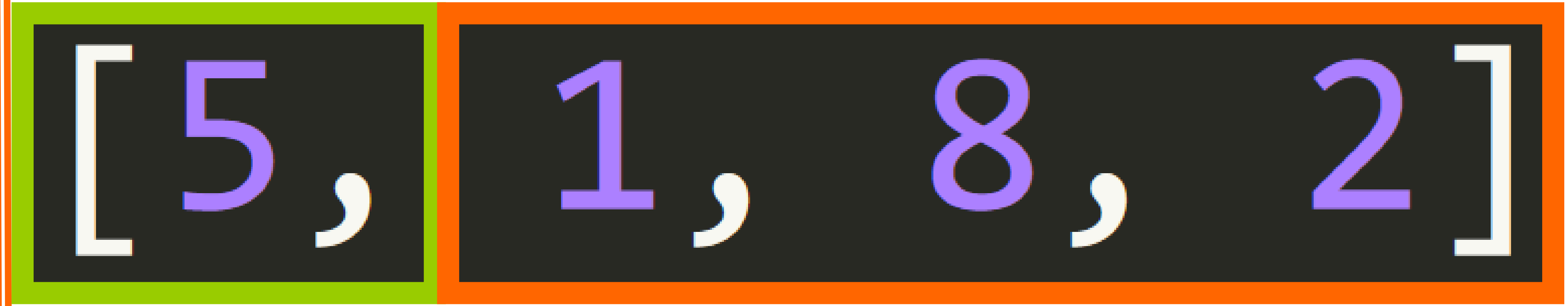
Algorithm

Insertion Sort

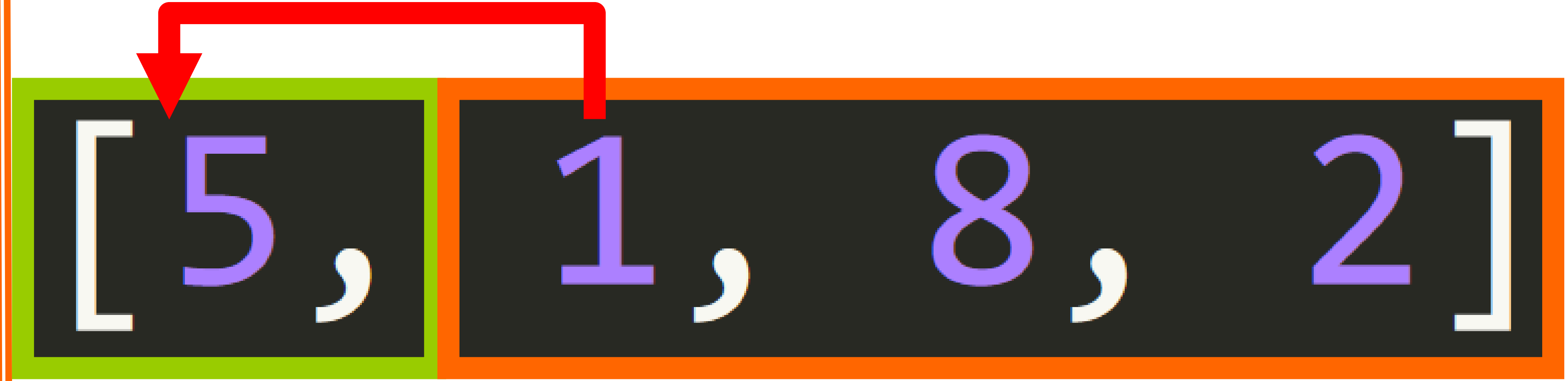
Example | Visual

Insertion Sort



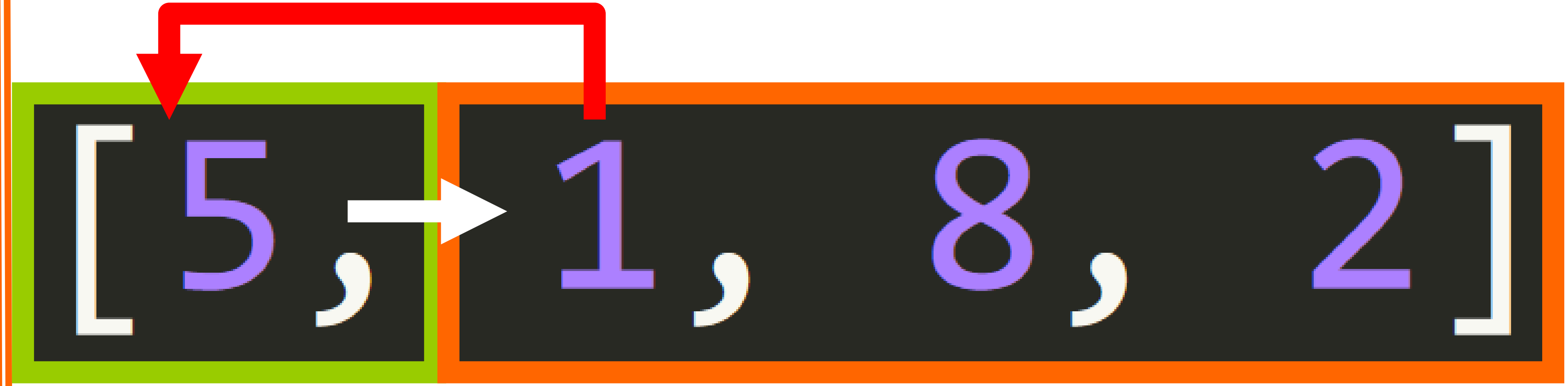
```
[5, 1, 8, 2]
```

Insertion Sort



