Insertion Sort

Author: Wadd Franklin

Insertion Sort

0	1	2	3	4	5
9	2	7	0	4	6

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

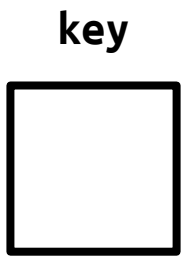
Insertion Sort

0	1	2	3	4	5
9	2	7	0	4	6

size = 6

```
void insertionSort(int a*, int size) {
    int i, j, key;
    for (i = 1 ; i < size ; i++) {
        key = a[i];
        j = i - 1;
        while ((j >= 0) && (a[j] > key)) {
            a[j + 1] = a[j];
            j--;
        }
        a[j + 1] = key;
    }
}
```

Insertion Sort

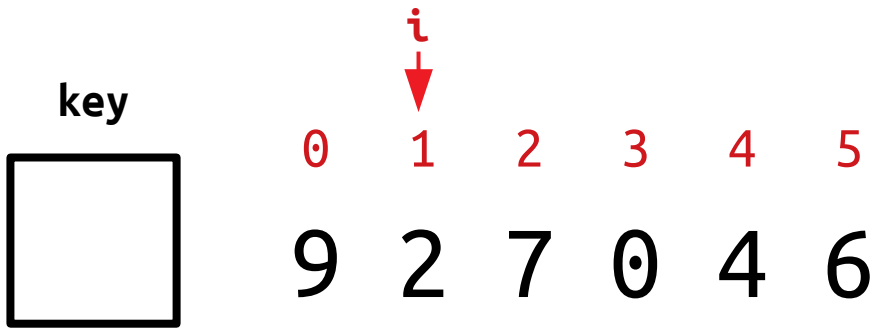


0	1	2	3	4	5
9	2	7	0	4	6

```
void insertionSort(int a*, int size) {
    int i, j, key;
    for (i = 1 ; i < size ; i++) {
        key = a[i];
        j = i - 1;
        while ((j >= 0) && (a[j] > key)) {
            a[j + 1] = a[j];
            j--;
        }
        a[j + 1] = key;
    }
}
```

size = 6
i =
j =

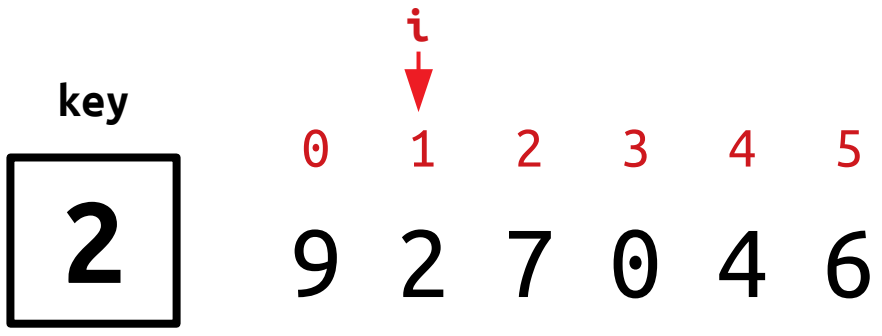
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 1
j =

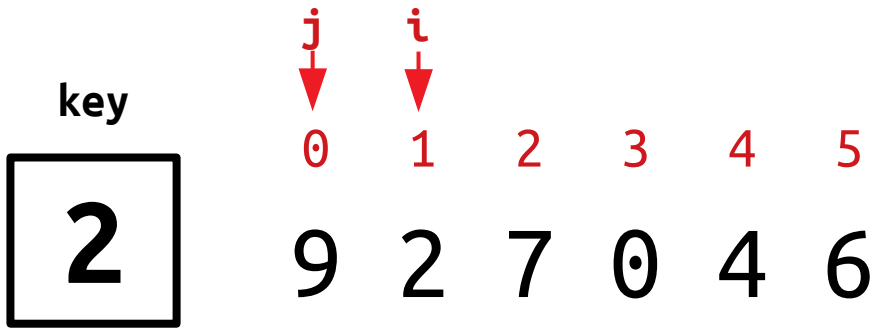
Insertion Sort




```
size = 6
i = 1
j =
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

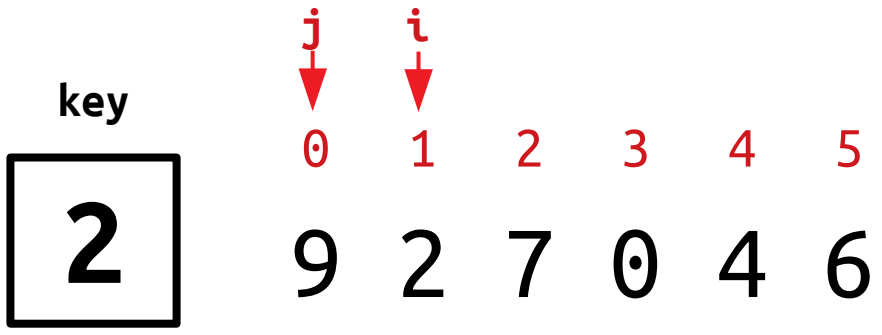
Insertion Sort




```
size = 6
i = 1
j = 0
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

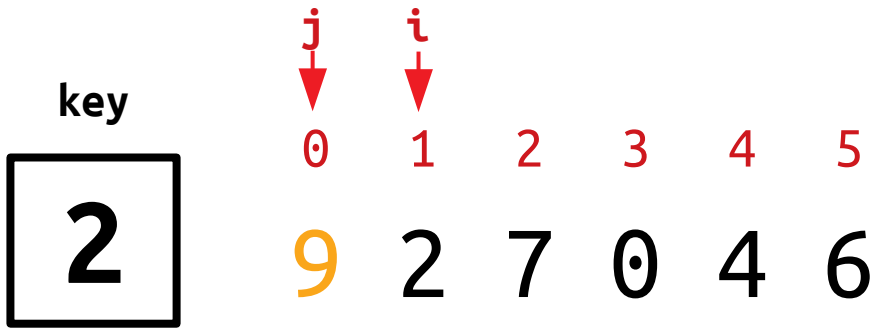
Insertion Sort




```
size = 6
i = 1
j = 0
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

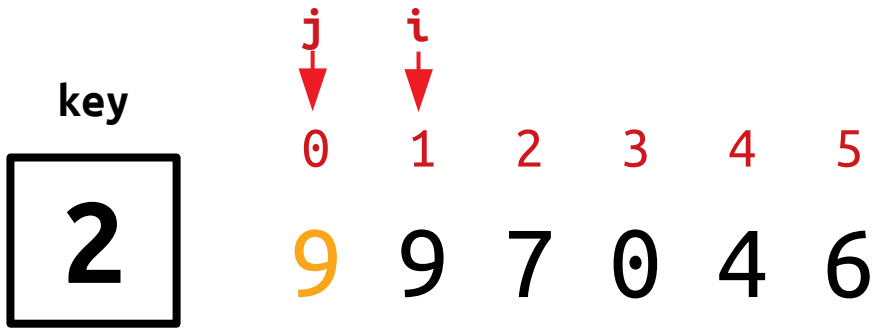
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 1
j = 0

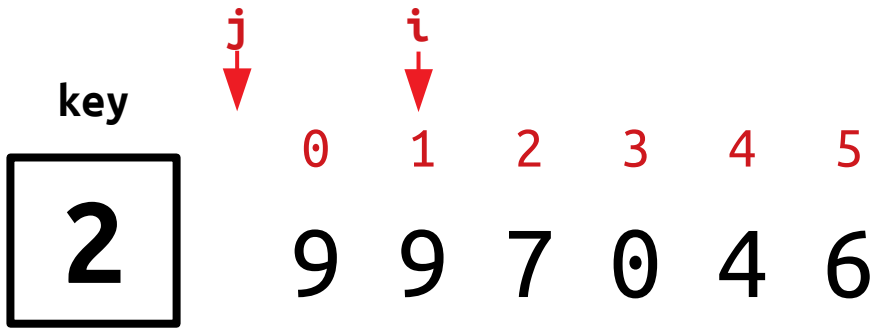
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 1
j = 0

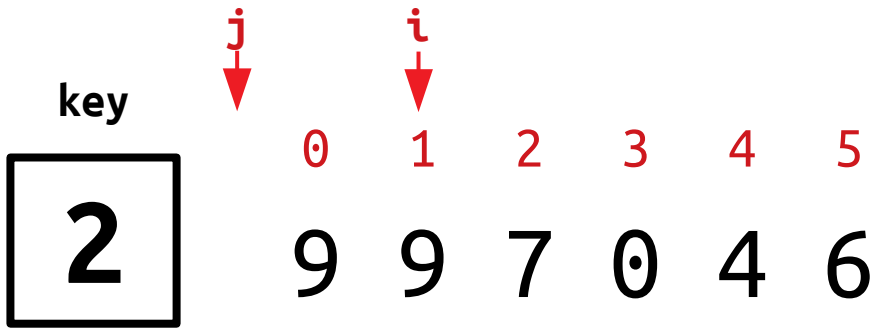
Insertion Sort




