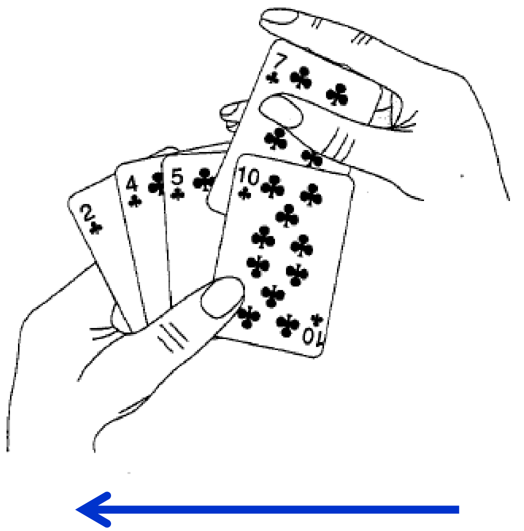
An Example: Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
      key = A[i]
      j = i - 1;
      while (j > 0) and (A[j] > key) {
           A[j+1] = A[j]
           j = j - 1
      }
      A[j+1] = key
  }
}
```



An Example: Insertion Sort

30	10	40	20
1	2	3	4

i = ∅ j = ∅ key = ∅
A[j] = ∅ A[j+1] = ∅

```
InsertionSort(A, n) {
  for i = 2 to n {
      key = A[i]
      j = i - 1;
      while (j > 0) and (A[j] > key) {
            A[j+1] = A[j]
            j = j - 1
      }
      A[j+1] = key
  }
}
```

An Example: Insertion Sort

30	*10*	*40*	*20*
1	2	3	4

i = 2 j = 1 key = 10
A[j] = 30 A[j+1] = 10

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

30	***30***	*40*	*20*
1	2	3	4

i = 2 j = 1	key = 10
A[j] = 30	A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
                A[j+1] = A[j]
                j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

30	*30*	*40*	*20*
1	2	3	4

i = 2 j = 1 key = 10
A[j] = 30 A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

1	2	3	4
30	*30*	*40*	*20*

i = 2 **j = 0** key = 10
A[j] = ∅ A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

1	2	3	4
30	*30*	*40*	*20*

i = 2   j = 0   key = 10

A[j] = ∅   A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

i = 2   j = 0   key = 10
A[j] = ∅   **A[j+1] = 10**

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	30	40	20
1	2	3	4

i = 3 j = 0 key = 10

A[j] = ∅ A[j+1] = 10

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

$i = 3$ $j = 0$ **key = 40**

$A[j] = \varnothing$ $A[j+1] = 10$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
                A[j+1] = A[j]
                j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	30	40	20
1	2	3	4

$i = 3 \quad j = 0 \quad key = 40$

$A[j] = \varnothing \quad A[j+1] = 10$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

i = 3 **j = 2** key = 40

A[j] = 30 **A[j+1] = 40**

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	30	40	20
1	2	3	4

$i = 3$ $j = 2$ $key = 40$
$A[j] = 30$ $A[j+1] = 40$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

$i = 3 \quad j = 2 \quad key = 40$

$A[j] = 30 \quad A[j+1] = 40$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

1	2	3	4
10	*30*	*40*	*20*

i = 4　　j = 2　　key = 40
A[j] = 30　　A[j+1] = 40

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
                A[j+1] = A[j]
                j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

i = 4 j = 2 **key = 20**

A[j] = 30 A[j+1] = 40

```
InsertionSort(A, n) {
   for i = 2 to n {
         key = A[i]
         j = i - 1;
         while (j > 0) and (A[j] > key) {
               A[j+1] = A[j]
               j = j - 1
         }
         A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	30	40	20
1	2	3	4

i = 4   j = 2   key = 20
A[j] = 30   A[j+1] = 40

```
InsertionSort(A, n) {
  for i = 2 to n {
       key = A[i]
       j = i - 1;
       while (j > 0) and (A[j] > key) {
             A[j+1] = A[j]
             j = j - 1
       }
       A[j+1] = key
  }
}
```

An Example: Insertion Sort

10	*30*	*40*	*20*
1	2	3	4

i = 4 **j = 3** key = 20

A[j] = 40 **A[j+1] = 20**

```
InsertionSort(A, n) {
  for i = 2 to n {
       key = A[i]
       j = i - 1;
       while (j > 0) and (A[j] > key) {
             A[j+1] = A[j]
             j = j - 1
       }
       A[j+1] = key
  }
}
```

An Example: Insertion Sort

10	30	40	20
1	2	3	4

$i = 4 \quad j = 3 \quad key = 20$

$A[j] = 40 \quad A[j+1] = 20$

```
InsertionSort(A, n) {
  for i = 2 to n {
       key = A[i]
       j = i - 1;
       while (j > 0) and (A[j] > key) {
             A[j+1] = A[j]
             j = j - 1
       }
       A[j+1] = key
  }
}
```

