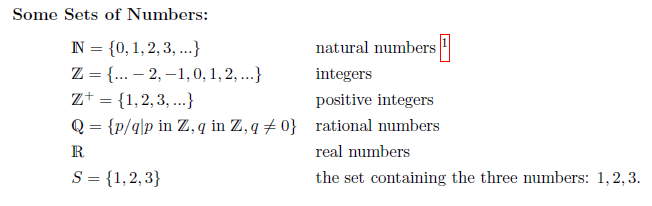
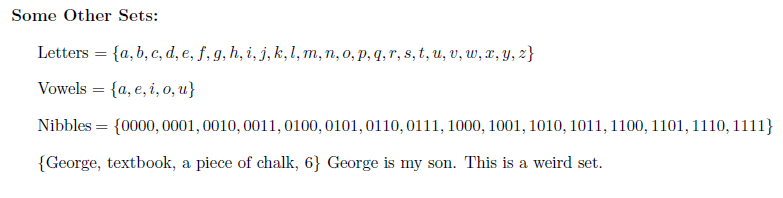
# Chapter 7 Notes: Sets

**Sets:** an unordered collection of objects or **elements.**



We use ***braces (also called curly brackets) to show the elements of a set***. The elements of a set do not have to be numbers.



## Set Basics

If x is an element of S, we write **x ∈ S** which may also be read as “***x is in S***."

If ***x is not in S***, we write, **x ∉ S**.

Two sets are **equal** if and **only if they have the same elements**.

{a, b, c} = {b, c, a} = {c, b, a}

If a set has a **finite number of elements**, we say it is a **finite set** and the **cardinality or size** of the set is the **number of elements it contains**.

**|S|** to denote the cardinality of **S**.

If S = {1, 2, 5, 7}, then |S| = 4

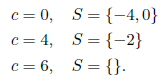
**Cardinality** is also defined for **infinite sets**, i.e. sets with **infinitely many elements**.

Though and both R and Z have infinitely many elements, they do not have the same cardinality. In fact, there are an infinite number of different infinities, based on **Theory of Computation**.

A **set** may also have **no elements** at all, known as **empty set**, denotes by ∅ or {}

When you declare a set variable in computing, it is an empty set until you put something in it.

Variable declaration: S = x is element of c^2 + 4x + c = 0



## Set Builder Notation

We define a set by listing all the elements of the set inside curly brackets:

Finite set example: S = {1, 2, 3}

Infinite set example: Z+ = {1, 3, 5, 20….} or Z = {-2, -1, 0, 1, 2…}

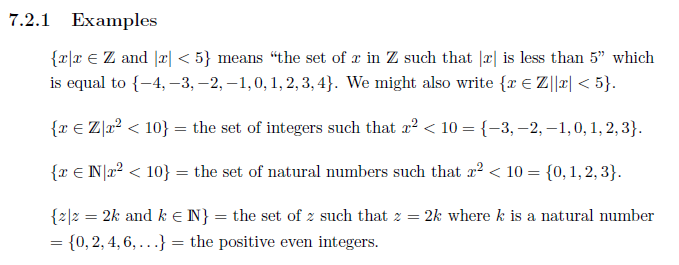
When we need to ***describe which elements are in the set*** instead of just leaving it to the reader to figure out how to continue a list, we use **set-builder notation**.

A set description in set-builder notation looks like: Set description and set builder notation.

Where v is a variable and S is a set. The braces { and } tell us to say “the set of" and the vertical bar `|' is read as “such that." We sometimes use a colon “:" in place of the vertical bar.

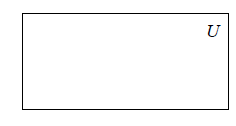
{v| condition on v} is read as “the set of v such that" the condition on v holds.

{v ∈ S| condition on v} is read as “the set of v in S such that" the condition on v holds.

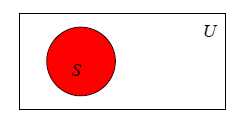


## Venn diagram

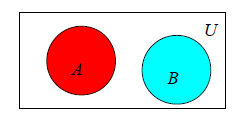
**Venn Diagrams** are a way of **using pictures to describe sets and set operations**. The first thing we do is draw **a universe or universal set, U.** The set **U contains all the objects that we might want to be in the sets.** For example, ***U might be all real numbers*** or ***all people living today*** or all current Northeastern University students. We use a rectangle to show U.