```
size = 6
i = 1
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

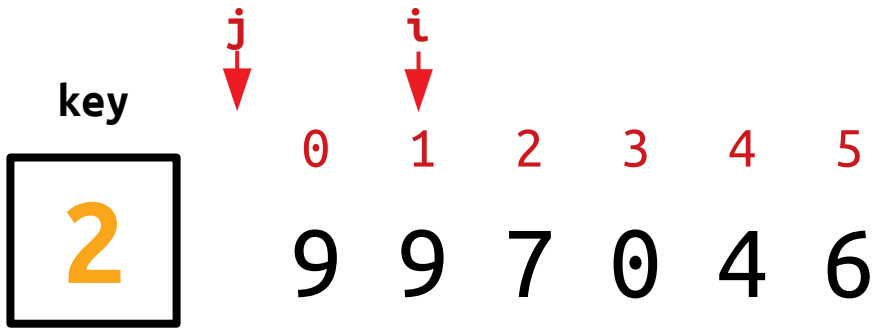
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 1
j = -1

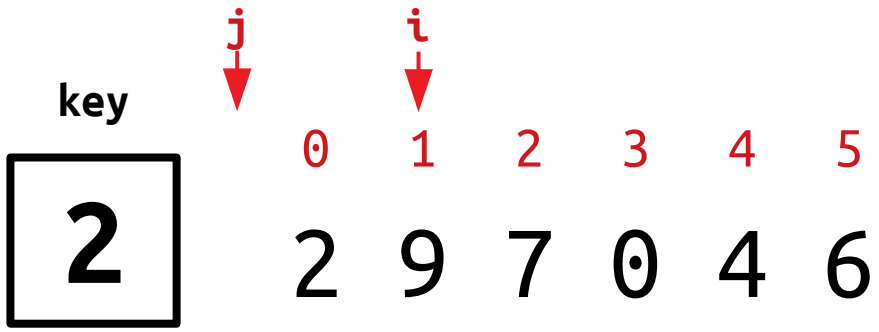
Insertion Sort




```
size = 6
i = 1
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

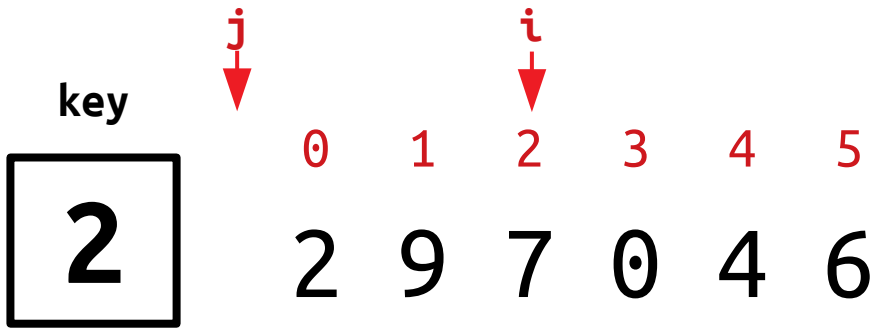
Insertion Sort




```
size = 6
i = 1
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

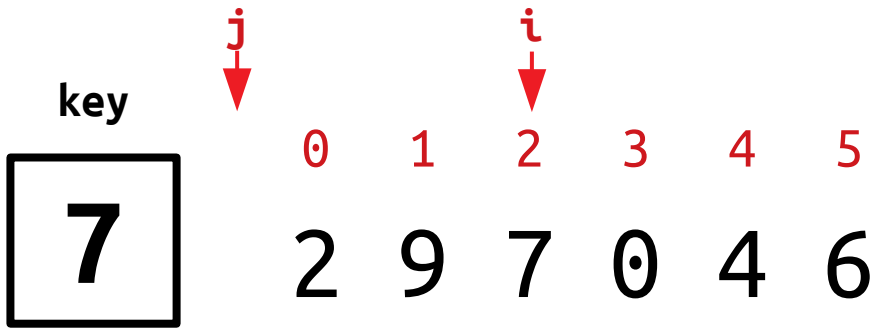
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 2
j = -1

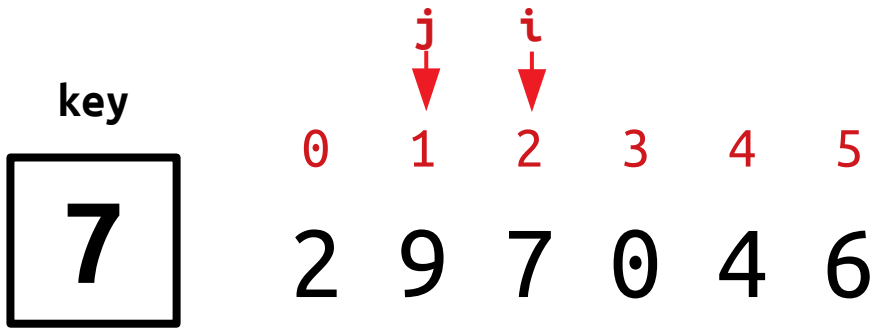
Insertion Sort




```
size = 6
i = 2
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

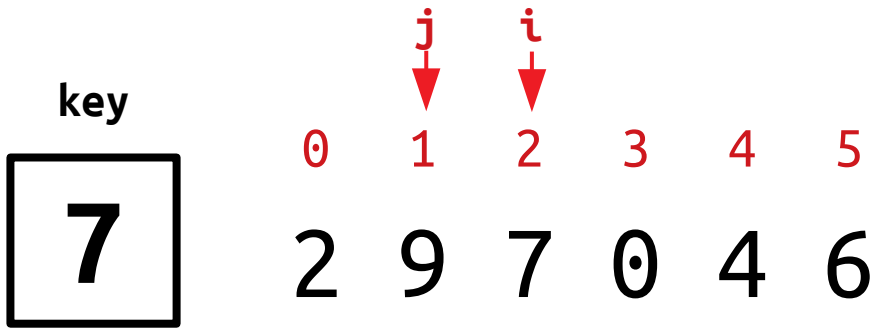
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 2
j = 1

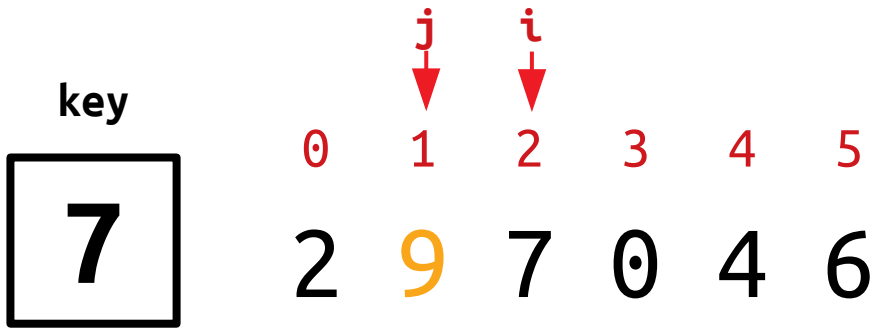
Insertion Sort