An Example: Insertion Sort

10	30	40	40
1	2	3	4

i = 4 j = 3 key = 20
A[j] = 40 **A[j+1] = 40**

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*40*
1	2	3	4

$i = 4 \quad j = 3 \quad key = 20$

$A[j] = 40 \quad A[j+1] = 40$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	30	40	40
1	2	3	4

$i = 4$	$j = 3$	$key = 20$
$A[j] = 40$		$A[j+1] = 40$

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
                A[j+1] = A[j]
                j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

1	2	3	4
10	*30*	*40*	*40*

i = 4 j = 2 key = 20

A[j] = 30 A[j+1] = 40

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*40*	*40*
1	2	3	4

i = 4 j = 2 key = 20
A[j] = 30 A[j+1] = 40

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	***30***	***40***
1	2	3	4

i = 4 j = 2 key = 20
A[j] = 30 **A[j+1] = 30**

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*30*	*40*
1	2	3	4

i = 4   j = 2   key = 20

A[j] = 30   A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*30*	*30*	*40*
1	2	3	4

i = 4   j = 1   key = 20

A[j] = 10   A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
                A[j+1] = A[j]
                j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

1	2	3	4
10	*30*	*30*	*40*

i = 4 j = 1 key = 20

A[j] = 10 A[j+1] = 30

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*20*	*30*	*40*
1	2	3	4

i = 4   j = 1   key = 20
A[j] = 10   **A[j+1] = 20**

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

An Example: Insertion Sort

10	*20*	*30*	*40*
1	2	3	4

i = 4 j = 1 key = 20

A[j] = 10 A[j+1] = 20

```
InsertionSort(A, n) {
   for i = 2 to n {
        key = A[i]
        j = i - 1;
        while (j > 0) and (A[j] > key) {
              A[j+1] = A[j]
              j = j - 1
        }
        A[j+1] = key
   }
}
```

Done!

Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
      key = A[i]
      j = i - 1;
      while (j > 0) and (A[j] > key) {
            A[j+1] = A[j]
            j = j - 1
      }
      A[j+1] = key
  }
}
```

*What is the **precondition** for this loop?*

Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
     key = A[i]
     j = i - 1;
     while (j > 0) and (A[j] > key) {
          A[j+1] = A[j]
          j = j - 1
     }
     A[j+1] = key
  }
}
```

How many times will this loop execute?

Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
     key = A[i]
     j = i - 1;
     while (j > 0) and (A[j] > key) {
          A[j+1] = A[j]
          j = j - 1
     }
     A[j+1] = key
  }
}
```

*What is the **post-condition** for this loop?*

Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
     key = A[i]
     j = i - 1;
     while (j > 0) and (A[j] > key) {
          A[j+1] = A[j]
          j = j - 1
     }
     A[j+1] = key
  }
}
```

Invariant: A [1..i-1] consists of the elements originally in A [1..i-1], but in sorted order

Insertion Sort

```
InsertionSort(A, n) {
  for i = 2 to n {
     key = A[i]
     j = i - 1;
     while (j > 0) and (A[j] > key) {
          A[j+1] = A[j]
          j = j - 1
     }
     A[j+1] = key
  }
}
```

Termination: when i == n+1 we have A[1..i-1] which leads to A[1..n]

