ЧАСТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕДЛЬНОЕ УЧРЕЖДЕНИЕ «СЕВЕРО-КАВКАЗСКИЙ КОЛЛЕДЖ ИННОВАЦИОННЫХ ТЕХНОЛОГИЙ»

ЛЕКЦИЯ 1. ТЕМА 1. ОСНОВЫ ТЕОРИИ МНОЖЕСТВ

1. Основные понятия и определения теории множеств

Теория множеств представляет собой фундаментальный раздел математики, формализующий понятие совокупности объектов и отношения между ними. В контексте дискретной математики и IT-технологий теория множеств служит базой для моделирования структур данных, описания семантики языков программирования, построения реляционных баз данных и формальных спецификаций.

Множество — это совокупность чётко определённых, различимых объектов, рассматриваемых как единое целое. Объекты, составляющие множество, называются его **элементами**. Принадлежность элемента a множеству A обозначается как $a \in A$; непринадлежность — как $a \notin A$.

Множества принято обозначать заглавными латинскими буквами ($A, B, C, \ldots$), элементы — строчными ($a, b, c, \ldots$).

Способы задания множеств:

- Перечислением элементов: $A = \{a_1, a_2, \ldots, a_n\}$. Применимо для конечных множеств.
- Указанием характеристического свойства: $A = \{x \mid P(x)\}$, где $P(x)$ — предикат, определяющий принадлежность элемента множеству.

Ключевые понятия:

- **Пустое множество** $\varnothing$ — множество, не содержащее элементов. Обладает свойством $\forall A$: $\varnothing \subseteq A$.
- **Универсальное множество** U — множество, содержащее все объекты, рассматриваемые в рамках конкретной задачи.
- **Конечное множество** — множество, число элементов которого конечно. **Бесконечное множество** — в противном случае.
- **Мощность множества** $|A|$ — количество элементов в конечном множестве; для бесконечных множеств используется понятие кардинального числа.
- **Подмножество**: множество B является подмножеством A ($B \subseteq A$), если каждый элемент B принадлежит A. Если $B \subseteq A$ и $B \neq A$, то B — собственное подмножество A ($B \subset A$).
- **Булеан** $\mathcal{P}(A)$ — множество всех подмножеств A. Мощность булеана конечного множества из n элементов равна 2^n.

Примеры:

- $A = \{1, 2, 3\}$ — конечное множество.
- $\mathbb{N} = \{1, 2, 3, \ldots\}$ — множество натуральных чисел (бесконечное).
- $B = \{x \in \mathbb{Z} \mid -5 \leq x \leq 5\}$ — множество целых чисел от -5 до 5.

2. Операции над множествами и их свойства

Операции над множествами позволяют конструировать новые множества на основе существующих.

Основные операции:

- **Объединение** $A \cup B = \{x \mid x \in A \vee x \in B\}$.
- **Пересечение** $A \cap B = \{x \mid x \in A \wedge x \in B\}$.
- **Разность** $A \setminus B = \{x \mid x \in A \wedge x \notin B\}$.
- **Симметрическая разность** $A \triangle B = (A \setminus B) \cup (B \setminus A)$.
- **Дополнение** $\overline{A} = U \setminus A$ (относительно универсального множества U).

Свойства операций:

- Коммутативность: $A \cup B = B \cup A$, $A \cap B = B \cap A$.
- Ассоциативность: $(A \cup B) \cup C = A \cup (B \cup C)$, аналогично для пересечения.
- Дистрибутивность: $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$; $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.
- Законы де Моргана: $\overline{A \cup B} = \overline{A} \cap \overline{B}$, $\overline{A \cap B} = \overline{A} \cup \overline{B}$.
- Идемпотентность: $A \cup A = A$, $A \cap A = A$.
- Свойства пустого и универсального множеств: $A \cup \varnothing = A$, $A \cap U = A$, $A \cap \varnothing = \varnothing$, $A \cup U = U$.
- Закон двойного дополнения: $\overline{\overline{A}} = A$.

Пример:

Пусть $A = \{1, 2, 3\}$, $B = \{2, 4\}$.

Тогда:

$A \cup B = \{1, 2, 3, 4\}$,

$A \cap B = \{2\}$,

$A \setminus B = \{1, 3\}$,

$B \setminus A = \{4\}$,

$A \triangle B = \{1, 3, 4\}$.

Алгоритм выполнения операций над множествами:

1. Определить множества A и B и требуемую операцию.
2. Для объединения: собрать все уникальные элементы из A и B.
3. Для пересечения: отобрать элементы, присутствующие в обоих множествах.
4. Для разности $A \setminus B$: отобрать элементы A, отсутствующие в B.
5. Для симметрической разности: объединить разности $A \setminus B$ и $B \setminus A$.
6. Для дополнения: отобрать все элементы универсального множества U, не входящие в A.
7. Записать результат в виде множества.

