Techniques for Algorithm Verification: Array Component Assignment Example

Stefan D. Bruda

CS 310, Winter 2025

In-place swap

- Swapping two variables without using extra space:

```
ASSERT(x==x0 && y==y0)
ASSERT(y == y0 && x - y + y == x0)
x = x - y;
ASSERT(y == y0 && x + y == x0)
ASSERT(x + y - x == y0 && x + y == x0)
y = x + y;
ASSERT(y - x == y0 && y == x0)
x = y - x;
ASSERT(x==y0 && y==x0)
```

- Will the same statement be valid for two indices in an array?

```
ASSERT(A[i]==A0[i] && A[j]==A0[j])
A[i] = A[i] - A[j];
A[j] = A[i] + A[j];
A[i] = A[j] - A[i];
ASSERT(A[i]==A0[j] && A[j]==A0[i])
```

In-place swap, the array edition

```
ASSERT(A[i]==A0[i] && A[j]==A0[j] && j != i) // necessarily stronger than before
ASSERT( ( j == i && 0 == A0[j] && 0 == A0[i] ) ||
        ( j != i && A[j] == A0[j] && A[i] == A0[i] ) )
ASSERT( ( j == i && 0 == A0[j] && 0 == A0[i] ) ||
        ( j != i && A[j] == A0[j] && A[i] - A[j] + A[j] == A0[i] ) )
ASSERT( ( j == i && 0 == A0[j] && 0 == A0[i] ) ||
        ( j != i && (A|i->A[i] - A[j])[j] == A0[j] &&
          (A|i->A[i] - A[j])[i] + (A|i->A[i] - A[j])[j] == A0[i] ) )
A[i] = A[i] - A[j];
ASSERT( ( j == i && 0 == A0[j] && 0 == A0[i] ) ||
        ( j != i && A[j] == A0[j] && A[i] + A[j] == A0[i] ) )
ASSERT( ( j == i && 0 == A0[j] || j != i && A[j] == A0[j] ) &&
        ( j == i && 0 == A0[i] || j != i && A[i] + A[j] == A0[i] )
ASSERT( ( j == i && A[i] + A[j] - A[i] - A[j] == A0[j] ||
          j != i && A[i] + A[j] - A[i] == A0[j] ) &&
        ( j == i && A[i] + A[j] - A[i] - A[j] == A0[i] ||
          j != i && A[i] + A[j] == A0[i] )
ASSERT( A[i] + A[j] - (A|j->A[i] + A[j])[i] == A0[j] &&
        ( j == i && A[i] + A[j] - A[i] - A[j] == A0[i] ||
          j != i && A[i] + A[j] == A0[i] )
ASSERT( (A|j->A[i] + A[j])[j] - (A|j->A[i] + A[j])[i] == A0[j] &&
        ( j == i && (A|j->A[i] + A[j])[j] - (A|j->A[i] + A[j])[i] == A0[i] ||
          j != i && (A|j->A[i] + A[j])[j] == A0[i] )
A[j] = A[i] + A[j];
ASSERT( A[j] - A[i] == A0[j] && ( j == i && A[j] - A[i] == A0[i] ||
                                  j != i && A[j] == A0[i] )
ASSERT( (A|i->A[j] - A[i])[i] == A0[j] && (A|i->A[j] - A[i])[j] == A0[i] )
A[i] = A[j] - A[i];

ASSERT(A[i]==A0[j] && A[j]==A0[i])
```


In-place swap, the array edition (cont'd)

```
ASSERT(A[i]==A0[i] && A[j]==A0[j]   && j != i)
A[i] = A[i] - A[j];
A[j] = A[i] + A[j];
A[i] = A[j] - A[i];
ASSERT(A[i]==A0[j] && A[j]==A0[i])
```

- In place swap of two indices `i` and `j` in the same array `A` will not work
  - Carrying on the proof shows that the statement is invalid whenever the two indices to be swapped coincide (that is, `i==j`)
  - Strengthening the pre-condition to forbid this situation will make the statement valid, though whether this is a reasonable pre-condition depends on the code that uses this fragment