We then show a single set S with elements in the universe U:



If two sets A and B with elements from the universe U have no elements in common, we say A and B are **disjoint**. Here is a **Venn diagram** showing the relationship between disjoint sets A and B.

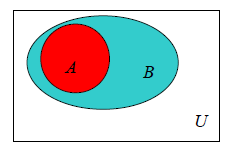


We say **A is a subset of B** (or **A is included in B** or **B includes A**) if **every element of A is also an element of B.**

**A ⊆ B** if **A is a subset of B**, In this case, **A might be equal to B**.

**A ⊂ B** if **A is a subset of B** but **A is not equal to B**.

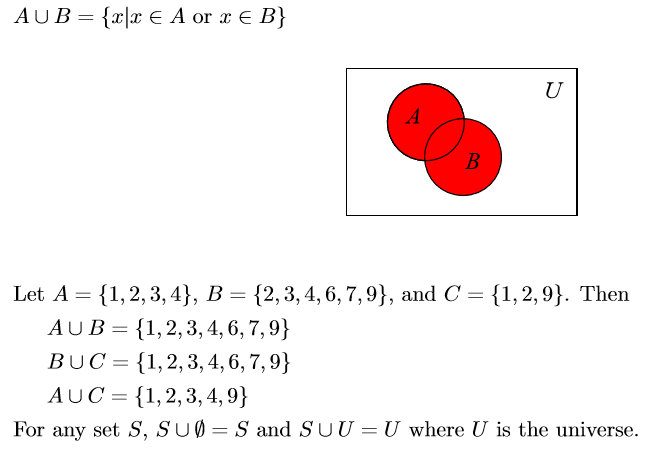
This means that **every element of A is an element of B** but t**here is at least one element of B that is not an element of A**. We say **A is a proper subset of B**.



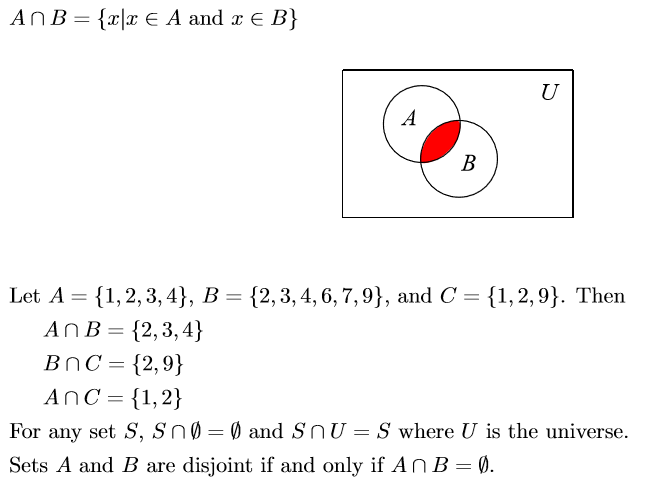
## Set Operations

We use **set operations** to combine sets to form new sets.

