Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Analyzing Selection Sort

Quick Recap of Relevant Topics

- Selection sort
  - Intuition
  - C++ implementation

Overview of This Lecture

- Analyzing performance of selection sort
    - Counting “basic” steps in sorting an array of size n

Selection Sort Animated

Total
24
18
17
25
27
24

Selection Sort in C++

```
int main() {
  … Declarations, input validation and reading elements of array A …
  // Selection sort
  int currTop, currMaxIndex;  // A[currTop] … A[n-1] is unsorted array
  for (currTop = 0; currTop < n; currTop ++) {
      currMaxIndex = findIndexOfMax(A, currTop, n);
      swap(A, currTop, currMaxIndex);
  }
   … Rest of code …
  return 0;
}
```

Selection Sort in C++

```
// PRECONDITION: start < end
// start, end within array bounds of A
int  findIndexOfMax(int A[],  int start,  int end) {
    int i, currMaxIndex = start;
    for ( i = start ; i < end; i++ ) {
      if (A[i] >= A[currMaxIndex]) { currMaxIndex = i; }
    }
   return currMaxIndex;
 }
// POSTCONDITION: A[currMaxIndex] at least as large as
// all elements in A[start] through A[end-1], no change in A
```

Selection Sort in C++

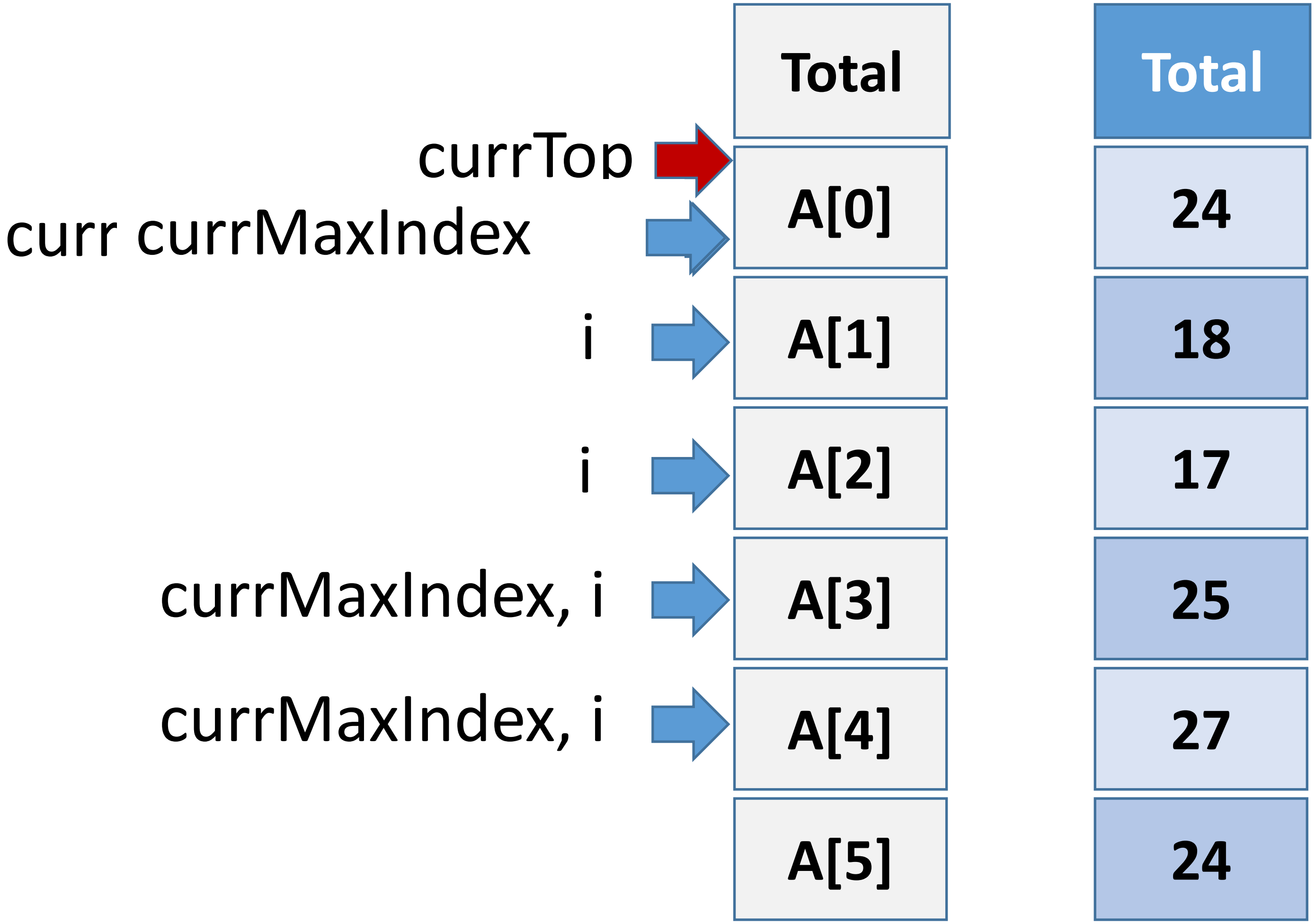
```
// PRECONDITION: index1, index2 within array
//                              bounds of A
void  swap(int A[],  int index1,  int index2) {
    int temp;
    temp = A[index1];
    A[index1] = A[index2];
    A[index2] = temp;
    return;
}
// POSTCONDITION: A[index1], A[index2] swapped
//                                Array A changed
```