```
size = 6
i = 2
j = 1
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

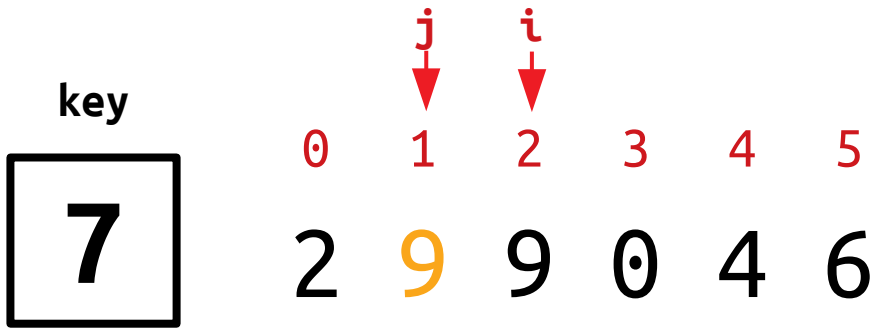
Insertion Sort




```
size = 6
i = 2
j = 1
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

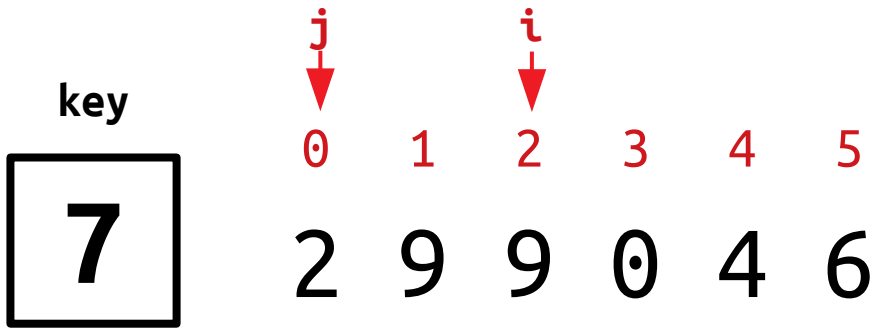
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 2
j = 1

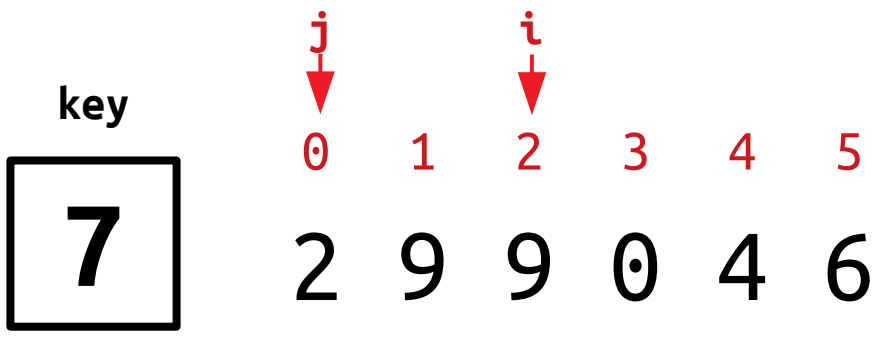
Insertion Sort




```
size = 6
i = 2
j = 0
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

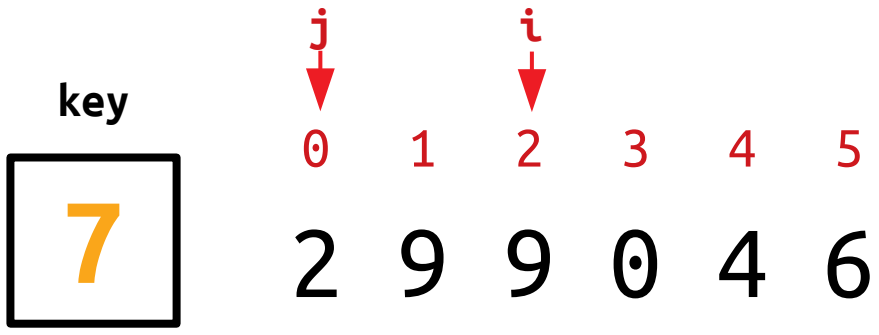
Insertion Sort




```
size = 6
i = 2
j = 0
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

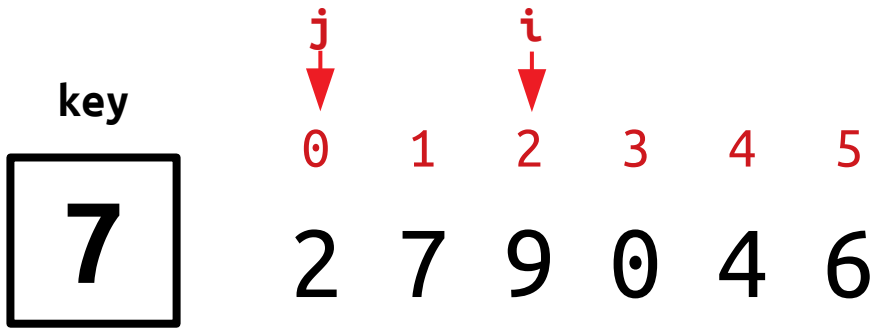
Insertion Sort




```
size = 6
i = 2
j = 0
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

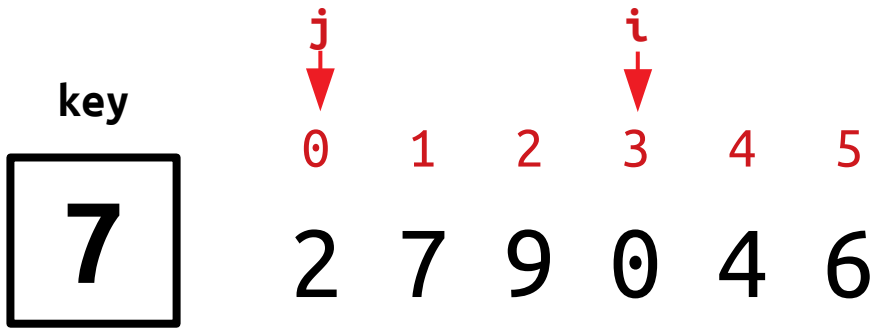
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 2
j = 0

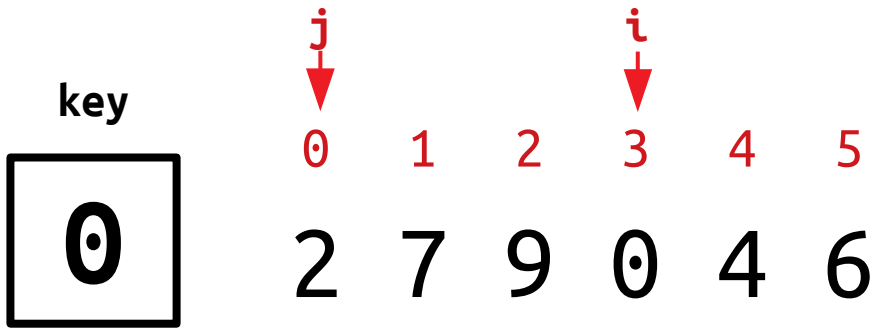
Insertion Sort




```
size = 6
i = 3
j = 0
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

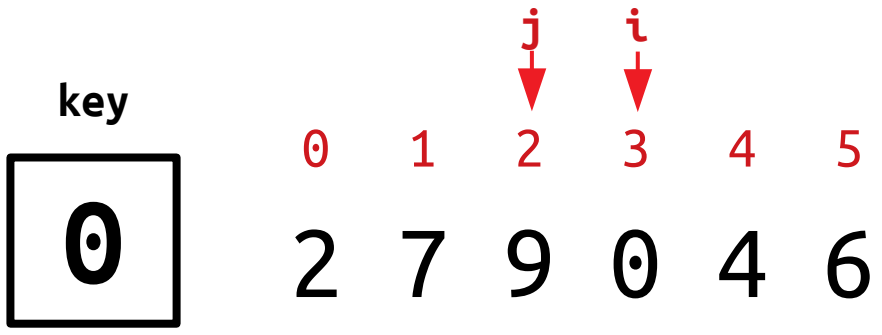
Insertion Sort