### Union **∪**

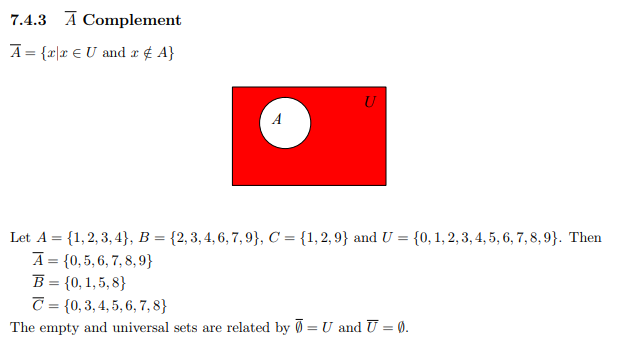


### Intersection ⋂

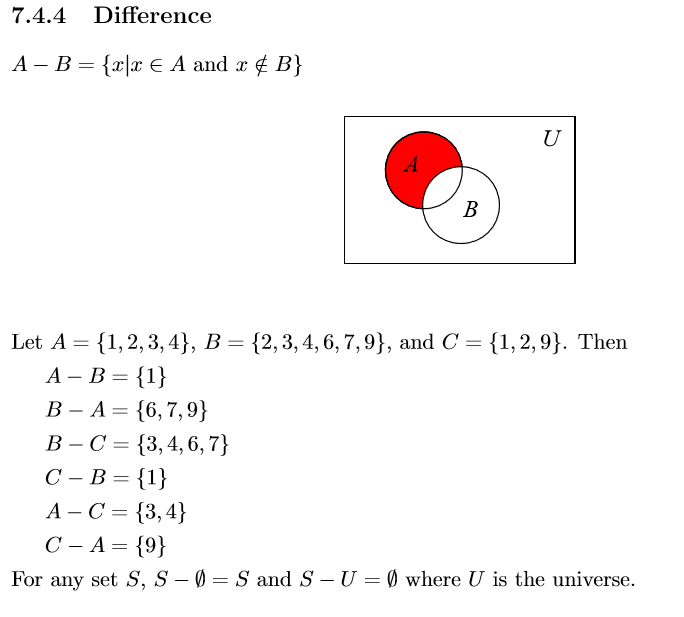
  
If A and B are finite sets, the cardinality of A ∪ B, is given by |A ∪ B| = |A| + |B| − |A ∩ B|.

This is the **Principle of Inclusion-Exclusion**. When we add up |A| + |B|, we have counted all the elements of A and all the elements of B but we have counted the elements in A ∩ B twice so we must subtract that number to get the correct result.

## Complement ∁

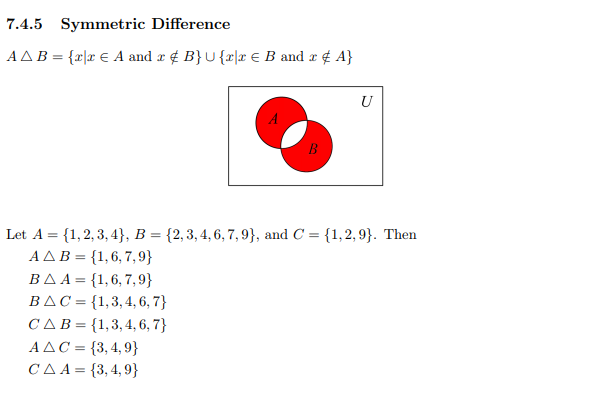


### Difference

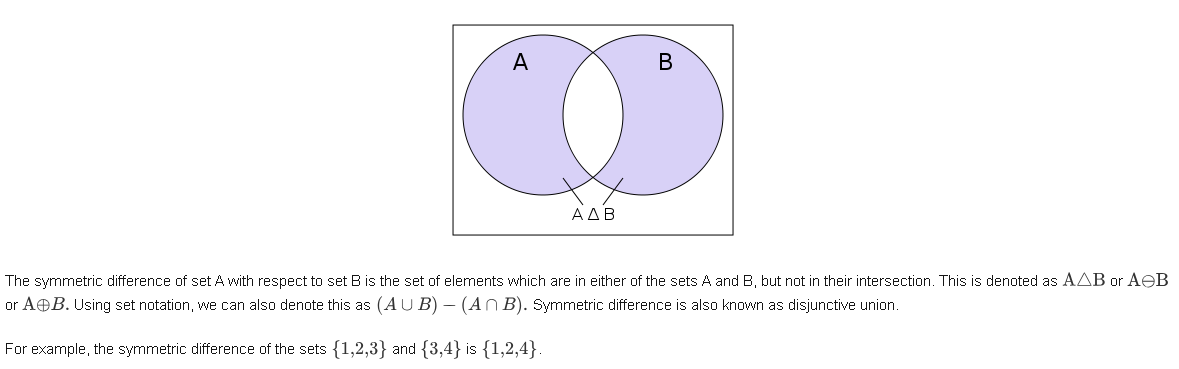


What is in set A but not in set B?

### Symmetric Difference



If S = {x}, P(S) = {∅, {x}} and P(P(S)) = {∅, {∅}, {{x}}, {∅, {x}}}. Note that |S| = 1, |P(S)| = 2 = 2|S| , and |P(P(S))| = 4 = 2|P(S)| .



## Power Sets

If A is a set, the power set P(A) is the set of all subsets of A.

