Answers to quiz #1 on Algebraic Specification

Christian Rinderknecht

17 May 2005

1 Arrays

We want an algebraic specification of arrays. An array is a list whose size is fixed at the construction time and whose elements are items of the same type. There is a special item which is used as a default element when creating a new array (i.e., a freshly created array contains these default elements). Then the user can change an element to another value by specifying the integer index (i.e., position) in the array and the new value.
Let us call ITEM the specification of some items whose signature is as follows.

- **Defined types** The type of the items is noted t
- **Constructors**
  DEFAULT : t
  The term DEFAULT is a distinguished item, whose interpretation is left to the user of this specification.

Let us call now ARRAY(ITEM) the specification of the arrays over items of type ITEM.t. The signature is as follows.

- **Defined types** The type of the arrays is noted t.
- **Constructors**
  - EMPTY : t
    The term EMPTY represents the array which contains no element.
  - CREATE : int × int → t
    The term CREATE(x, y) denotes an array whose elements are indexed from x to y, if $x < y$, or from y to x otherwise. The type int denotes the set of positive integers. Integers x and y are called *bounds*. If $x < y$ then x is the *lower bound* and y is the *upper bound*.

    For example, CREATE$(3, 5)$ is an array whose elements are indexed by integers 3, 4 and 5. It contains three elements. These elements are all equal to ITEM.DEFAULT. Also CREATE$(5, 3)$ is

the same as CREATE$(3, 5)$. Therefore, in both cases, the lower bound is 3 and the upper bound is 5.

- SET : $\mathsf{t} \times \mathsf{int} \times \text{ITEM.}\mathsf{t} \to \mathsf{t}$
  The term SET(a, n, e) represents an array equal to array a except that the element at index n is e. If n is out of bounds, i.e., if n is not between the bounds of a, the result of SET is unspecified. Same if a is empty.

- **Functions**

  - LOWER : $\mathsf{t} \to \mathsf{int}$
    The call LOWER(a) is the lower bound of array a, i.e., the smallest valid index. If a is empty, the result is unspecified.
  - UPPER : $\mathsf{t} \to \mathsf{int}$
    The call UPPER(a) is the upper bound of array a, i.e., the greatest valid index. If a is empty, the result is unspecified.
  - GET : $\mathsf{t} \times \mathsf{int} \to \text{ITEM.}\mathsf{t}$
    The call GET(a, n) denotes the element in array a at index n. If n is out of bounds or if a is empty, the result of GET is unspecified.

Question. Complete this signature with equations defining functions LOWER, UPPER and GET.

Answers.

- LOWER : $\mathsf{t} \to \mathsf{int}$
  First, we know that we have to left unspecified the case when the array is empty, i.e. there is no equation whose side is LOWER(EMPTY).

  We still have to consider the two other constructors, CREATE and SET:

$$\text{LOWER}(\text{CREATE}(x, y)) = \textbf{?} \tag{1}$$
$$\text{LOWER}(\text{SET}(a, n, e)) = \textbf{?} \tag{2}$$

  In case of equation 1, we must distinguish two cases: if $x < y$ or $x \geqslant y$:

$$\begin{aligned} \text{LOWER}(\text{CREATE}(x, y)) &= \textbf{?} && \text{if } x < y \\ \text{LOWER}(\text{CREATE}(x, y)) &= \textbf{?} && \text{if } x \geqslant y \\ \text{LOWER}(\text{SET}(a, n, e)) &= \textbf{?} \end{aligned}$$

  These cases are easy to complete because we know explicitly the lower bound:

$$\begin{aligned} \text{LOWER}(\text{CREATE}(x, y)) &= x && \text{if } x < y \\ \text{LOWER}(\text{CREATE}(x, y)) &= y && \text{if } x \geqslant y \\ \text{LOWER}(\text{SET}(a, n, e)) &= \textbf{?} \end{aligned}$$

The last case (equation 2) is also easy to solve because the argument is simply the array a with one element modified: the call to SET does not change the lower bound. Therefore:

$$\begin{aligned} \textsc{Lower}(\textsc{Create}(x, y)) &= x && \text{if } x < y \\ \textsc{Lower}(\textsc{Create}(x, y)) &= y && \text{if } x \geqslant y \\ \textsc{Lower}(\textsc{Set}(a, n, e)) &= \textsc{Lower}(a) \end{aligned}$$

- UPPER : t → int
  This case is the dual of LOWER:

$$\begin{aligned} \textsc{Upper}(\textsc{Create}(x, y)) &= y && \text{if } x < y && (3) \\ \textsc{Upper}(\textsc{Create}(x, y)) &= x && \text{if } x \geqslant y && (4) \\ \textsc{Upper}(\textsc{Set}(a, n, e)) &= \textsc{Upper}(a) && && (5) \end{aligned}$$

- GET : t × int → ITEM.t
  First, we know that the case of the empty array is not specified for GET (there is no element to get), so there is no equation whose side is $\textsc{Get}(\textsc{Empty}, n)$.