```
size = 6
i = 3
j = 0
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

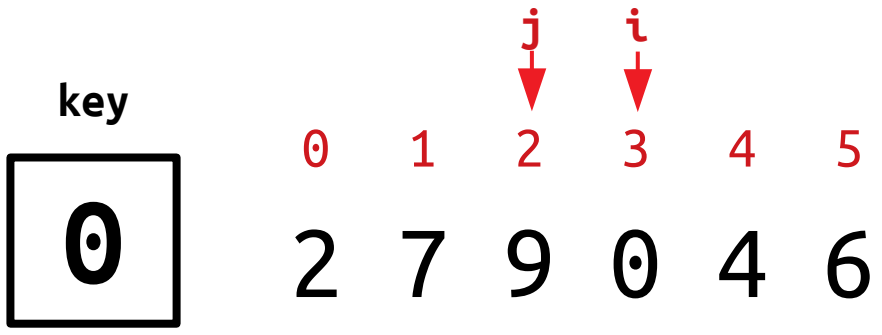
Insertion Sort




```
size = 6
i = 3
j = 2
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

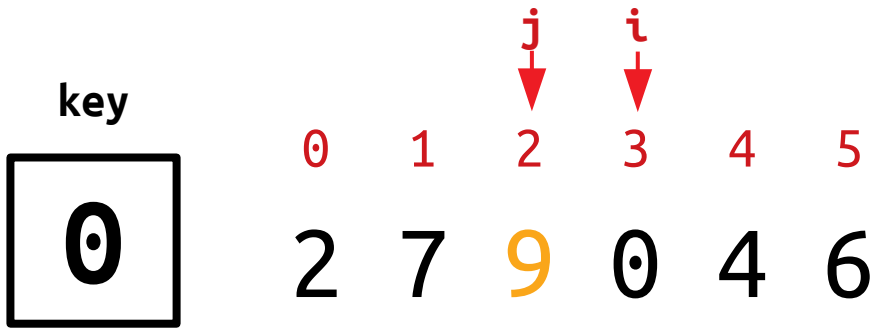
Insertion Sort




```
size = 6
i = 3
j = 2
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

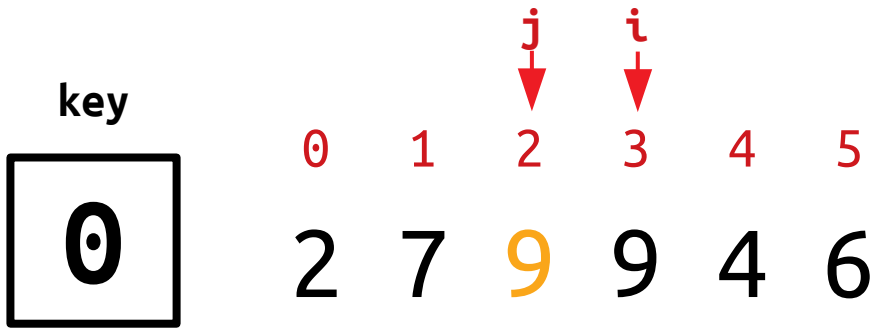
Insertion Sort




```
size = 6
i = 3
j = 2
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

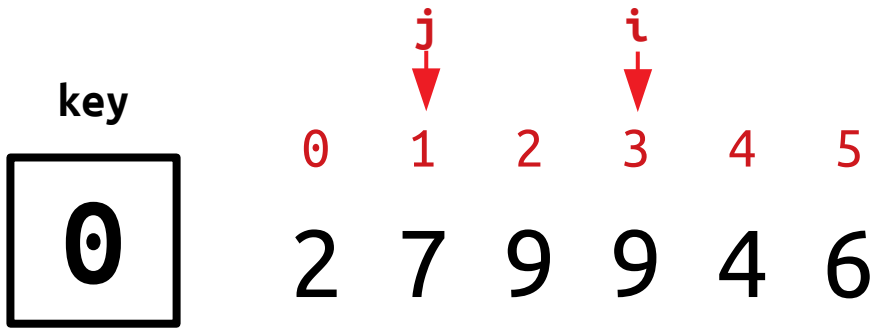
Insertion Sort




```
size = 6
i = 3
j = 2
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

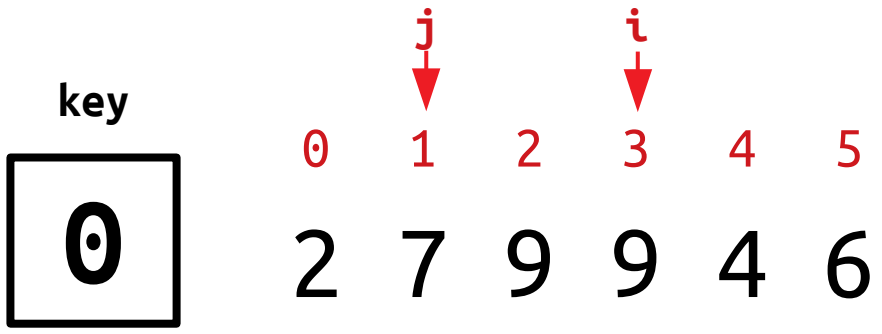
Insertion Sort




```
size = 6
i = 3
j = 1
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

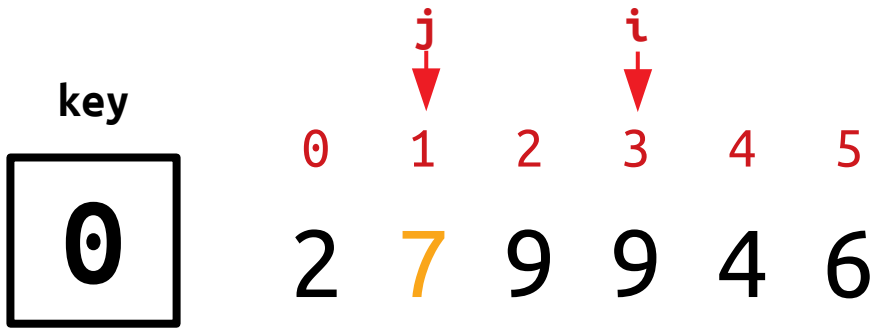
Insertion Sort




```
size = 6
i = 3
j = 1
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

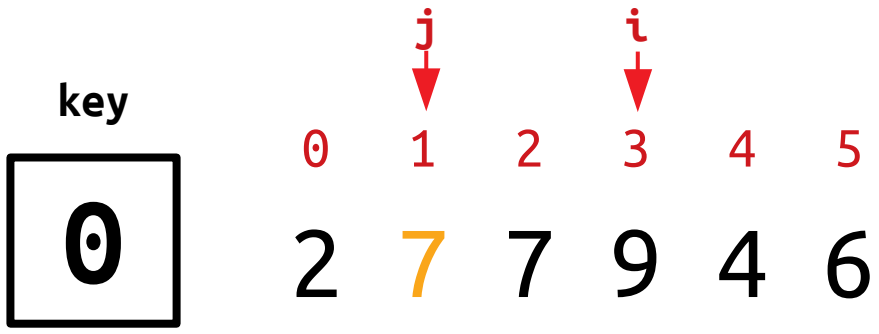
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 3
j = 1

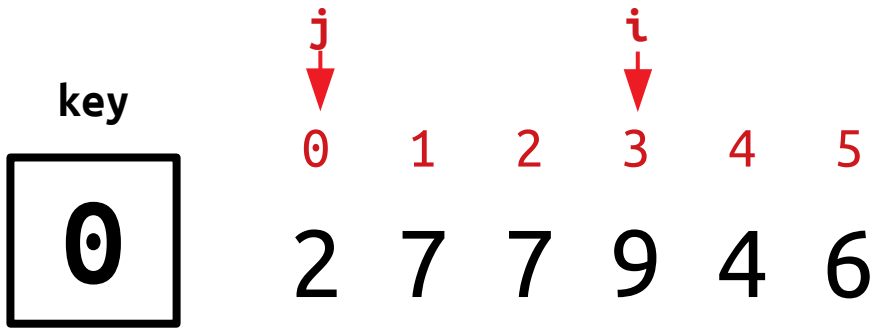
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 3
j = 1

