Insertion-Sort

Analysis and Design of Algorithms

Research Group on Artificial Life – Grupo de investigación en vida artificial – (Alife)
Computer and System Department
Engineering School
Universidad Nacional de Colombia

Outline

Outline

1. Sorting problem

2. Analyzing algorithms

Outline

1. Sorting problem

2. Analyzing algorithms

Sorting problem

Computational problem

Input: A sequence of n numbers $\langle a_1, a_2, \ldots, a_n \rangle$.

Output: A permutation (reordering) $\langle a'_1, a'_2, \ldots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

Sorting problem

Computational problem

Input: A sequence of n numbers $\langle a_1, a_2, \ldots, a_n \rangle$.

Output: A permutation (reordering) $\langle a'_1, a'_2, \ldots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

Sorting problem

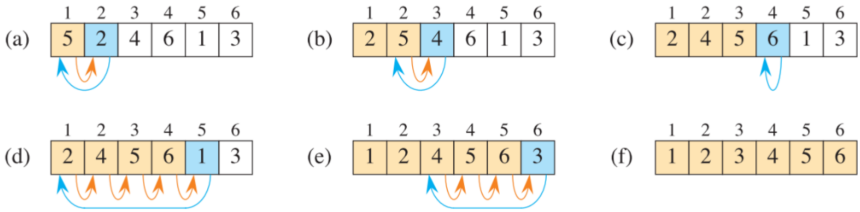
Computational problem

Input: A sequence of n numbers $\langle a_1, a_2, \ldots, a_n \rangle$.

Output: A permutation (reordering) $\langle a'_1, a'_2, \ldots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

Insertion-Sort on cards

Insertion-Sort example

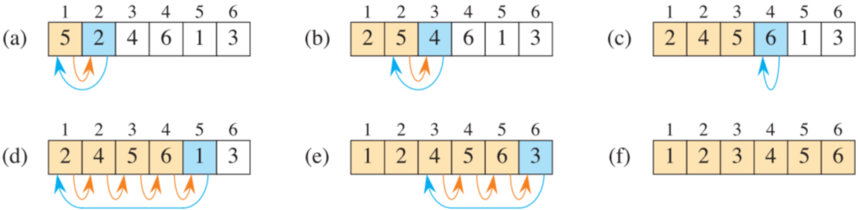


Insertion-Sort pseudocode I

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i – 1].
    j = i – 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j – 1
    A[j + 1] = key
```

Insertion-Sort pseudocode II

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i − 1].
    j = i − 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j − 1
    A[j + 1] = key
```



Loop invariants I

Definition (Loop Invariants)

A statement (property about a loop) is said to be a **loop invariant** if we can show following three properties.

Initialization: It is true prior to the first iteration of the loop.

Maintenance: If it is true before an iteration of the loop, it remains true before the next iteration.

Termination: When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

Loop invariants I

Definition (Loop Invariants)

A statement (property about a loop) is said to be a **loop invariant** if we can show following three properties.

Initialization: It is true prior to the first iteration of the loop.

Maintenance: If it is true before an iteration of the loop, it remains true before the next iteration.

Termination: When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

Loop invariants I

Definition (Loop Invariants)

A statement (property about a loop) is said to be a **loop invariant** if we can show following three properties.

Initialization: It is true prior to the first iteration of the loop.

Maintenance: If it is true before an iteration of the loop, it remains true before the next iteration.

Termination: When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

Loop invariants I

Definition (Loop Invariants)

A statement (property about a loop) is said to be a **loop invariant** if we can show following three properties.

Initialization: It is true prior to the first iteration of the loop.

Maintenance: If it is true before an iteration of the loop, it remains true before the next iteration.

Termination: When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

The loop invariants to help us understand why an algorithm is correct.

Loop invariants II

```
INSERTION-SORT(A, n)
1  for i = 2 to n
2      key = A[i]
3      // Insert A[i] into the sorted subarray A[1 : i − 1].
4      j = i − 1
5      while j > 0 and A[j] > key
6          A[j + 1] = A[j]
7          j = j − 1
8      A[j + 1] = key
```

Loop invariant

At the start of each iteration of the **for** loop of lines 1–8, the subarray $A[1 : i - 1]$ consists of the elements originally in $A[1 : i - 1]$, but in sorted order.

