Everything on Set Theory

A set represents a collection of items

univ - Universal set	It contains all items present in the model
none - empty set	It contains nothing
Int	Set of integers

A signature introduces a new set → sig fruit{}

An enumeration introduces a new set containing specific items → enum Vegetable{Tomato}

Multiplicities

When we add a set, we can specify how many members it contains by adding a multiplicity constraint

one	Exactly one member
lone	Less or one
some	At least 1 member

If you want to specify a different size, use a fact

Fact FiveFruit{# fruit = 5}

A set can be introduced as a subset of another set. There are 2 ways:

extends	sig Apple, Banana, Pear extends Fruit {}
in	sig Fresh, Expensive in Fruit{}

Extends is used for subsets that are mutually disjoint, no members in common

In includes subsets which may overlap

An abstract signature introduces a set that contains nothing apart from the members of sets that extend that signature

abstract sig Fruit{}

Operations

+ union	Something is in S+T when it's in S or T or both
& intersection	Something is in S&T when its in both S and T
- difference	Something is in S-T when its in S but not T
# number/cardinality	# S is the number of members in S
| such that	{i:Int | i > 5} - Int that is greater than 5

Logical expressions

in → subset	S in T is true if every member of S is also in T
in → membership	A in S is true if A is a member of S
= → equality	S=T is true if S and T have exactly the same members
some → non-emptiness	Some S is true if S has at least one member
no → emptiness	No P is true if p has no members

- General laws
  - A + A = A
  - A & A = A
  - A - A = none
- Commutative laws:
  - A + B = B + A
  - A & B = B & A
- Associative laws:
  - A + (B + C) = (A + B) + C
  - A & (B & C) = (A & B) & C
- Distributive laws:
  - A + (B & C) = (A + B) & (A + C)
  - A & (B + C) = (A & B) + (A & C)

Everything on Using Sigs and Sets

A signature may include fields

A signature may have constraints. These are boolean expressions and may refer directly to the fields. Constraints impose restrictions on the possible models

A predicate is a parameterised boolean expression. It can be run to find the example that makes the predicate true

A fact is a boolean expression that imposes some additional constraint on the specification

An assertion is a boolean expression, expressing some property that we think should follow our specification. Allow can check if there are any counterexamples to this assertion