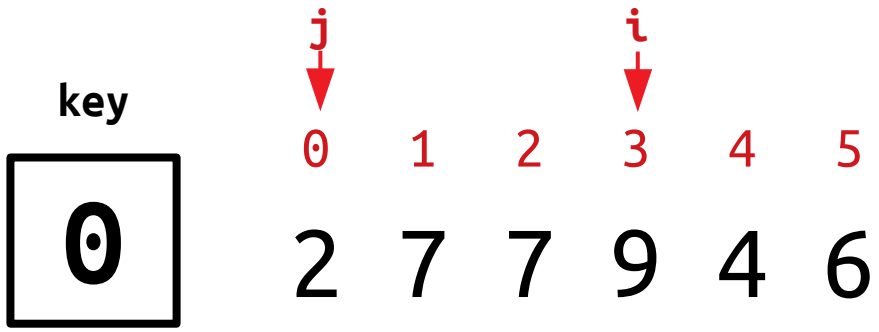
Insertion Sort




```
size = 6
i = 3
j = 0
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

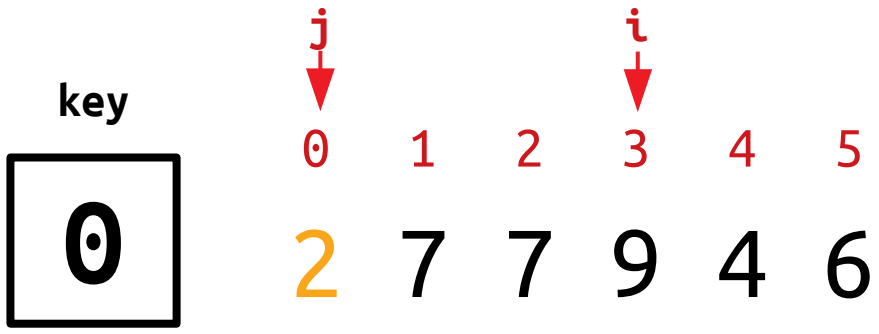
Insertion Sort




```
size = 6
i = 3
j = 0
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

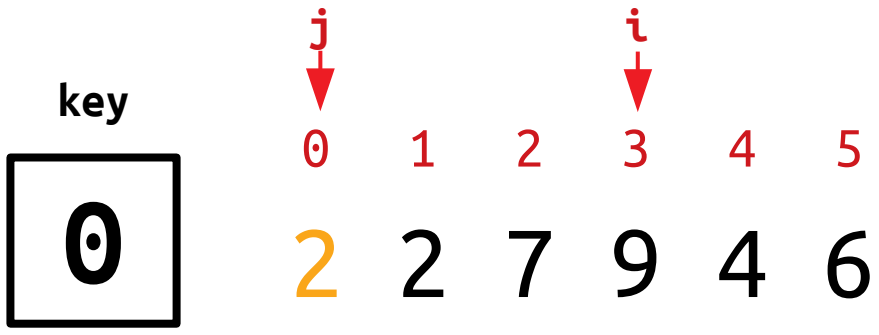
Insertion Sort




```
size = 6
i = 3
j = 0
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 3
j = 0

Insertion Sort


```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 3
j = -1

Insertion Sort


```
size = 6
i = 3
j = -1
```

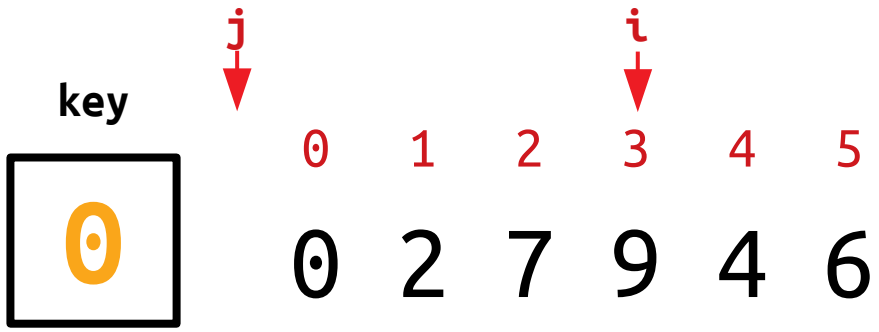
```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

Insertion Sort


```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 3
j = -1

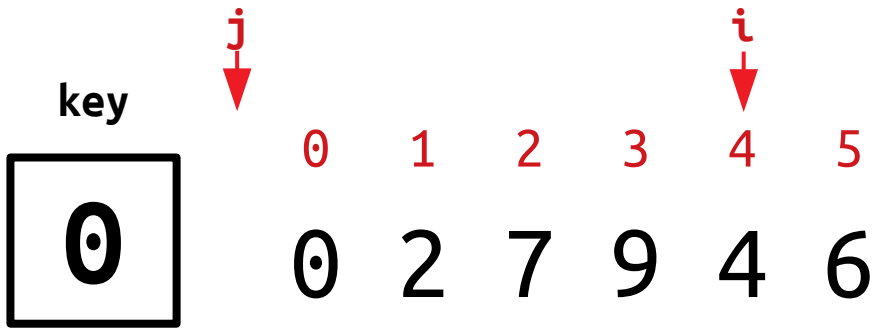
Insertion Sort




```
size = 6
i = 3
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

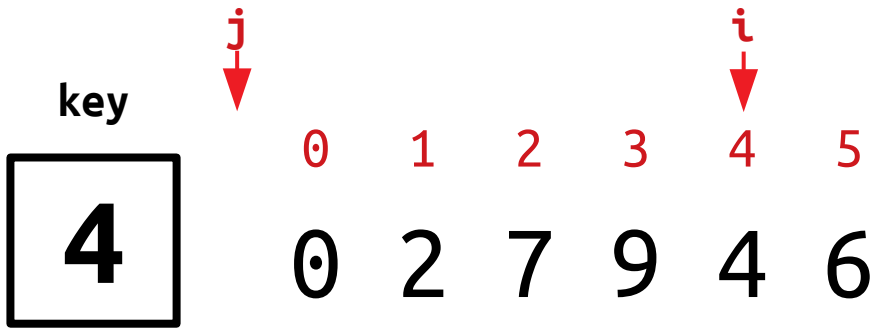
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = -1

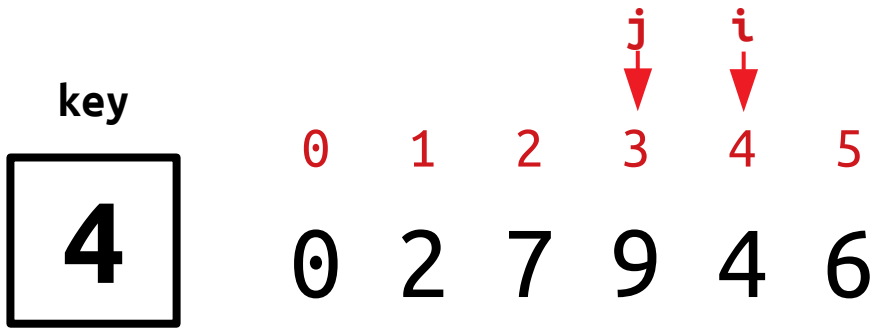
Insertion Sort




```
size = 6
i = 4
j = -1
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

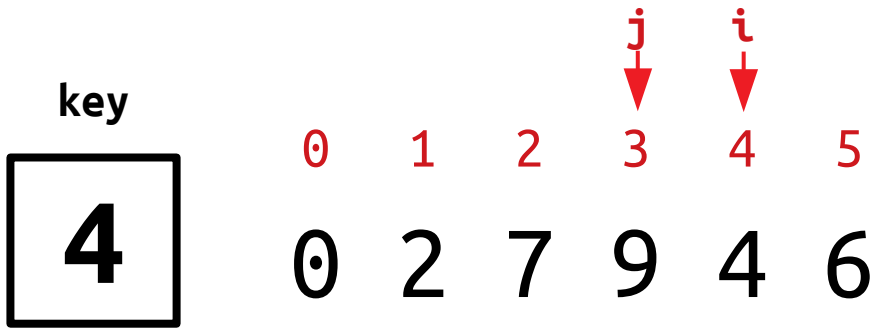
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 3