Loop invariants

Initialization

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i − 1].
    j = i − 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j − 1
    A[j + 1] = key
```

Initialization for Insertion-Sort loop

We start by showing that the loop invariant holds before the first loop iteration, when $i = 2$. The subarray $A[1 : i - 1]$ consists of just the single element $A[1]$, which is in fact the original element in $A[1]$. Moreover, this subarray is sorted, which shows that the loop invariant holds prior to the first iteration of the loop.

Loop invariants

Maintenance

```
INSERTION-SORT(A, n)
1  for i = 2 to n
2      key = A[i]
3      // Insert A[i] into the sorted subarray A[1 : i – 1].
4      j = i – 1
5      while j > 0 and A[j] > key
6          A[j + 1] = A[j]
7          j = j – 1
8      A[j + 1] = key
```

Maintenance for insertion-sort loop

Next, we tackle the second property: showing that each iteration maintains the loop invariant. Informally, the body of the for loop works by moving the values in $A[i-1]$, $A[i-2]$, $A[i-3]$, and so on by one position to the right until it finds the proper position for $A[i]$ (lines 4–7), at which point it inserts the value of $A[i]$ (line 8). The subarray $A[1:i]$ then consists of the elements originally in $A[1:i]$, but in sorted order. Incrementing i (increasing its value by 1) for the next iteration of the **for** loop then preserves the loop invariant.

Loop invariants

Termination

```
INSERTION-SORT(A, n)
1  for i = 2 to n
2      key = A[i]
3      // Insert A[i] into the sorted subarray A[1 : i – 1].
4      j = i – 1
5      while j > 0 and A[j] > key
6          A[j + 1] = A[j]
7          j = j – 1
8      A[j + 1] = key
```

Termination for insertion-sort loop

We examine loop termination. The loop variable i starts at 2 and increases by 1 in each iteration. Once i's value exceeds n in line 1, the loop terminates. That is, the loop terminates once i equals $n+1$. Substituting $n+1$ for i in the wording of the loop invariant yields that the subarray $A[1:n]$ consists of the elements originally in $A[1:n]$, but in sorted order. Hence, the algorithm is correct.

Outline

1 Sorting problem

2 Analyzing algorithms

Space required

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i − 1].
    j = i − 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j − 1
    A[j + 1] = key
```

$\Theta(1)$: Only two auxiliary variables are required: *key* and j.

In-place: The ordering is done *in situ*, it uses the same space of the array to store it in sorted order.

Space required

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i – 1].
    j = i – 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j – 1
    A[j + 1] = key
```

$\Theta(1)$: Only two auxiliary variables are required: *key* and j.

In-place: The ordering is done *in situ*, it uses the same space of the array to store it in sorted order.

Space required

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i − 1].
    j = i − 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j − 1
    A[j + 1] = key
```

$\Theta(1)$: Only two auxiliary variables are required: *key* and j.

In-place: The ordering is done *in situ*, it uses the same space of the array to store it in sorted order.

Important formulas

- $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \frac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \frac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \frac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \frac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \frac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \frac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \dfrac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \dfrac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \dfrac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \dfrac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \dfrac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} ca_i = c\sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n}(a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1+2+3+4+5+\cdots+(n-1)+n = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2+2^2+3^2+\cdots+(n-1)^2+n^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3+2^3+3^3+\cdots+(n-1)^3+n^3 = \frac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n}(i-1) = \frac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \dfrac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \dfrac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \dfrac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \dfrac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \dfrac{n(n-1)}{2}$

Important formulas

- $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$
- $\sum_{i=1}^{n} (a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$
- $\sum_{i=1}^{n} i = 1 + 2 + 3 + 4 + 5 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 = \frac{n^2(n+1)^2}{4}$
- $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$
- $\sum_{i=2}^{n} (i-1) = \frac{n(n-1)}{2}$

Running time

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

t_i: number of times line 5 is executed for i.

Running time

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $\quad$ $key = A[i]$	c_2	$n-1$
3 $\quad$ // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $\quad$ $j = i - 1$	c_4	$n-1$
5 $\quad$ **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $\quad\quad$ $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $\quad\quad$ $j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $\quad$ $A[j+1] = key$	c_8	$n-1$

t_i: number of times line 5 is executed for i.

