1. Using the array-component assignment we get:

\[
\text{ASSERT( ( (A | j \rightarrow x) | i \rightarrow (A | j \rightarrow x)[k])][j] == 2 )}
\]
\[
A[j] = x;
\]
\[
\text{ASSERT( (A | i \rightarrow A[k])[j] == 2 )}
\]
\[
A[i] = A[k];
\]
\[
\text{ASSERT( A[j] == 2 )}
\]

The assertion P can be read from the top line of the above tableau. Next we write P in an equivalent form without array-component substitution notation:

When \(i = j\), P becomes:

\[
(A | j \rightarrow x)[k]) == 2
\]
- When \(i = j = k\), we get \(x == 2\).
- When \(i = j \neq k\), we get \(A[k] == 2\).

When \(i \neq j\), P becomes:

\[
(A | j \rightarrow x)[j] == 2, \text{ that is, } x == 2.
\]

Thus we can write P as follows:

\[
(i = j = k \&\& x == 2) \lor (i=j \&\& j = k \&\& A[k] == 2) \lor (i \neq j \&\& x == 2)
\]

2. As the invariant I we choose

\[
1 <= k <= n \&\& \text{ForAll}(i = 0, i < k) A[i] == (i+1)*(3*i + 4)/2
\]

Proof tableau:
ASSERT(1 <= n < max)
ASSERT(1 <= n && true)
// variable declaration does not affect reasoning since k
// does not appear in the assertion
int k;
// in below assertion the equality simplifies to 2==2 which is true
ASSERT(1<=n && (A| 0+->2)[0] == (0+1)(3*0 + 4)/2)
//Above equality is ForAll-statement with value i==0
//Below 1<=1 is true
ASSERT(1<=1<n && ForAll(i=0,i<1) (A| 0+-> 2)[i] == (i+1)*(3*i+4)/2)
k = 1;
ASSERT(1 <= k <= n && ForAll(i=0, i<k) (A| 0+-> 2)[i] == (i+1)*(3*i+4)/2)
A[0] = 2;
ASSERT(I)
while (k < n) {
ASSERT( I && k < n)
// k<n implies k+1 <= n when k, n are integers
// When i==k, (A | k+-> A[k-1] + 3*k+2)[i] is A[k-1]+3*k+2 and
// according to the invariant this is equal to
// k*(3*(k-1) +4)/2 + 3*k + 2 == (3*k*k + 7*k + 4)/2 == (k+1)(3k+4)/2
// When i < k, (A | k+-> A[k-1] + 3*k+2)[i] == A[i] and, according
// to the invariant this is equal to (i+1)*(3*i + 4)/2
ASSERT(1<=k+1<=n && ForAll(i = 0, i < k+1)
    (A | k+-> A[k-1] + 3*k+2)[i] == (i+1)*(3*i + 4)/2)
A[k] = A[k-1] + 3*k + 2;
ASSERT(1<=k+1<=n && ForAll(i = 0, i < k+1) A[i] == (i+1)*(3*i + 4)/2)
k = k+1;
ASSERT(I)
} //end-while

ASSERT( I && k >= n)
// k>=n && k <= n implies k == n
// With k==n, the invariant yields the forall-statement of
// the postcondition
ASSERT( ForAll(i = 0; i < n) A[i] == (i+1)*(3*i + 4)/2 )

The loop terminates because, according to the invariant, 1 <= k <= n and each iteration of the loop increments k by one.