Summation

- Sum up the numbers from 1 to n.
Summation

I: s == sum of numbers from 1 to k.
!B: k == n.

s=1; k = 1;
// { I }
while (k != n)
{
    // { s == sum of 1 to k }
    k++;
    // { s == sum of 1 to k-1 }
    s = s + k;
    // { s == sum of 1 to k }
}
// { s == sum of 1 to n }
Linear Search

- Search for $x$ in the array $A[\ ]$. To make the problem more general, let’s assume we search in range $[a,b]$, $a \leq b$. 

![Diagram showing range $[a,b]$]
Linear Search

- Search for \( x \) in the array \( A[\ ] \). To make the problem more general, let’s assume we search in range \([a, b]\), \( a \leq b \).
Ranges and Empty Ranges

\[ [a, b] : \text{ Empty if } a == b+1 \]
\[ a[ b] \text{ or } (a, b] : \text{ Empty if } a == b \]
\[ [a ]b \text{ or } [a, b) : \text{ Empty if } a == b \]
\[ a[ ]b \text{ or } (a, b) : \text{ Empty if } a == b-1 \]
Linear Search

- Search for x in the array A[]. To make the problem more general, let’s assume we search in range [a,b], a <= b.

\[ [a \quad b] \]

\{ I: [a ]k has been processed and no x is in [a ]k \}
\{ !B: k == b+1 || A[k] == x \}
Linear Search

- Search for x in the array A[]. To make the problem more general, let’s assume we search in range [a,b] and a <= b.

\[
\begin{align*}
&\{ I: [a \, ]k \text{ has been processed and no } x \text{ is in } [a \, ]k \} \\
&\{ !B: k == b+1 || A[k] == x \}
\end{align*}
\]

\[
k = a;
\]
\[
\text{While (k != b+1 && A[k] != x)} \\
\{ \\
\quad k = k+1; \\
\}
\]
\[
\text{if (k == b+1) x is not found; else it is A[k];}
\]
Find the Largest Number in an Array

- Find the largest number in $A[i..j]$. 
Find the Largest Number in an Array

- Find the largest number in A[i..j].

\{I: \text{x is the largest number in } A[i..k]\}
\{!B: k == j\}
Find the Largest Number in an Array

- Find the largest number in $A[i..j]$.

\[
\{I: \text{x is the largest number in } A[i..k]\} \\
\{!B: k == j\}
\]

\[
x = a[i]; k = i; \\
while (k != j) \\
{ \\
    k++;
    if (a[k] > x) x = a[k];
}
\]
Merge Two Sorted Arrays

- Problem: Merge two sorted arrays so that the resulted array is still sorted.

Example: \(A = \{1, 4, 6\}\)
\[B = \{2, 5\}\]

The merged result will be:
\[\{1, 2, 4, 5, 6\}\]
Merge Two Sorted Arrays

- Problem: Merge two sorted arrays so that the resulted array is still sorted.

  Merge A[i .. j] and B[m .. n] and put the result in C[0 .. ]
Merge Two Sorted Arrays

- **Problem**: Merge two sorted arrays so that the resulted array is still sorted.

Merge \( A[i..j] \) and \( B[m..n] \) and put the result in \( C[0..] \)

\[
\{ I: \text{A[i]p and B[m]q have been processed and the result has been put in C[0]k in sorted order, if p is in [i,j], A[p] >= C[0]k, if q is in [m,n], B[q] >= C[0]k,} \\
\{ !B: p == j+1 and q == n+1 \}\]
Merge Two Sorted Arrays

- Problem: Merge two sorted arrays so that the resulted array is still sorted.

Merge A[i .. j] and B[m .. n] and put the result in C[0 .. ]

```java
{ I: A[i ]p and B[m ]q have been processed and the result has been put in C[0 ]k in sorted order}
{!B: p == j+1 and q == n+1}
```

```
p=i; q = m; k=0;
while (p != j+1 || q != n+1)
{
    if (p == j+1) { C[k] = B[q]; k++; q++;}
    else if (q == n+1) {C[k] = A[p]; k++; p++;}
    else { if (A[p] < B[q]) { C[k] = A[p]; k++; p++;}
        else {C[k] = B[q]; k++; q++;}
    }
}
```