General case

INSERTION-SORT(A, n)		cost	times
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n}(t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n}(t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

t_i: number of times line 5 is executed for i.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n}(t_i - 1) + c_7 \sum_{i=2}^{n}(t_i - 1) + c_8(n-1)$$

General case

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$\quad key = A[i]$	c_2	$n-1$
3	$\quad$ // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$\quad j = i - 1$	c_4	$n-1$
5	$\quad$ **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$\qquad A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n}(t_i - 1)$
7	$\qquad j = j - 1$	c_7	$\sum_{i=2}^{n}(t_i - 1)$
8	$\quad A[j+1] = key$	c_8	$n-1$

t_i: number of times line 5 is executed for i.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n}(t_i - 1) + c_7 \sum_{i=2}^{n}(t_i - 1) + c_8(n-1)$$

Best case: $t_i = 1$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $\quad key = A[i]$	c_2	$n-1$
3 $\quad$ // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $\quad j = i - 1$	c_4	$n-1$
5 $\quad$ **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $\quad\quad A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $\quad\quad j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $\quad A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \le 0, \vee, A[j] \le key}$$

$$\begin{aligned} T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1) \\ &= (c_1 + c_2 + c_4 + c_5 + c_8) n - (c_2 + c_4 + c_5 + c_8) \end{aligned}$$

Best case: $t_i = 1$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $\quad key = A[i]$	c_2	$n-1$
3 $\quad$ // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $\quad j = i-1$	c_4	$n-1$
5 $\quad$ **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $\quad\quad A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n}(t_i - 1)$
7 $\quad\quad j = j-1$	c_7	$\sum_{i=2}^{n}(t_i - 1)$
8 $\quad A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\sum_{i=2}^{n} t_i + c_6\sum_{i=2}^{n}(t_i-1) + c_7\sum_{i=2}^{n}(t_i-1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \le 0, \vee, A[j] \le key}$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$$
$$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$$

Best case: $t_i = 1$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i - 1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \le 0, \vee, A[j] \le key}$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$$
$$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$$

Best case: $t_i = 1$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) =$$
$$c_1 n + c_2(n-1) + c_4(n-1)$$
$$+ c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1)$$
$$+ c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \leq 0, \vee, A[j] \leq key}$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$$
$$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$$

Best case: $t_i = 1$

	Insertion-Sort(A, n)	cost	times
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \leq 0, \vee, A[j] \leq key}$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$$
$$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$$

Best case: $t_i = 1$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i-1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n}(t_i - 1)$
7	$j = j-1$	c_7	$\sum_{i=2}^{n}(t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\sum_{i=2}^{n} t_i + c_6\sum_{i=2}^{n}(t_i-1) + c_7\sum_{i=2}^{n}(t_i-1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = 1; \qquad j \le 0, \vee, A[j] \le key}$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1)$$
$$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = i}$.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i-1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j-1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = i}$.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i - 1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$\boxed{t_i = i}$.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

		cost	times
	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = i}$.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right)n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right)n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i - 1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = i}$.

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n - (c_2 + c_4 + c_5 + c_8)$$

Worst case: $t_i = i$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = i}$.

$$\begin{aligned} T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) \\ &\quad + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1) \\ &= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right) n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right) n \\ &\quad - (c_2 + c_4 + c_5 + c_8) \end{aligned}$$

Average case: $t_i = i/2$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i-1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j-1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = \frac{i}{2}}.$$

$$T(n) = ???$$

Hint: $\sum_{i=1}^{n} ca_i = c \sum_{i=1}^{n} a_i$

Average case: $t_i = i/2$

INSERTION-SORT(A, n)	*cost*	*times*
1 **for** $i = 2$ **to** n	c_1	n
2 $key = A[i]$	c_2	$n-1$
3 // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4 $j = i - 1$	c_4	$n-1$
5 **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6 $A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7 $j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8 $A[j+1] = key$	c_8	$n-1$

$T(n) =$

$$c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = \frac{i}{2}}.$$

$$T(n) = ???$$

Hint: $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$

Average case: $t_i = i/2$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i-1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j-1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = \frac{i}{2}}.$$

$$T(n) = ???$$

Hint: $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$

Average case: $t_i = i/2$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = \frac{i}{2}}$.

$$T(n) = ???$$

Hint: $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$

Average case: $t_i = i/2$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$
\begin{aligned}
T(n) &= \\
& c_1 n + c_2(n-1) + c_4(n-1) \\
& + c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1) \\
& + c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)
\end{aligned}
$$

t_i: number of times line 5 is executed for i; $\boxed{t_i = \dfrac{i}{2}}$.