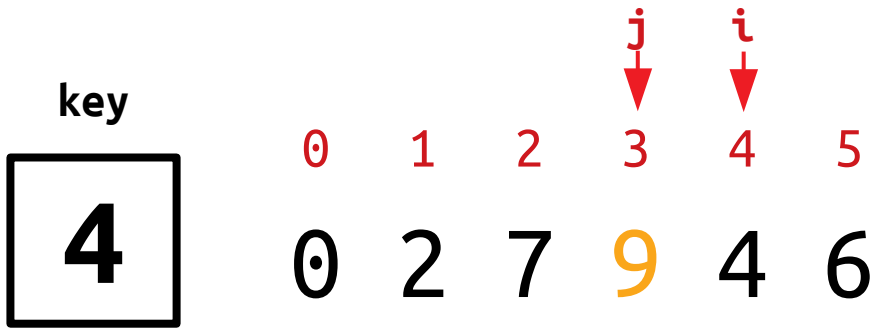
Insertion Sort




```
size = 6
i = 4
j = 3
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

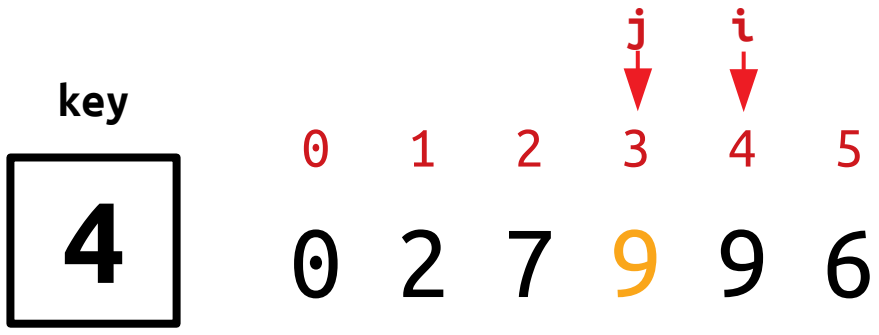
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 3

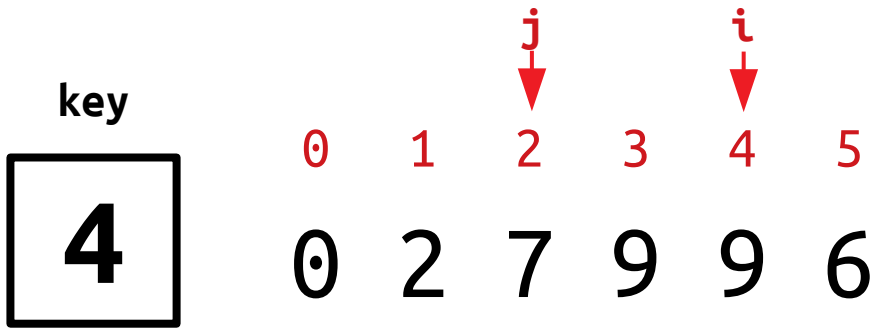
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 3

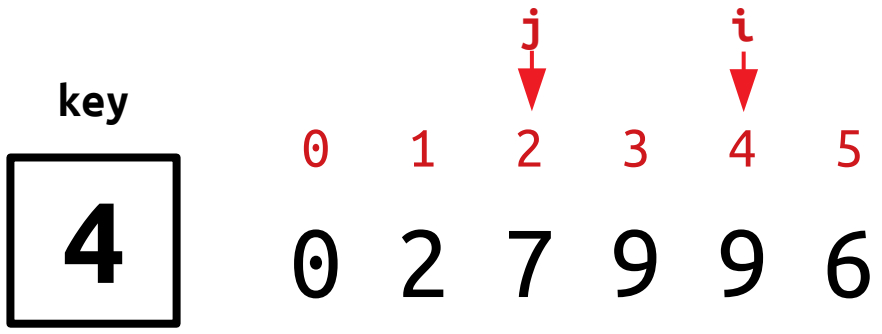
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 2

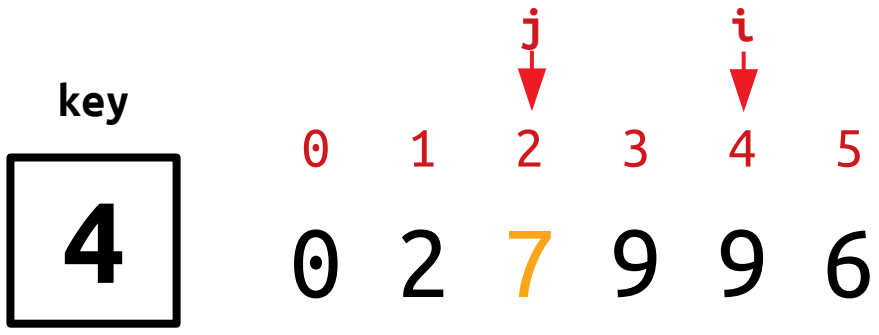
Insertion Sort




```
size = 6
i = 4
j = 2
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

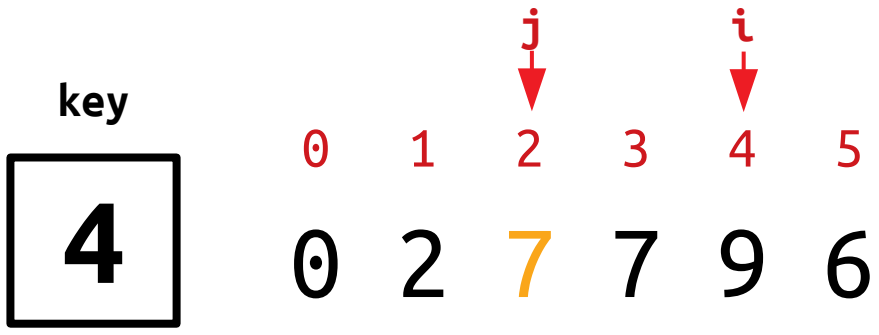
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 2

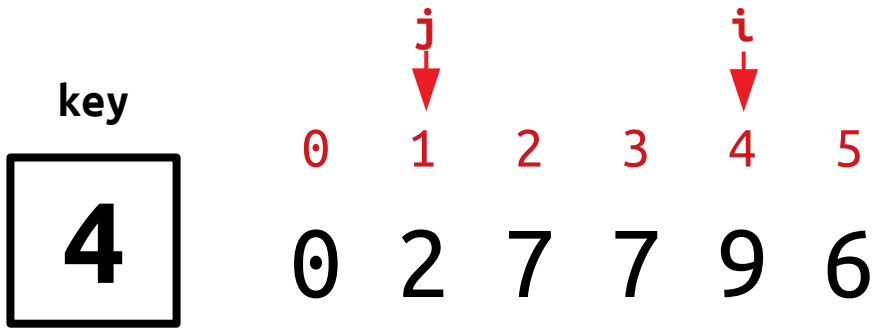
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 2

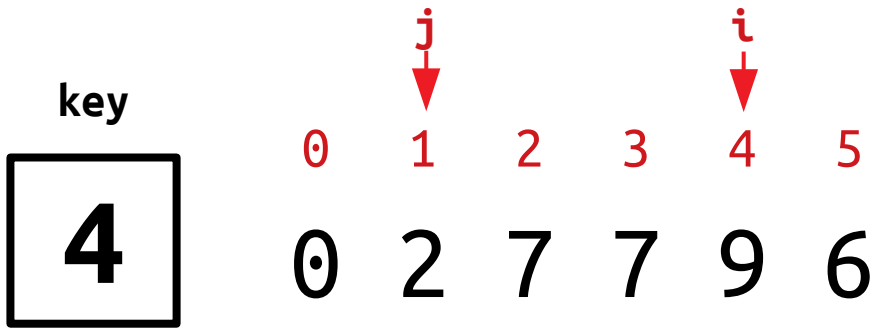
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 1