3. Декартово произведение и степень множества

Декартово произведение множеств A и B определяется как множество всех упорядоченных пар (a, b), где $a \in A$, $b \in B$:

$$A \times B = \{(a, b) \mid a \in A, b \in B\}.$$

Для n множеств декартово произведение:

$$A_1 \times A_2 \times \ldots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A_i\}.$$

Степень множества A^n — это декартово произведение множества A на себя n раз:

$$A^n = A \times A \times \ldots \times A \quad (n \text{ раз}).$$

Пример:
Если $A = \{0, 1\}$, то $A^2 = \{(0, 0), (0, 1), (1, 0), (1, 1)\}$.
Это соответствует множеству всех двоичных векторов длины 2, что широко применяется в теории кодирования и булевой алгебре.

Применение в IT:

- Реляционные базы данных: кортежи в таблицах представляют собой элементы декартова произведения доменов атрибутов.
- Теория автоматов: состояния и переходы часто моделируются через декартовы произведения множеств.
- Криптография и кодирование: векторы над конечными полями — элементы степеней множеств.

Алгоритм построения декартова произведения:

1. Зафиксировать множества A и B.
2. Инициализировать пустое множество C.
3. Для каждого элемента $a \in A$:
    - Для каждого элемента $b \in B$:
        - Добавить пару (a, b) в C.
4. Вернуть множество C как результат $A \times B$.

4. Отношения в множествах

Отношение между множествами — это подмножество декартова произведения этих множеств.
Бинарное отношение R на множествах A и B: $R \subseteq A \times B$.
Если $A = B$, то отношение называется отношением на множестве A.

Способы задания отношений:

- Перечислением пар.
- Характеристическим свойством: $R = \{(a, b) \mid P(a, b)\}$.
- Матрицей смежности (для конечных множеств).
- Графом (ориентированным, если отношение несимметрично).

Типы отношений:

- **Рефлексивное**: $\forall a \in A : (a, a) \in R$.
- **Антирефлексивное**: $\forall a \in A : (a, a) \notin R$.
- **Симметричное**: $(a, b) \in R \Rightarrow (b, a) \in R$.
- **Антисимметричное**: $(a, b) \in R$ и $(b, a) \in R \Rightarrow a = b$.
- **Транзитивное**: $(a, b) \in R$ и $(b, c) \in R \Rightarrow (a, c) \in R$.

Важные классы отношений:

- **Отношение эквивалентности** — рефлексивное, симметричное и транзитивное. Разбивает множество на классы эквивалентности. Пример: отношение равенства, отношение сравнимости по модулю.
- **Частичный порядок** — рефлексивное, антисимметричное и транзитивное отношение. Пример: отношение включения множеств $\subseteq$, отношение делимости на натуральных числах.
- **Линейный порядок** — частичный порядок, в котором любые два элемента сравнимы.

Пример:
На множестве $A = \{1, 2, 3\}$ отношение $R = \{(1, 1), (2, 2), (3, 3), (1, 2), (2, 3), (1, 3)\}$ является отношением частичного порядка (транзитивное замыкание рефлексивного отношения).

Алгоритм проверки свойств отношения:

1. Задать отношение $R \subseteq A \times A$ (например, списком пар).
2. Проверить рефлексивность: убедиться, что $\forall a \in A : (a, a) \in R$.
3. Проверить симметричность: для каждой пары (a, b) проверить наличие (b, a).
4. Проверить антисимметричность: если $(a, b) \in R$ и $(b, a) \in R$, то должно быть $a = b$.
5. Проверить транзитивность: для всех пар $(a, b), (b, c) \in R$ проверить наличие (a, c).
6. На основании результатов классифицировать отношение (эквивалентность, частичный/линейный порядок и т.д.).

Алгоритм определения принадлежности элемента множеству:

1. Идентифицировать способ задания множества (перечисление или характеристическое свойство).
2. Если задано перечислением: проверить наличие элемента в списке.
3. Если задано характеристическим свойством: подставить значение элемента в предикат $P(x)$.
4. Оценить истинность $P(a)$: если истинно, то $a \in A$; иначе $a \notin A$.
5. Зафиксировать результат.

Применение в информатике:

- Моделирование связей в базах данных (один-ко-многим, многие-ко-многим).
- Построение иерархий (наследование в ООП, файловая система).
- Алгоритмы сортировки и поиска (линейные и частичные порядки).
- Формальные спецификации и верификация программ.