$T(n) = ???$

Hint: $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$

Average case: $t_i = i/2$

	INSERTION-SORT(A, n)	*cost*	*times*
1	**for** $i = 2$ **to** n	c_1	n
2	$key = A[i]$	c_2	$n-1$
3	// Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
4	$j = i - 1$	c_4	$n-1$
5	**while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
6	$A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
7	$j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
8	$A[j+1] = key$	c_8	$n-1$

$$T(n) =$$
$$c_1 n + c_2(n-1) + c_4(n-1)$$
$$+ c_5 \sum_{i=2}^{n} t_i + c_6 \sum_{i=2}^{n} (t_i - 1)$$
$$+ c_7 \sum_{i=2}^{n} (t_i - 1) + c_8(n-1)$$

t_i: number of times line 5 is executed for i;

$$\boxed{t_i = \frac{i}{2}}.$$

$$T(n) = ???$$

Hint: $\sum_{i=1}^{n} c a_i = c \sum_{i=1}^{n} a_i$

Asymptotic analysis I

Problem

- Let $T(n)$ denote the running time of Insertion-Sort.
- Fill the following table by determining, in each cell, which Δ in $\{\Theta, \Omega, \mathcal{O}, \omega, o\}$ will make the expression $T(n) = \Delta(f(n))$ true.

Asymptotic analysis I

Problem

- Let $T(n)$ denote the running time of Insertion-Sort.
- Fill the following table by determining, in each cell, which Δ in $\{\Theta, \Omega, \mathcal{O}, \omega, o\}$ will make the expression $T(n) = \Delta(f(n))$ true.

Asymptotic analysis I

Problem

- Let $T(n)$ denote the running time of Insertion-Sort.
- Fill the following table by determining, in each cell, which Δ in $\{\Theta, \Omega, \mathcal{O}, \omega, o\}$ will make the expression $T(n) = \Delta(f(n))$ true.

Case $\setminus$ $f(n)$	$\Delta(1)$	$\Delta(n)$	$\Delta(n^2)$	$\Delta(n^3)$
Best Case				
Worst Case				
Average Case				
General Case				

Asymptotic analysis II

Problem

What is the complexity of the following algorithm in the general case, best and worst?

```
MISTERY(n)
  x = 0
  for i = 1 to n do
      for j = 1 to i do
```

Asymptotic analysis II

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$
for $i = 1$ **to** n **do**
    for $j = 1$ **to** i **do**

Asymptotic analysis II

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

```
x = 0
for i = 1 to n do
    for j = 1 to i do
        x = x * x + 2 * x + 1
```

Asymptotic analysis II

Problem

What is the complexity of the following algorithm in the general case, best and worst?

```
MISTERY(n)
x = 0
for i = 1 to n do
    for j = 1 to i do
        x = x * x + 2 * x + 1
```

Asymptotic analysis II

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$

for $i = 1$ **to** n **do**

  for $j = 1$ **to** i **do**

    $x = x * x + 2 * x + 1$

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

```
x = 0
for i = 1 to n do
    k = i
    while k > 1 do
```

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$
for $i = 1$ **to** n **do**
    $k = i$
    while $k > 1$ **do**

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$
for $i = 1$ **to** n **do**
    $k = i$
    while $k > 1$ **do**

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$
for $i = 1$ **to** n **do**
    $k = i$
    while $k > 1$ **do**
        $x = x + 1$
        $k = k/2$

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)

$x = 0$
for $i = 1$ **to** n **do**
    $k = i$
    while $k > 1$ **do**
        $x = x + 1$
        $k = k/2$

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

```
MISTERY(n)
x = 0
for i = 1 to n do
    k = i
    while k > 1 do
        x = x + 1
        k = k/2
```

Asymptotic analysis III

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(n)
1: $x = 0$
2: **for** $i = 1$ **to** n **do**
3: $\quad k = i$
4: $\quad$ **while** $k > 1$ **do**
5: $\quad\quad x = x + 1$
6: $\quad\quad k = k/2$

Asymptotic analysis IV

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(A, n)

for $i = 2$ **to** n **do**
  INSERTION-SORT(A, i)

Asymptotic analysis IV

Problem

What is the complexity of the following algorithm in the general case, best and worst?

MISTERY(A, n)
1: **for** $i = 2$ **to** n **do**
2: INSERTION-SORT(A, i)

Asymptotic analysis IV

Problem

What is the complexity of the following algorithm in the general case, best and worst?

$\textsc{Mistery}(A, n)$

1: **for** $i = 2$ **to** n **do**
2: $\quad\textsc{Insertion-Sort}(A, i)$