“Basic” Steps in Selection Sort

- Reading two elements of array A, comparing them and updating currMaxIndex, if necessary
- Swapping two specified elements of array A

Given an array of n integers, how many “basic” steps (as a function of n) are needed to sort by selection sort?

Counting “Basic” Steps In Selection Sort

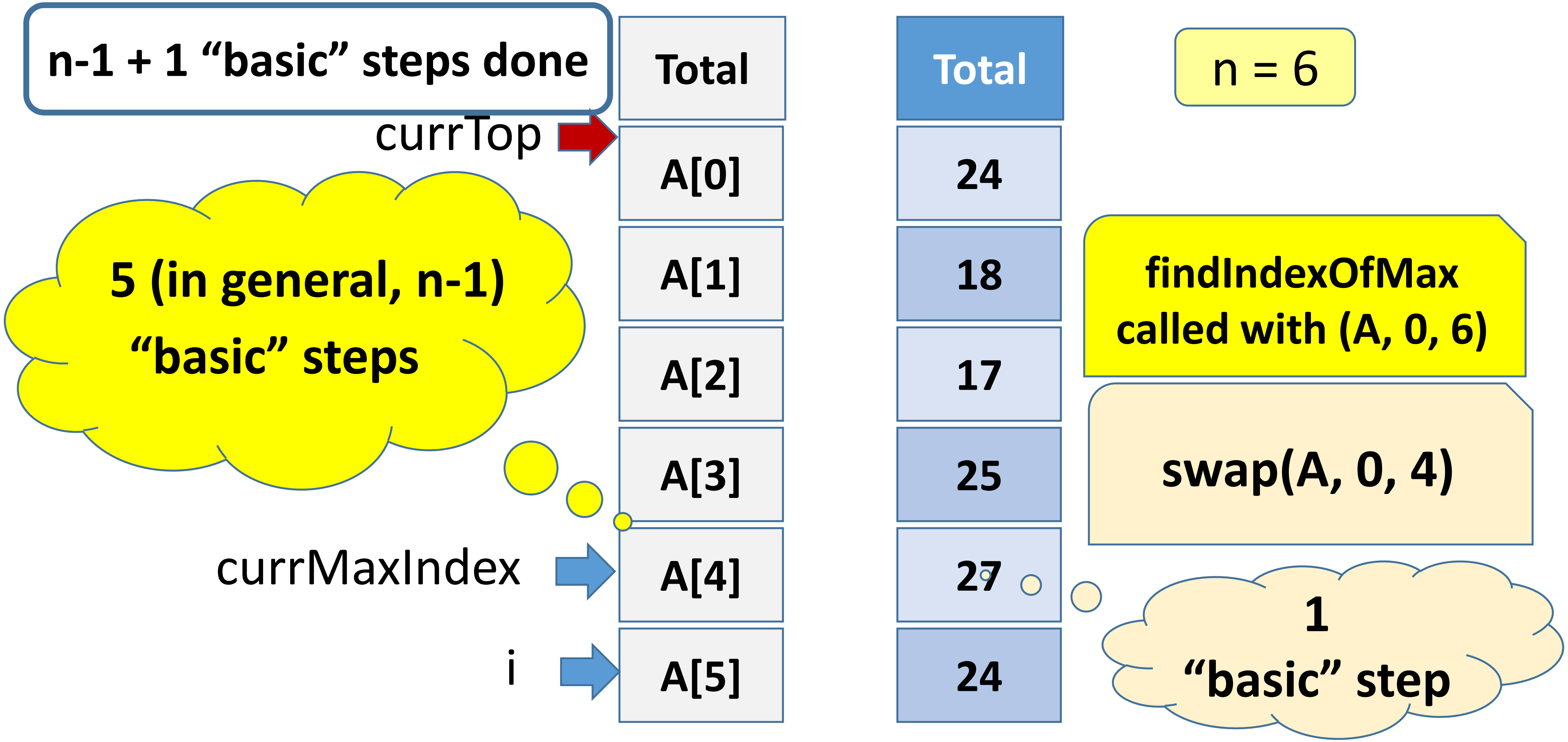


	Total	Total
currTop curr currMaxIndex	A[0]	24
i	A[1]	18
i	A[2]	17
currMaxIndex, i	A[3]	25
currMaxIndex, i	A[4]	27
	A[5]	24

n = 6

findIndexOfMax called with (A, 0, 6)

Counting “Basic” Steps In Selection Sort



n-1 + 1 “basic” steps done

n = 6

Total	Total
A[0]	24
A[1]	18
A[2]	17
A[3]	25
A[4]	27
A[5]	24

5 (in general, n-1) “basic” steps

findIndexOfMax called with (A, 0, 6)

swap(A, 0, 4)

