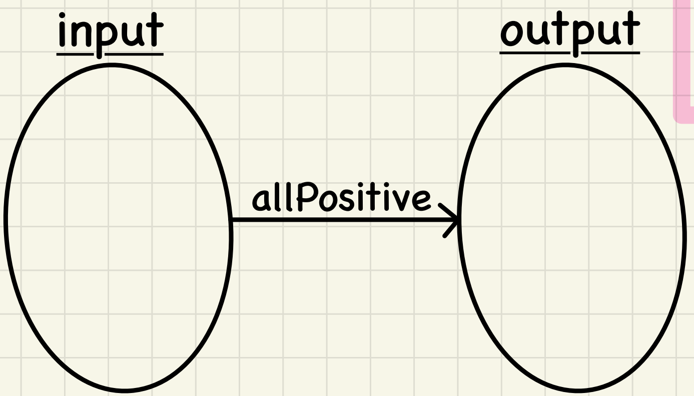
Bidirectional Subset Relations: Programming

```
/* Return the set of positive elements from input. */
HashSet<Integer> allPositive(HashSet<Integer> input)
```

Say:

- S denotes the subset all positive elements from `input`.
- Set `output` denotes the return value from `allPositive`.

Relate the two sets S and output with set operators.



Postcondition checks to see if the output is correct w.r.t. the input.

(R1) $\{x \mid x \in input \land x > 0\} \subseteq output$

(R2) $output \subseteq \{x \mid x \in input \land x > 0\}$

Q1: Why is (R1) alone incomplete?

Q2: Why is (R2) alone incomplete?

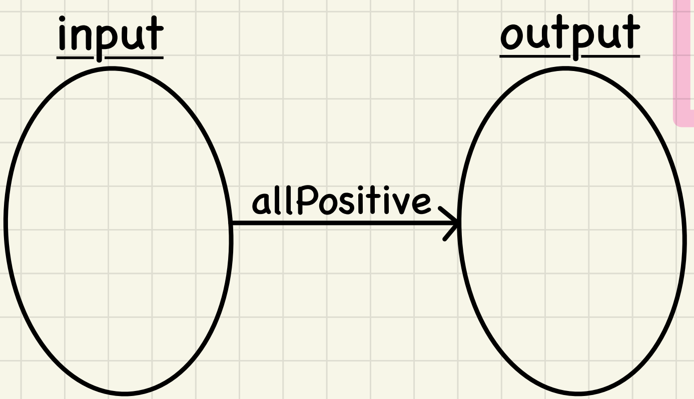
Bidirectional Subset Relations: Programming

```
/* Return the set of positive elements from input. */
HashSet<Integer> allPositive(HashSet<Integer> input)
```

Say:

- S denotes the subset all positive elements from `input`.
- Set `output` denotes the return value from `allPositive`.

Relate the two sets S and output with set operators.



Postcondition checks to see if the output is correct w.r.t. the input.

(R1)

(R2)

Q1: Why is (R1) alone incomplete?

Q2: Why is (R2) alone incomplete?

Set of Tuples

Given n sets $S_1, S_2, \ldots, S_n$, a ***cross/Cartesian product*** of theses sets is a set of n-tuples.

Each ***n-tuple*** $(e_1, e_2, \ldots, e_n)$ contains n elements, each of which a member of the corresponding set.

$$S_1 \times S_2 \times \cdots \times S_n = \{(e_1, e_2, \ldots, e_n) \mid e_i \in S_i \wedge 1 \leq i \leq n\}$$

Example: Calculate {a, b} ✘ {2, 4} ✘ {$, &}