0	1	0	0	0
0	1	1	1	0	1	0
0	0	1	0	0	1	1

Для того чтобы проверить, не является ли некоторое высказывание тавтологией, не обязательно составлять для него таблицу истинности. Для такой проверки *Mathematica* имеет специальный оператор `TautologyQ`, который выдает в ответе `True`, если высказывание является тавтологией и выдает `False` в противном случае. Проверим, например, две тавтологии, рассмотренные на лекции:

```
(α⇔β)⇔((α⇒β)∧(β⇒α))//TautologyQ
True
(α⇒β)⇔(¬β⇒¬α)//TautologyQ
True
```

Для работы с предикатами необходимы кванторы. Система *Mathematica* предоставляет квантор всеобщности в виде оператора `ForAll`, который можно с помощью палитры инструментов набрать как $\forall$. Имеется и квантор существования в виде оператора `Exists` или символа палитры $\exists$. Истинность предикатов проверяет оператор `Resolve`.

Зададим, например, предикат, рассмотренный на лекции:

```
∀x∃y y == x^2
```
$$\forall_x \exists_y\, y == x^2$$

Чтобы ввести нижний индекс надо воспользоваться сочетанием клавиш `Ctrl + _`, а верхний — сочетанием клавиш `Ctrl + ^`. Проверим истинность заданного предиката:

```
Resolve[%]
True
```

Знак процента означает, что аргументом оператора является результат предыдущего действия.

Возьмем отрицание от заданного оператора:

```
¬∀x∃y y == x^2
```
$$\neg\forall_x \exists_y\, y == x^2$$
$$\exists_x \forall_y\, y \neq x^2$$

и проверим его истинность:

```
Resolve[%]
False
```

[2] Переключательной схемой будем называть схематическое изображение электрической цепи, состоящей из переключателей, соединенных проводниками, и источника питания.

Пусть каждый переключатель имеет только два состояния: замкнут и разомкнут, причем, в первом состоянии он проводит ток, а во втором — нет. Каждому переключателю X_i, $i = \overline{1,n}$ поставим в соответствие высказывание x_i = «Переключатель X_i замкнут», которое в теории переключательных схем называется логической переменной. Рассмотрим также высказывание F = «Цепь проводит ток», называемое функцией логических переменных.

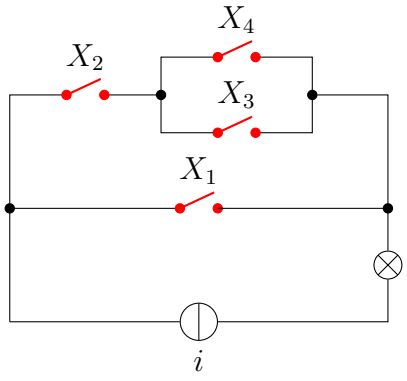
Очевидно, для каждой комбинации включенных и выключенных переключателей мы сможем сказать, проводит цепь ток или не проводит. Чтобы описать работу переключательной схемы с помощью средств математической логики, мы должны каждому ноль-единичному набору значений логических переменных $x_1, \ldots, x_n$, описывающему замкнутость и разомкнутость переключателей, поставить в соответствие значение 1 функции F, если при этом ток в цепи идет, и 0, если ток не идет. Оказывается, это всегда можно сделать, если подходящим образом соединить логические переменные между собой с помощью всего трех логических операций: отрицания, дизъюнкции и конъюнкции. Правда, в общем случае сделать это можно не единственным способом.

Наоборот, для любого высказывания, записанного с помощью логических переменных и связывающих их логических операций, можно составить переключательную схему, которая будет работать в соответствии с логикой этого высказывания.

Для примера на анимационном рис. 3 показана переключательная схема, в которую добавлена электрическая лампочка, наглядно показывающая, когда в цепи идет ток. Очевидно, ток будет идти, если включен переключатель x_1, или переключатели x_2 и x_3, или x_2 и x_4. Работа переключательной схемы описывается функцией

$$F = x_1 \vee (x_2 \wedge (x_3 \vee x_4)).$$

Щелкая на рис. мышкой убедитесь, что каждому варианту замыкания и размыкания переключателей соответствует правильный набор значений логических переменных, зажженной лампочке отвечает значение 1 функции F, а незажженной — 0. Значение этой функции можно вычислить, пользуясь таблицей истинности 1.



x_1	x_2	x_3	x_4	F
0	0	0	0	0

Рис. 3. Переключательная схема.

[3)] Возможности системы *Mathematica* в области теории множеств ограничиваются конечными множествами. Множества представляются списками, но не всякий список может считаться множеством. Например, в списке могут быть одинаковые элементы, а во множестве — нет. Пусть имеется список

```
A = {a, b, c, a, 1, 2, 3, 2, 4};
```

Точка с запятой после оператора означает, что *Mathematica* не должна выдавать на дисплей результат выполнения оператора. Кроме того, в обращении к системе *Mathematica* имеется

конструкция «`A =`», которая дает списку имя `A`. Вообще практически любой объект в системе *Mathematica* можно поименовать, чтобы в дальнейшем вместо объекта использовать его имя.

Так вот, теперь у нас есть список `A`, причем, множеством он не является, так как имеет одинаковые элементы. Чтобы список сделать множеством, применим к нему оператор `Union`, который вообще-то выполняет объединение множеств, но, в частности, превращает список в множество, удаляя из списка элементы-дубликаты и выполняя сортировку результата:

```
A = Union[A]
{1, 2, 3, 4, a, b, c}
```

Обратите внимание, что имя `A` теперь присвоено полученному множеству.

Зададим еще одно множество:

```
B = {2, 4, 5, d, a};
```

Найдем объединение этих множеств, но, чтобы не набирать `Union`, выщелкнем из палитры символ $\cup$:

$A \cup B$

```
{1, 2, 3, 4, 5, a, b, c, d}
```

Точно так же находится пересечение множеств:

$A \cap B$

```
{2, 4, a}
```

Для получения разности множеств $A - B$ применим оператор `Complement`:

```
Complement[A, B]
{1, 3, b, c}
```

А как быть с дополнением множества? Очень просто, ведь $\overline{A} = U - A$. Так что надо только задать какое-нибудь универсальное множество U, в которое входит A, и найти их разность, чтобы получить дополнение множества A.

Литература

[1] Бугров Я.С., Никольский С.М. *Высшая математика. Дифференциальное и интегральное исчисление.* — М.: Наука, 1984, — с. 9-12.