1 “basic” step

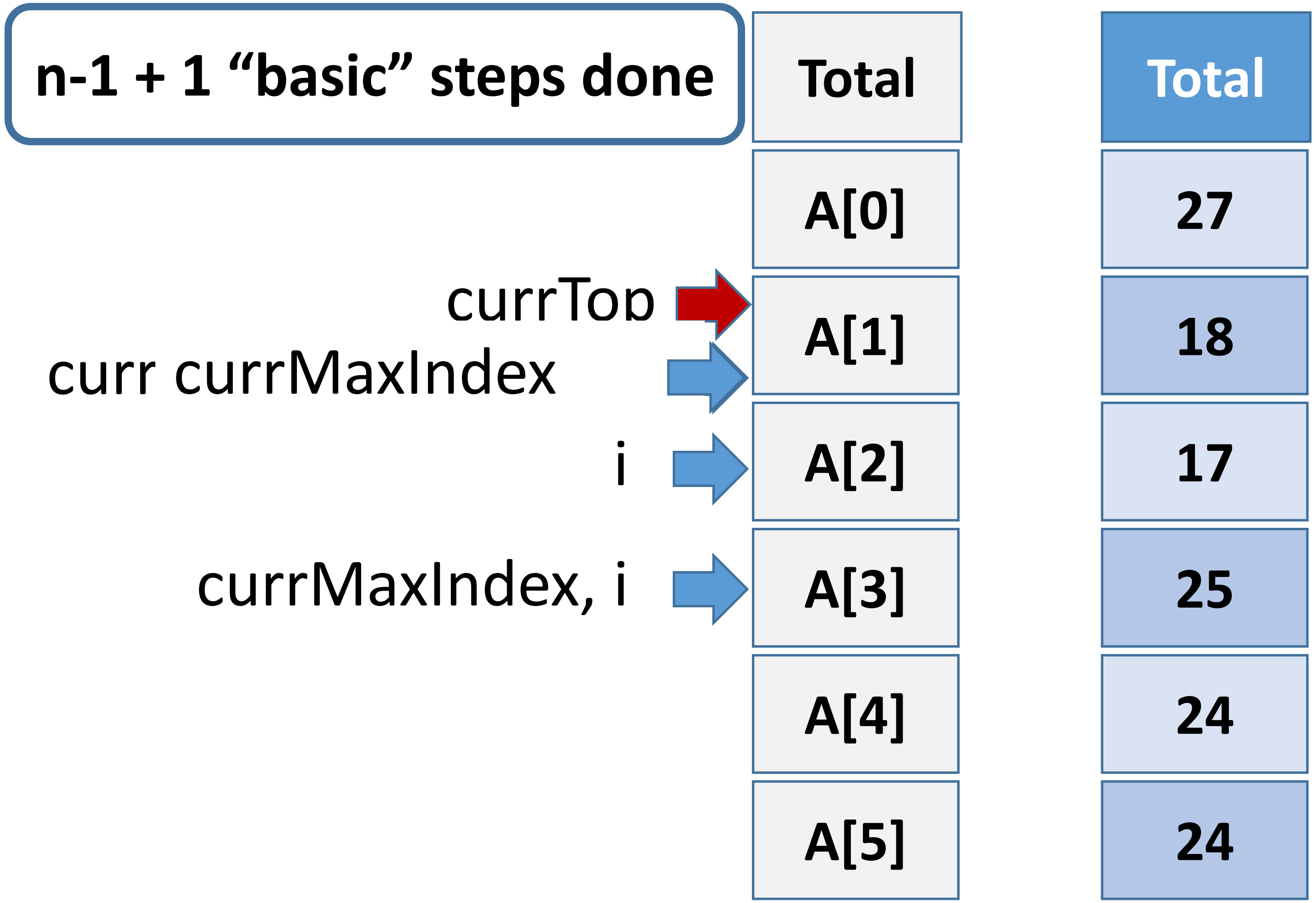
Recall: Selection Sort in C++

```
int main() {
   … Declarations, input validation and reading elements of array A …
  // Selection sort
  int currTop, currMaxIndex;  // A[currTop] … A[n-1] is unsorted array
  for (currTop = 0; currTop < n; currTop ++) {
      currMaxIndex = findIndexOfMax(A, currTop, n);
      swap(A, currTop, currMaxIndex);
  }
   … Rest of code …
  return 0;
}
```

Counting “Basic” Steps In Selection Sort

n-1 + 1 “basic” steps done

n = 6



Total	Total
A[0]	27
A[1]	18
A[2]	17
A[3]	25
A[4]	24
A[5]	24

findIndexOfMax called with (A, 1, 6)

Counting “Basic” Steps In Selection Sort

IIT Bombay

n-1 + 1 “basic” steps done

n = 6

	Total	Total
	A[0]	27
currTop	A[1]	18
	A[2]	17
currMaxIndex	A[3]	25
i	A[4]	24
i	A[5]	24

findIndexOfMax called with (A, 1, 6)