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`for i = 2 to n {`	$c_1 n$
`key = A[i]`	$c_2(n-1)$
`j = i - 1;`	$c_3(n-1)$
`while (j > 0) and (A[j] > key) {`	$c_4\sum_{j=2}^{n} t_j$
`A[j+1] = A[j]`	$c_5\sum_{j=2}^{n}(t_j - 1)$
`j = j - 1`	$c_6\sum_{j=2}^{n}(t_j - 1)$
`}`	0
`A[j+1] = key`	$c_7(n-1)$
`}`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`for i = 2 to n {`	$c_1 n$
`key = A[i]`	$c_2(n-1)$
`j = i - 1;`	$c_3(n-1)$
`while (j > 0) and (A[j] > key) {`	$c_4\sum_{j=2}^{n} t_j$
`A[j+1] = A[j]`	$c_5\sum_{j=2}^{n}(t_j - 1)$
`j = j - 1`	$c_6\sum_{j=2}^{n}(t_j - 1)$
`}`	0
`A[j+1] = key`	$c_7(n-1)$
`}`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`for i = 2 to n {`	$c_1 n$
`key = A[i]`	$c_2(n-1)$
`j = i - 1;`	$c_3(n-1)$
`while (j > 0) and (A[j] > key) {`	$c_4\sum_{j=2}^{n} t_j$
`A[j+1] = A[j]`	$c_5\sum_{j=2}^{n}(t_j - 1)$
`j = j - 1`	$c_6\sum_{j=2}^{n}(t_j - 1)$
`}`	0
`A[j+1] = key`	$c_7(n-1)$
`}`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`  for i = 2 to n {`	$c_1 n$
`    key = A[i]`	$c_2(n-1)$
`    j = i - 1;`	$c_3(n-1)$
`    while (j > 0) and (A[j] > key) {`	$c_4 \sum_{j=2}^{n} t_j$
`      A[j+1] = A[j]`	$c_5 \sum_{j=2}^{n} (t_j - 1)$
`      j = j - 1`	$c_6 \sum_{j=2}^{n} (t_j - 1)$
`    }`	0
`    A[j+1] = key`	$c_7(n-1)$
`  }`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`  for i = 2 to n {`	$c_1 n$
`    key = A[i]`	$c_2(n-1)$
`    j = i - 1;`	$c_3(n-1)$
`    while (j > 0) and (A[j] > key) {`	$c_4 \sum_{j=2}^{n} t_j$
`      A[j+1] = A[j]`	$c_5 \sum_{j=2}^{n} (t_j - 1)$
`      j = j - 1`	$c_6 \sum_{j=2}^{n} (t_j - 1)$
`    }`	0
`    A[j+1] = key`	$c_7(n-1)$
`  }`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`  for i = 2 to n {`	$c_1 n$
`    key = A[i]`	$c_2(n-1)$
`    j = i - 1;`	$c_3(n-1)$
`    while (j > 0) and (A[j] > key) {`	$c_4 \sum_{j=2}^{n} t_j$
`      A[j+1] = A[j]`	$c_5 \sum_{j=2}^{n} (t_j - 1)$
`      j = j - 1`	$c_6 \sum_{j=2}^{n} (t_j - 1)$
`    }`	0
`    A[j+1] = key`	$c_7(n-1)$
`  }`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`  for i = 2 to n {`	$c_1 n$
`    key = A[i]`	$c_2(n-1)$
`    j = i - 1;`	$c_3(n-1)$
`    while (j > 0) and (A[j] > key) {`	$c_4\sum_{j=2}^{n} t_j$
`      A[j+1] = A[j]`	$c_5\sum_{j=2}^{n}(t_j - 1)$
`      j = j - 1`	$c_6\sum_{j=2}^{n}(t_j - 1)$
`    }`	0
`    A[j+1] = key`	$c_7(n-1)$
`  }`	0
`}`	

Insertion Sort

Statement	Effort
`InsertionSort(A, n) {`	
`  for i = 2 to n {`	$c_1 n$
`    key = A[i]`	$c_2(n-1)$
`    j = i - 1;`	$c_3(n-1)$
`    while (j > 0) and (A[j] > key) {`	$c_4 \sum_{j=2}^{n} t_j$
`      A[j+1] = A[j]`	$c_5 \sum_{j=2}^{n} (t_j - 1)$
`      j = j - 1`	$c_6 \sum_{j=2}^{n} (t_j - 1)$
`    }`	0
`    A[j+1] = key`	$c_7(n-1)$
`  }`	0
`}`	

$\sum_{j=2}^{n} t_j = t_2 + t_3 + \ldots + t_n$ where t_i is number of while expression evaluations for the i^{th} for loop iteration

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

- What can *T* be?

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed
  - $T = t_2 + t_3 + \ldots + t_n$

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed
  - $T = t_2 + t_3 + \ldots + t_n$

$$\sum_{j=2}^{n} t_j = t_2 + t_3 + \ldots + t_n = 1 + 1 + \ldots + 1 = n - 1$$

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n-1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed
  - $T = t_2 + t_3 + \ldots + t_n$

$$\sum_{j=2}^{n} t_j = t_2 + t_3 + \ldots + t_n = 1 + 1 + \ldots + 1 = n - 1$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4(n-1) + c_7(n-1)$$

Analysing Insertion Sort

- $T(n)$ = $c_1 n + c_2(n\text{-}1) + c_3(n\text{-}1) + \mathrm{c}_4\sum_{j=2}^{n} t_j + \mathrm{c}_5\sum_{j=2}^{n}(t_j - 1) + \mathrm{c}_6\sum_{j=2}^{n}(t_j - 1) + c_7(n\text{-}1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed
  - $T = t_2 + t_3 + \ldots + t_n$

$$\sum_{j=2}^{n} t_j = t_2 + t_3 + \ldots + t_n = 1 + 1 + \ldots + 1 = n - 1$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4(n-1) + c_7(n-1)$$

$$T(n) = (c_1 + c_2 + c_3 + c_4 + c_7) \cdot n - (c_2 + c_3 + c_4 + c_7)$$

Analysing Insertion Sort

- $T(n) = c_1 n + c_2(n\text{-}1) + c_3(n\text{-}1) + c_4\sum_{j=2}^{n} t_j + c_5\sum_{j=2}^{n}(t_j - 1) + c_6\sum_{j=2}^{n}(t_j - 1) + c_7(n\text{-}1)$