  The remaining constructors CREATE and SET have to be considered:

$$\begin{aligned} \textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && (6) \\ \textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} && (7) \end{aligned}$$

  We know from the signature that if the index n is out of bounds, the value of the call to GET is unspecified.

  This means that equation 6 should be restricted to the case

$$\textsc{Lower}(\textsc{Create}(x, y)) \leqslant n \leqslant \textsc{Upper}(\textsc{Create}(x, y))$$

  which can be split into two cases, depending on x and y:

$$\begin{aligned} \textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } x \leqslant n \leqslant y \\ \textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } y \leqslant n \leqslant x \\ \textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} \end{aligned}$$

  We have to restrict also equation 7 to the case

$$\textsc{Lower}(\textsc{Set}(a, p, e)) \leqslant n \leqslant \textsc{Upper}(\textsc{Set}(a, p, e))$$

  which, using the equations 2 and 5 of LOWER and UPPER, is equivalent to $\textsc{Lower}(a) \leqslant n \leqslant \textsc{Upper}(a)$. Therefore we have:

$$\begin{aligned} \textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } x \leqslant n \leqslant y \\ \textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } y \leqslant n \leqslant x \\ \textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} && \text{if } \textsc{Lower}(a) \leqslant n \leqslant \textsc{Upper}(a) \end{aligned}$$

Also p must be in-between bounds: $\textsc{Lower}(a) \leqslant p \leqslant \textsc{Upper}(a)$. So:

$$\begin{aligned}
\textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } x \leqslant n \leqslant y \\
\textsc{Get}(\textsc{Create}(x, y), n) &= \textbf{?} && \text{if } y \leqslant n \leqslant x \\
\textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} && \text{if } \textsc{Lower}(a) \leqslant p, n \leqslant \textsc{Upper}(a)
\end{aligned}$$

The signature tells us that a newly created array is filled with the special element Item.Default. So if we pick any element between the bounds, it is always equal to Item.Default:

$$\begin{aligned}
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } x \leqslant n \leqslant y \\
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } y \leqslant n \leqslant x \\
\textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} && \text{if } \textsc{Lower}(a) \leqslant p, n \leqslant \textsc{Upper}(a)
\end{aligned}$$

The last equation has to be split in two cases, because we can express differently the result if $p = n$ or not:

$$\begin{aligned}
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } x \leqslant n \leqslant y \\
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } y \leqslant n \leqslant x \\
\textsc{Get}(\textsc{Set}(a, n, e), n) &= \textbf{?} && \text{if } \textsc{Lower}(a) \leqslant n \leqslant \textsc{Upper}(a) \\
\textsc{Get}(\textsc{Set}(a, p, e), n) &= \textbf{?} && \text{if } p \neq n \\
& && \text{and } \textsc{Lower}(a) \leqslant p, n \leqslant \textsc{Upper}(a)
\end{aligned}$$

If $p = n$ the result is e because, by construction, it is the element at position n in $\textsc{Set}(a, n, e)$. It $p \neq n$, the result cannot be e, by construction. Hence we must look for it in a. As a conclusion:

$$\begin{aligned}
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } x \leqslant n \leqslant y \\
\textsc{Get}(\textsc{Create}(x, y), n) &= \textsc{Default} && \text{if } y \leqslant n \leqslant x \\
\textsc{Get}(\textsc{Set}(a, n, e), n) &= e && \text{if } \textsc{Lower}(a) \leqslant n \leqslant \textsc{Upper}(a) \\
\textsc{Get}(\textsc{Set}(a, p, e), n) &= \textsc{Get}(a, n) && \text{if } p \neq n \\
& && \text{and } \textsc{Lower}(a) \leqslant p \leqslant \textsc{Upper}(a)
\end{aligned}$$

Notice we removed the condition $\textsc{Lower}(a) \leqslant n \leqslant \textsc{Upper}(a)$ from the last equation. This is a simplication, because, in this case, we can delay the condition check on n to the recursive call $\textsc{Get}(a, n)$. Then, if a is an unmodified array (Create), n will be checked against the bounds (see the first two equations). Otherwise, n may equal an index in bounds (third equation), which settles the problem too.