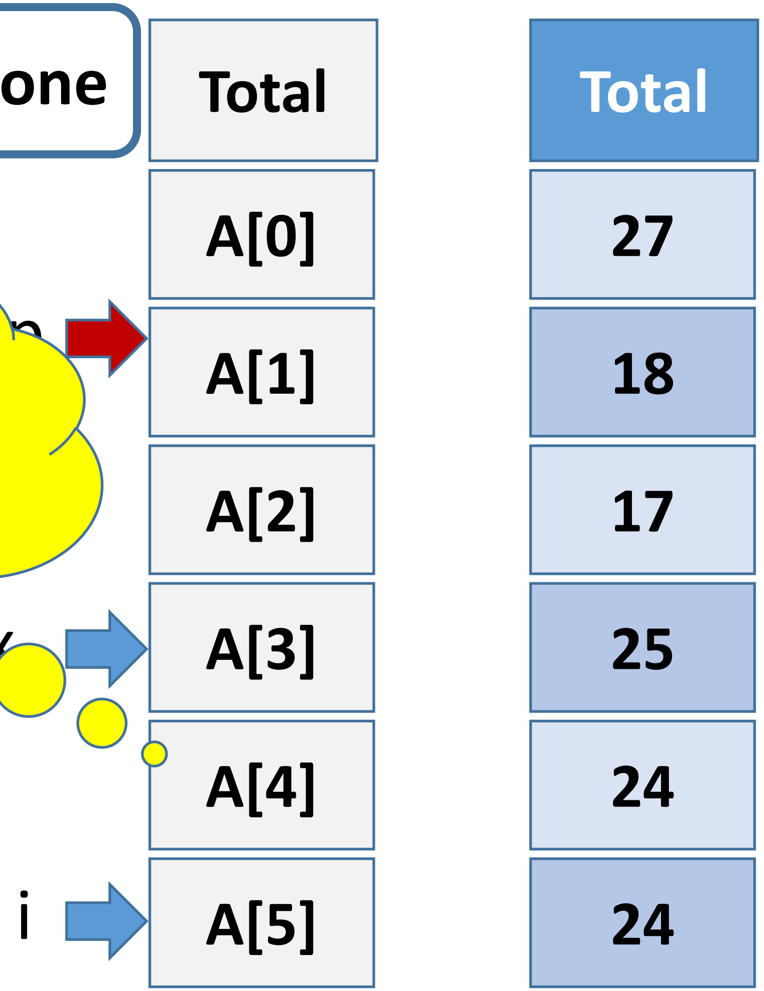
Counting "Basic" Steps In Selection Sort

IIT Bombay

n-1 + 1 "basic" steps done

n = 6

4 (in general, n-2) "basic" steps



Total	Total
A[0]	27
A[1]	18
A[2]	17
A[3]	25
A[4]	24
A[5]	24

findIndexOfMax called with (A, 1, 6)