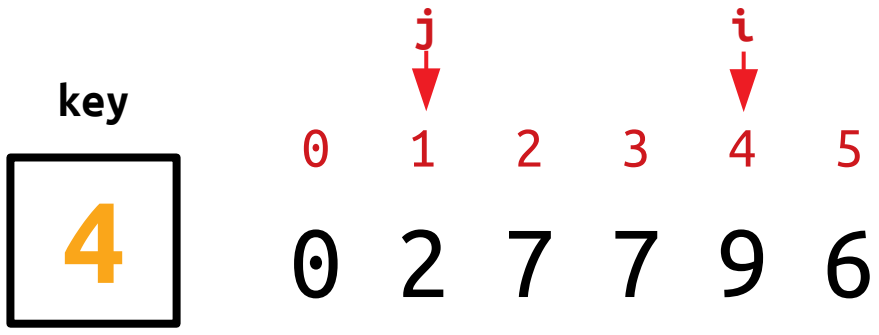
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 1

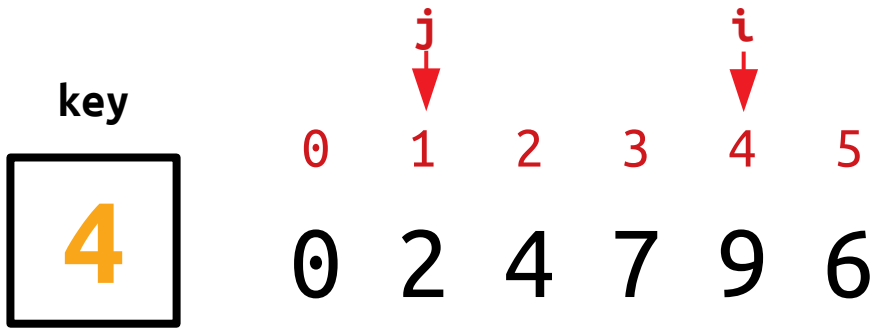
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 1

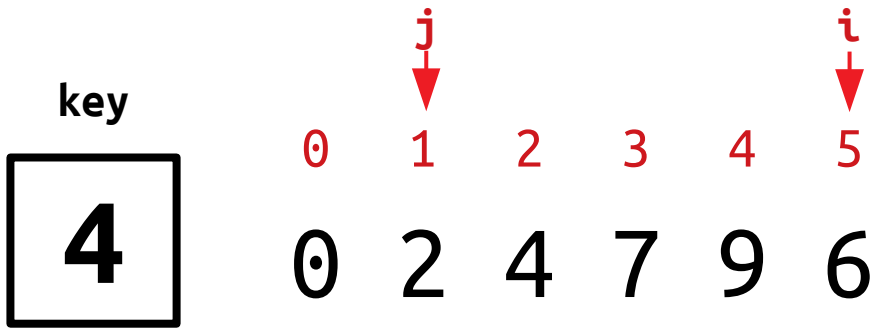
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 4
j = 1

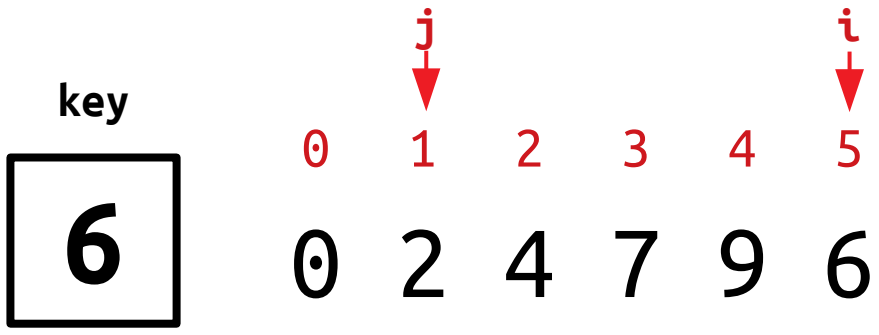
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 1

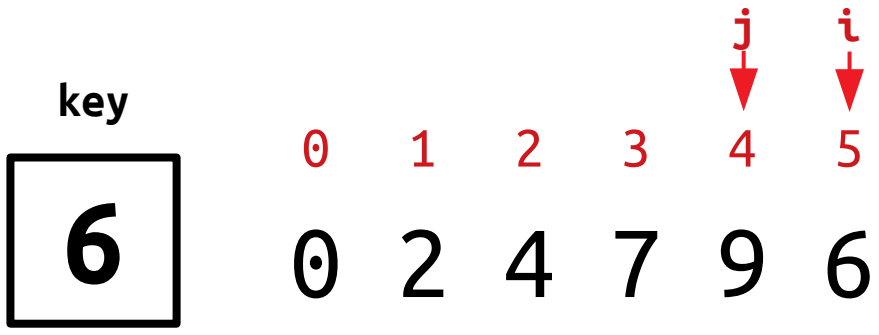
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 1

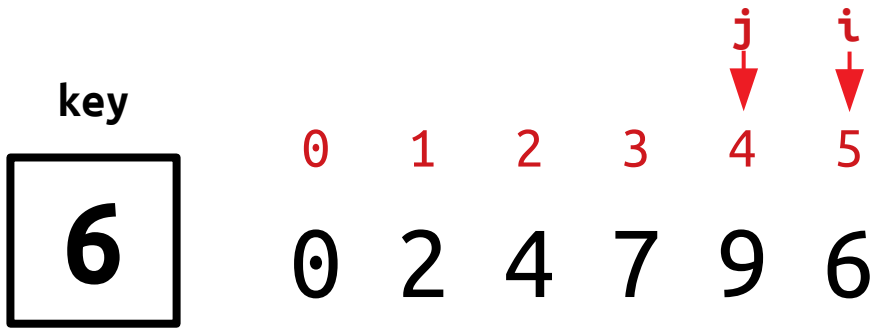
Insertion Sort




```
size = 6
i = 5
j = 4
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

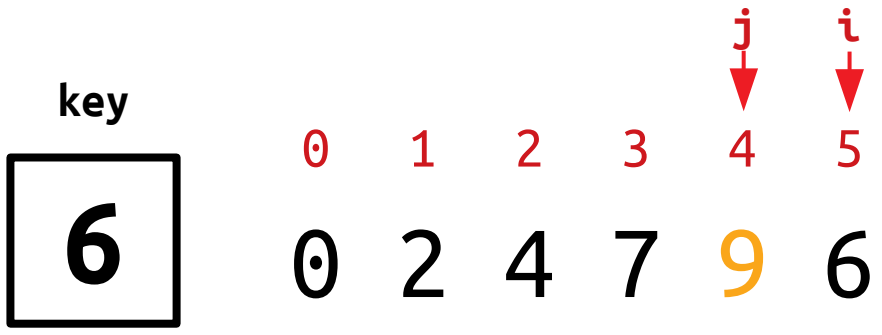
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 4

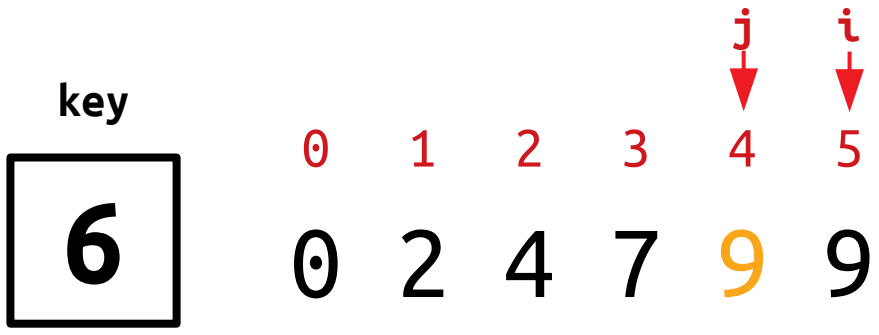
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 4

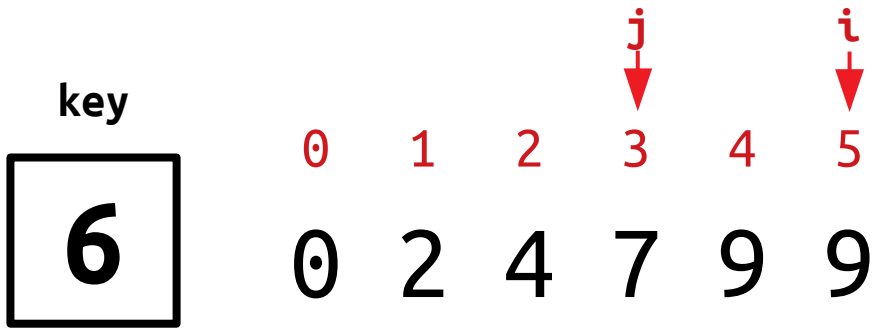
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 4

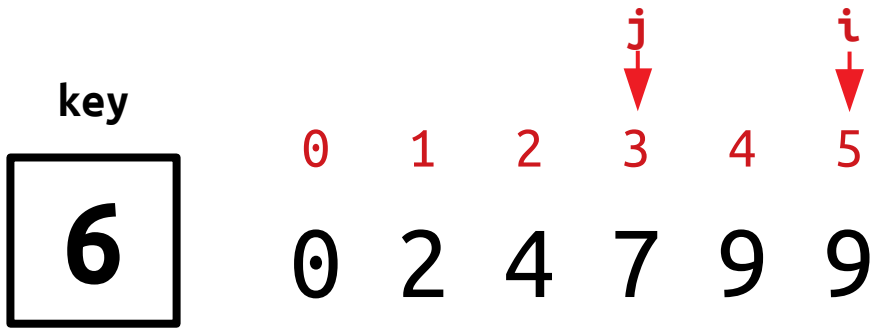
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 3

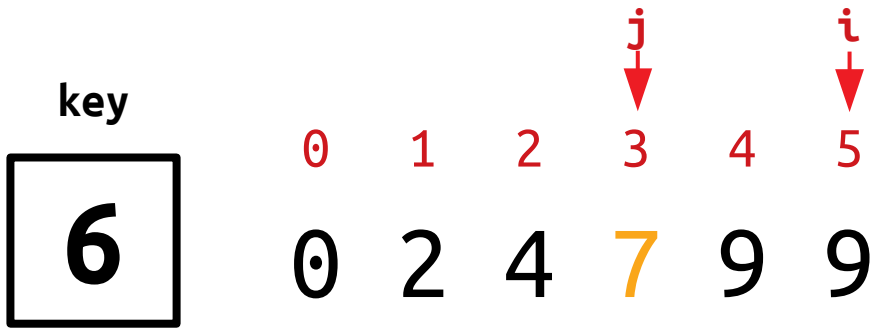
Insertion Sort