If A = {1, 2} then P(A) = {∅, {1}, {2}, {1, 2}}. The elements of P(A) are sets, not numbers. In general, the cardinality of the power set |P(A)| = 2|A| and sometimes we use 2A instead of P(A) to denote the power set of A.

Example:

For a set S={a,b,c,d} let us calculate the subsets −

Subsets with 0 elements − {∅} (the empty set)

Subsets with 1 element − **{a},{b},{c},{d}**

Subsets with 2 elements − **{a,b},{a,c},{a,d},{b,c},{b,d},{c,d}**

Subsets with 3 elements − **{a,b,c},{a,b,d},{a,c,d},{b,c,d}**

Subsets with 4 elements − **{a,b,c,d}**

Hence, P(S)={{∅},{a},{b},{c},{d},{a,b},{a,c},{a,d},{b,c},{b,d},{c,d},{a,b,c},{a,b,d},{a,c,d},{b,c,d},{a,b,c,d}}

|P(S)|=24=16

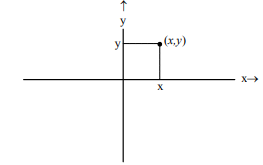
Note − The **power set** of an **empty set** is also an **empty set**.

|P({∅})|=20=1

## Cartesian Product

**Cartesian product** is a mathematical operation that returns a set (or product set or simply product) from multiple sets.

R × R = {(x, y)|x, y ∈ R} and it is usually visualized as a plane.

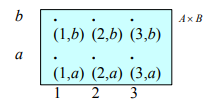


Points correspond to **order pairs** (x, y). Unlike sets, order matters when ***we write an ordered pair***. The **ordered pairs (1, 2) and (2, 1) are not equal** whereas **the sets {1, 2} and {2, 1} are equal**. The Cartesian product is named after Rene Descartes.

We can define the Cartesian product of any two sets, A and B, in a similar way.

**A × B = {(x, y)|x ∈ A and y ∈ B}**

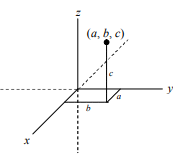
Let A = {1, 2, 3} and B = {a, b}, then **A × B = {(1, a),(1, b),(2, a),(2, b),(3, a),(3, b)}.** We can visualize A × B similarly to the way we visualized R × R.



In general, the cardinality of A × B = |A × B| = |A| × |B|.

We often need the Cartesian product of many sets, e.g. A × B × C. The elements of the Cartesian product A × B × C are similar to ordered pairs but they have three components instead of two, e.g. (a, b, c). As with ordered pairs, the order matters. We call such an ordered **triple a 3-tuple**. **An n-tuple has n components.**

R × R × R = {(x, y, z)|x, y, z ∈ R} is usually used to represent 3-dimensional space



If A = {1, 2}, B = {a, b}, and C = {X, Y } then

A × B × C = {(1, a, X),(1, a, Y ),(1, b, X),(1, b, Y ),(2, a, X),(2, a, Y ),(2, b, X),(2, b, Y )}

In general, the cardinality **|A1 × A2 × · · · × An| = |A1| × |A2| × · · · × |An|**

The Cartesian product A × B is **not commutative**,

A x B ≠ B x A

Cartesian product is **not associative** (unless one of the involved sets is empty).

A x (B x C) ≠ (A x B) x C

## Computer Representation of Sets

Just like numbers, sets can be represented on a computer by 0s and 1s. First, we order the elements of the universe. We use bit-strings whose length is the cardinality of the universe U to represent the subsets of U. Each position in the bit-string corresponds to an element of U. A one in some position means the corresponding element is in the set while a zero means the element is not in the set.

Examples

Let U = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Here are some subsets of U

U = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} 1111111111

210 – 1 = 1023 no empty set

All elements are in U.

S = {1, 2, 5, 6, 7, 9} 0110011101

S = {0, 3, 4, 8} 1001100010

This is the bit-wise complement of the bit-string for S.

A = {2, 5, 6, 7, 9} 0010011101

B = {1, 4, 6, 8, 9} 0100101011

A ∩ B 0000001001

This is the bit-wise and of the bit-strings for A and B.

A ∪ B 0110111111

This is the bit-wise or of the bit-strings for A and B.