Counting "Basic" Steps In Selection Sort

IIT Bombay

n-1 + 1 "basic" steps done

n-2 + 1 "basic" steps done

p

4 (in general, n-2) "basic" steps

xIndex

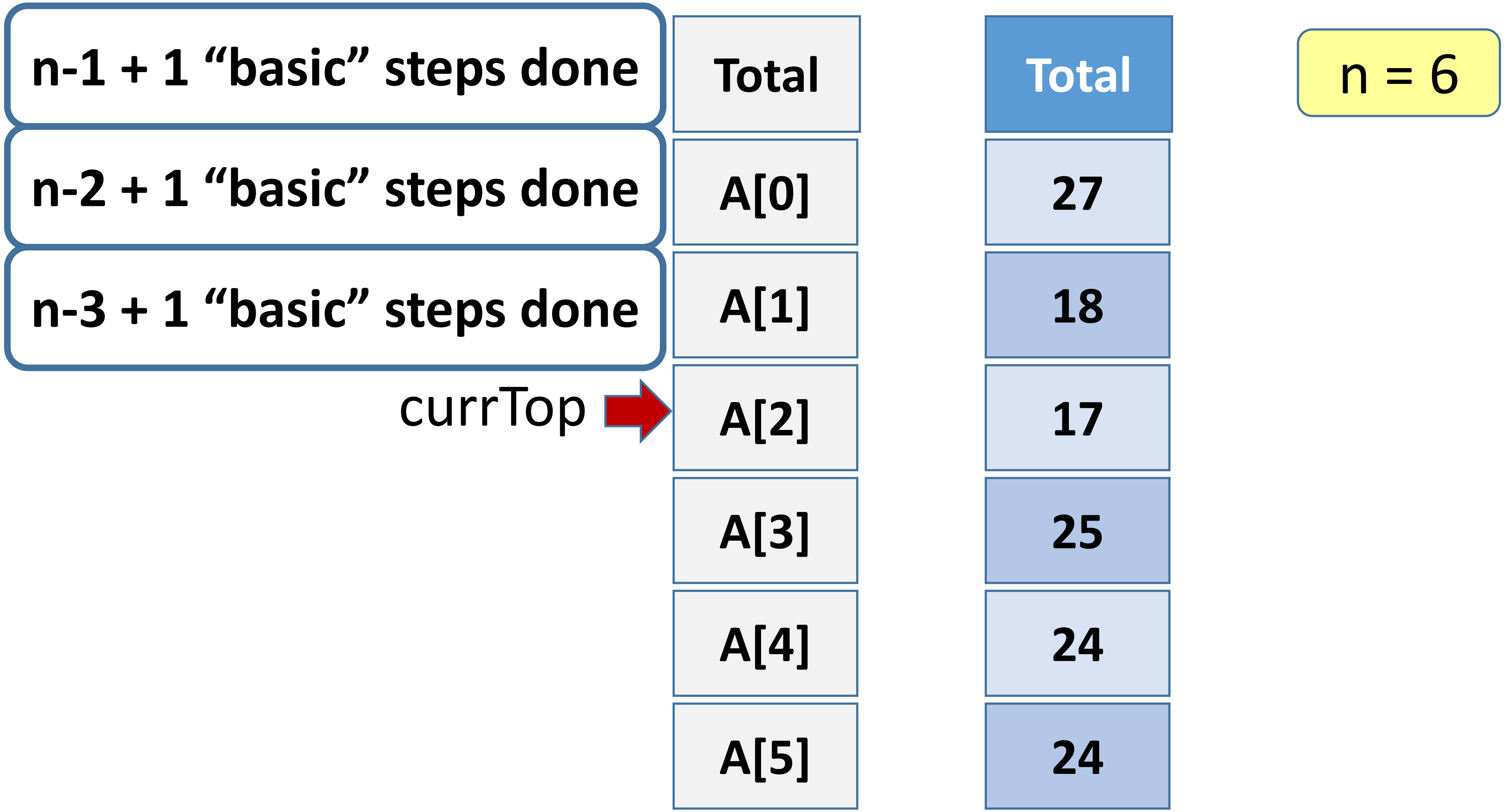
Total	Total
A[0]	27
A[1]	18
A[2]	17
A[3]	25
A[4]	24
A[5]	24

n = 6

swap(A, 1, 3)