```
size = 6
i = 5
j = 3
```

```c
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

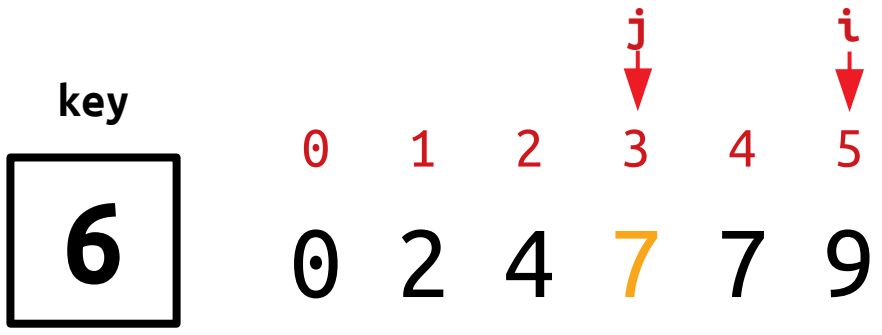
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 3

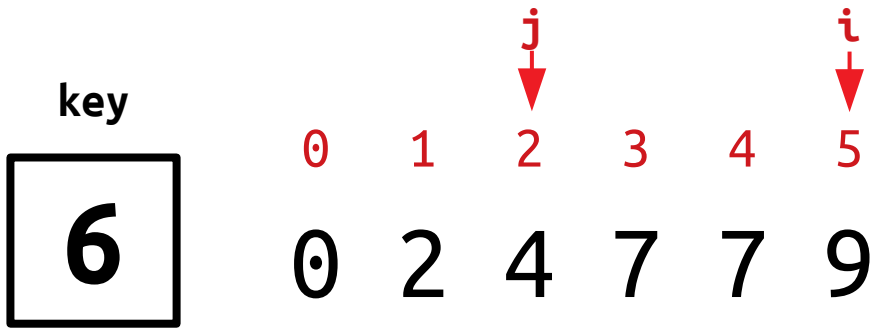
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 3

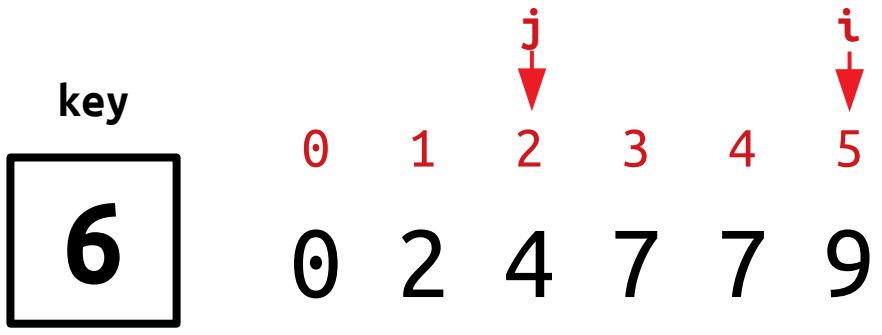
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 2

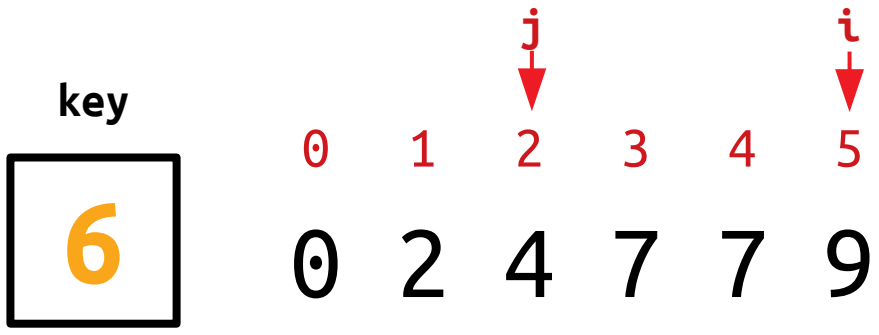
Insertion Sort




```
size = 6
i = 5
j = 2
```

```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

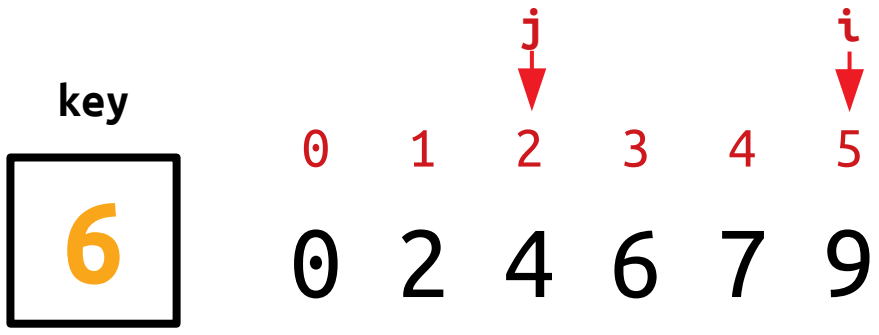
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 2

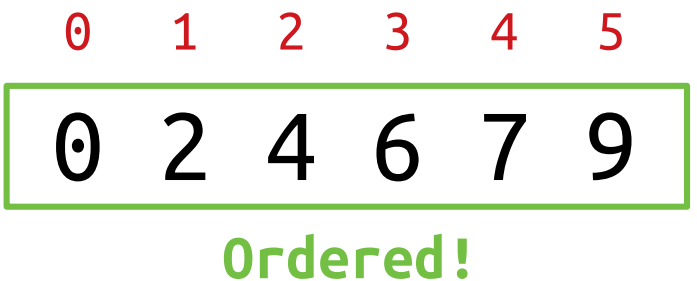
Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```

size = 6
i = 5
j = 2

Insertion Sort




```
void insertionSort(int a*, int size) {

    int i, j, key;

    for (i = 1 ; i < size ; i++) {

        key = a[i];
        j = i - 1;

        while ((j >= 0) && (a[j] > key)) {

            a[j + 1] = a[j];
            j--;
        }

        a[j + 1] = key;
    }
}
```