- What can *T* be?
  - **Best case** -- inner loop body never executed
  - $T = t_2 + t_3 + \ldots + t_n$

$$\sum_{j=2}^{n} t_j = t_2 + t_3 + \ldots + t_n = 1 + 1 + \ldots + 1 = n - 1$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4(n-1) + c_7(n-1)$$

$$T(n) = (c_1 + c_2 + c_3 + c_4 + c_7) \cdot n - (c_2 + c_3 + c_4 + c_7)$$

  - *T(n) = an - b*

Sum Review

Gaussian Closed Form can be defined as:

- $$\sum_{j=1}^{n} j = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}$$

Thus, we have:

$$\sum_{j=2}^{n} j = 2 + 3 + \ldots + n = \frac{n(n+1)}{2} - 1$$

Similarly, we obtain:

$$\sum_{j=2}^{n} (j-1) = \ldots = \frac{n(n+1)}{2} - n = \frac{n(n+1) - 2n}{2} = \frac{n(n-1)}{2}$$

Sum Review

Gaussian Closed Form can be defined as:

$$\sum_{j=1}^{n} j = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}$$

Thus, we have:

- $$\sum_{j=2}^{n} j = 2 + 3 + \ldots + n = \frac{n(n+1)}{2} - 1$$

Similarly, we obtain:

$$\sum_{j=2}^{n} (j-1) = \ldots = \frac{n(n+1)}{2} - n = \frac{n(n+1) - 2n}{2} = \frac{n(n-1)}{2}$$

Sum Review

Gaussian Closed Form can be defined as:

$$\sum_{j=1}^{n} j = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}$$

Thus, we have:

$$\sum_{j=2}^{n} j = 2 + 3 + \ldots + n = \frac{n(n+1)}{2} - 1$$

Similarly, we obtain:

- $$\sum_{j=2}^{n} (j-1) = \ldots = \frac{n(n+1)}{2} - n = \frac{n(n+1) - 2n}{2} = \frac{n(n-1)}{2}$$

Analysing Insertion Sort

- **Worst case** -- inner loop body executed for all previous elements

$$\sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1 \qquad \sum_{j=2}^{n} (j-1) = \frac{n(n-1)}{2}$$

Analysing Insertion Sort

- **Worst case** -- inner loop body executed for all previous elements

$$\sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1 \qquad \sum_{j=2}^{n} (j-1) = \frac{n(n-1)}{2}$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\left(\frac{n(n+1)}{2} - 1\right) + c_5\left(\frac{n(n-1)}{2}\right) + c_6\left(\frac{n(n-1)}{2}\right)$$

$$+ c_7(n-1) = \left(\frac{c_4}{2} + \frac{c_5}{2} + \frac{c_6}{2}\right)n^2 + \left(c_1 + c_2 + c_3 + \frac{c_4}{2} - \frac{c_5}{2} - \frac{c_6}{2} + c_7\right)n - (c_2 + c_3 + c_4 + c_7)$$

Analysing Insertion Sort

- **Worst case** -- inner loop body executed for all previous elements

$$\sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1 \qquad \sum_{j=2}^{n} (j-1) = \frac{n(n-1)}{2}$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\left(\frac{n(n+1)}{2} - 1\right) + c_5\left(\frac{n(n-1)}{2}\right) + c_6\left(\frac{n(n-1)}{2}\right)$$

$$+ c_7(n-1) = \left(\frac{c_4}{2} + \frac{c_5}{2} + \frac{c_6}{2}\right)n^2 + \left(c_1 + c_2 + c_3 + \frac{c_4}{2} - \frac{c_5}{2} - \frac{c_6}{2} + c_7\right)n - (c_2 + c_3 + c_4 + c_7)$$

- o $T(n) = an^2 + bn - c$

Analysing Insertion Sort

- **Worst case** -- inner loop body executed for all previous elements

$$\sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1 \qquad \sum_{j=2}^{n} (j-1) = \frac{n(n-1)}{2}$$

$$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4\left(\frac{n(n+1)}{2} - 1\right) + c_5\left(\frac{n(n-1)}{2}\right) + c_6\left(\frac{n(n-1)}{2}\right)$$

$$+ c_7(n-1) = \left(\frac{c_4}{2} + \frac{c_5}{2} + \frac{c_6}{2}\right)n^2 + \left(c_1 + c_2 + c_3 + \frac{c_4}{2} - \frac{c_5}{2} - \frac{c_6}{2} + c_7\right)n - (c_2 + c_3 + c_4 + c_7)$$

  - o $T(n) = an^2 + bn - c$
- **Average case**
  - o ???