1 "basic" step

Counting “Basic” Steps In Selection Sort



n-1 + 1 “basic” steps done

n-2 + 1 “basic” steps done

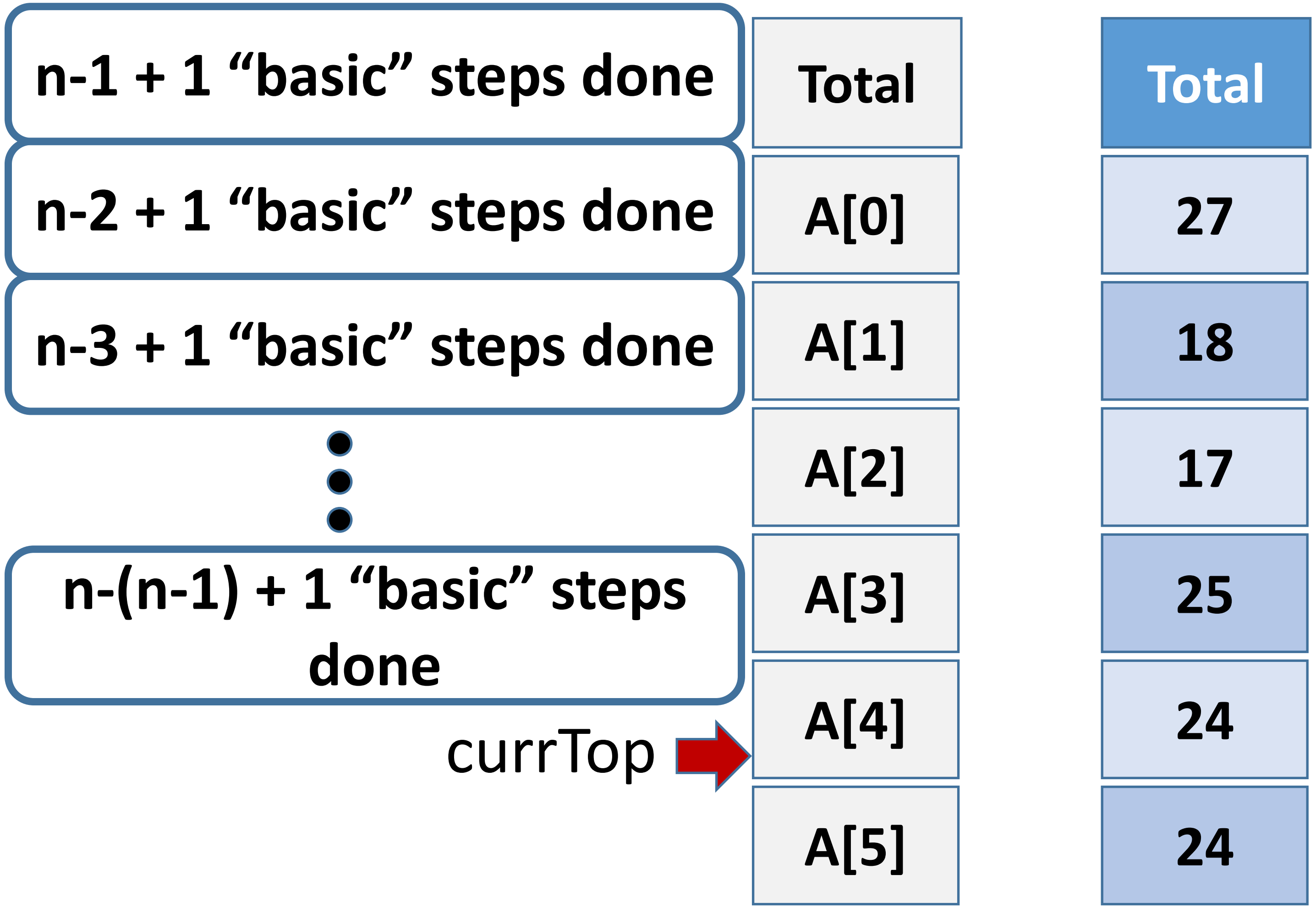
n-3 + 1 “basic” steps done

currTop

n = 6

Total	Total
A[0]	27
A[1]	18
A[2]	17
A[3]	25
A[4]	24
A[5]	24

Counting “Basic” Steps In Selection Sort



n-1 + 1 “basic” steps done

n-2 + 1 “basic” steps done

n-3 + 1 “basic” steps done

⋮

n-(n-1) + 1 “basic” steps done

Total	Total
A[0]	27
A[1]	18
A[2]	17
A[3]	25
A[4]	24
A[5]	24

n = 6

Counting “Basic” Steps in Selection Sort

- Count of “basic” steps to sort an array of n elements:

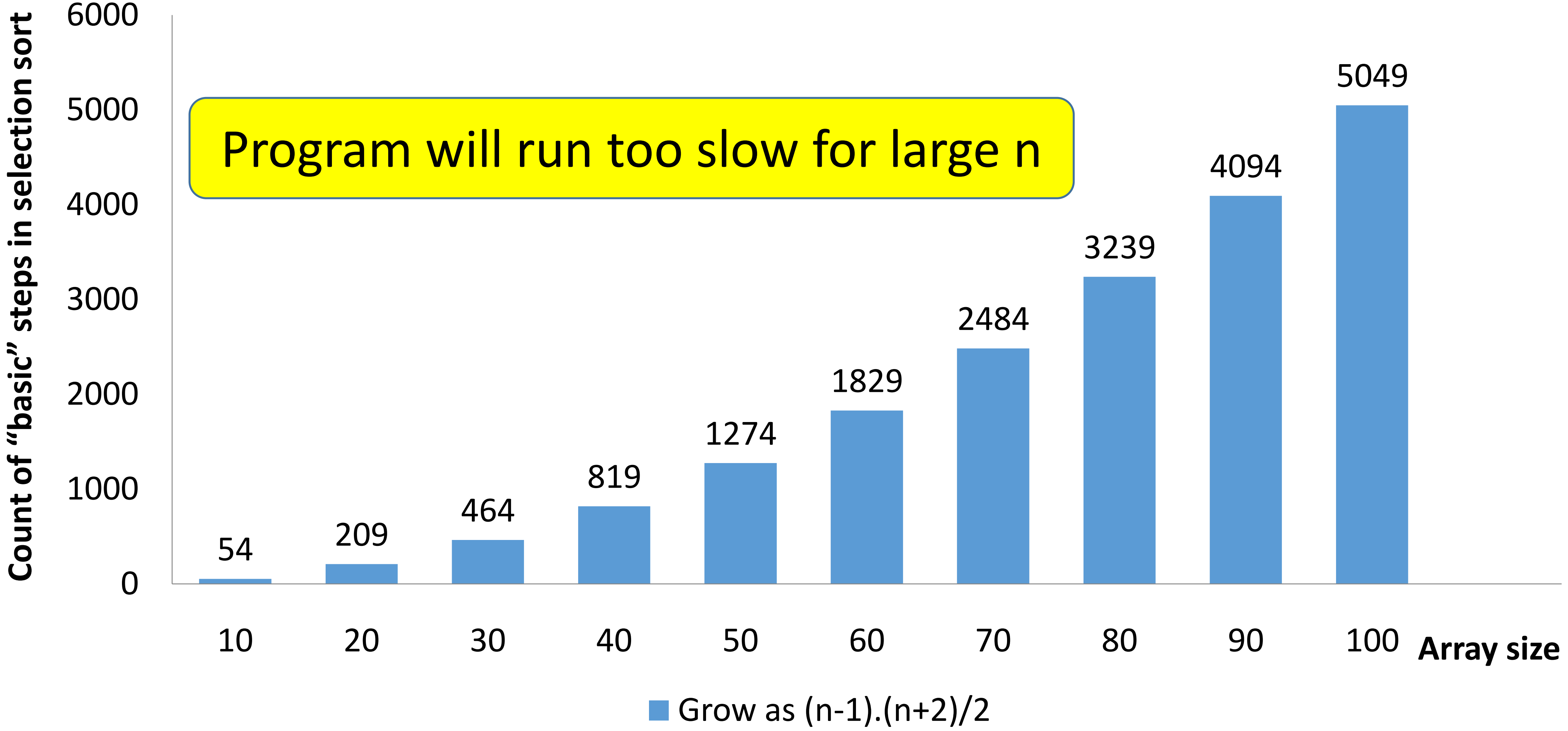
$$
\begin{aligned}
&(n-1 \quad + \quad 1) \;+ \\
&(n-2 \quad + \quad 1) \;+ \\
&\qquad\quad \vdots \\
&(n-(n-1) \;+\; 1)
\end{aligned}
$$

$$= \quad (1 + 2 + \ldots\ n-1) + n-1 \;= (n - 1) \times (n+2)/2$$

Increases quadratically with n

Quadratic Growth With n

IIT Bombay



Program will run too slow for large n

Is Selection Sort Fast Enough?

- Real-world sorting requirements
  - Query generating 1 million data items, each with a score
  - Selection sort too slow for such applications
    - With $n = 10^6$, $(n-1).(n+2)/2 \approx 5 \times 10^{11}$
    - If each "basic" step takes 20 ns (memory reads and writes, comparison, etc.), we need 10^4 seconds (approx. 2.78 hours)!!!
- Can we do better?
  - Yes, much better !!!
  - Approximately $(n. \log_2 n)$ "basic" steps to sort an array of size n
    - **10^6 elements can be sorted in no more than a few seconds!**
  - Topic of next few lectures …

Summary

- Analysis of performance of selection sort
    - Count of "basic" steps grows quadratically with size of array
- Need for faster sorting techniques