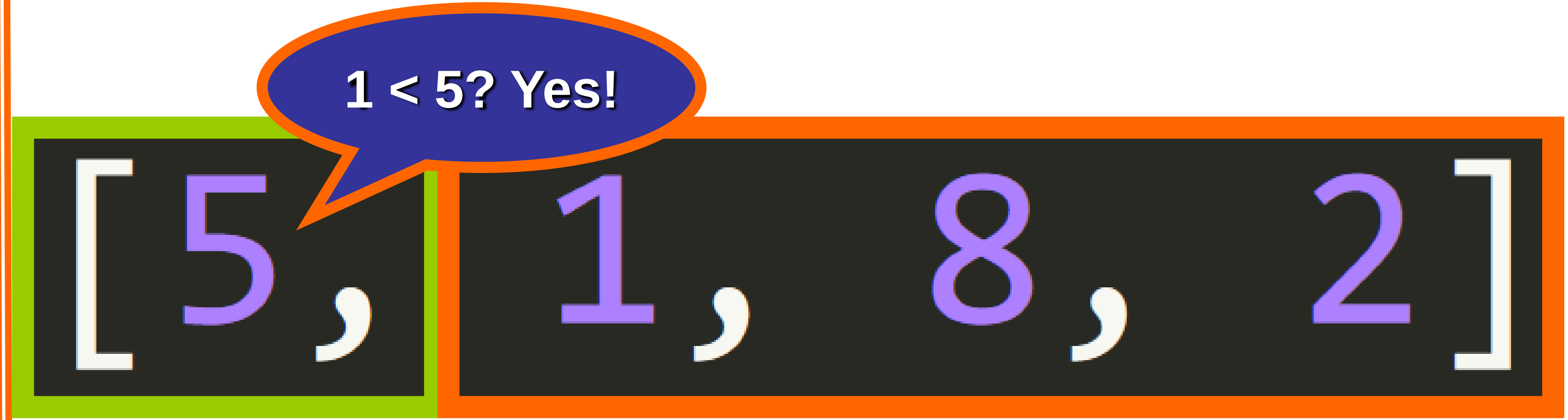
elem_selected: 1

Insertion Sort



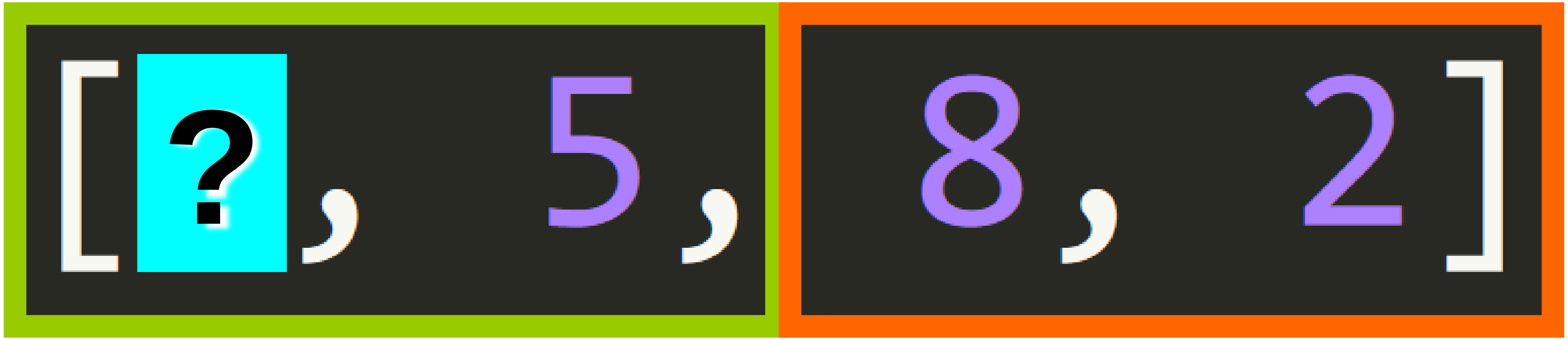
elem_selected: 1

Insertion Sort



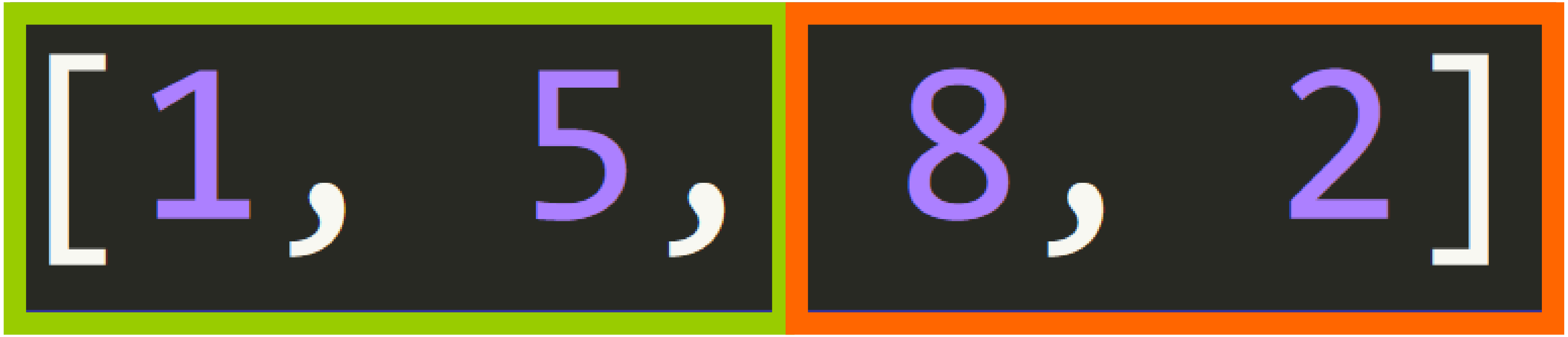
elem_selected: 1

Insertion Sort



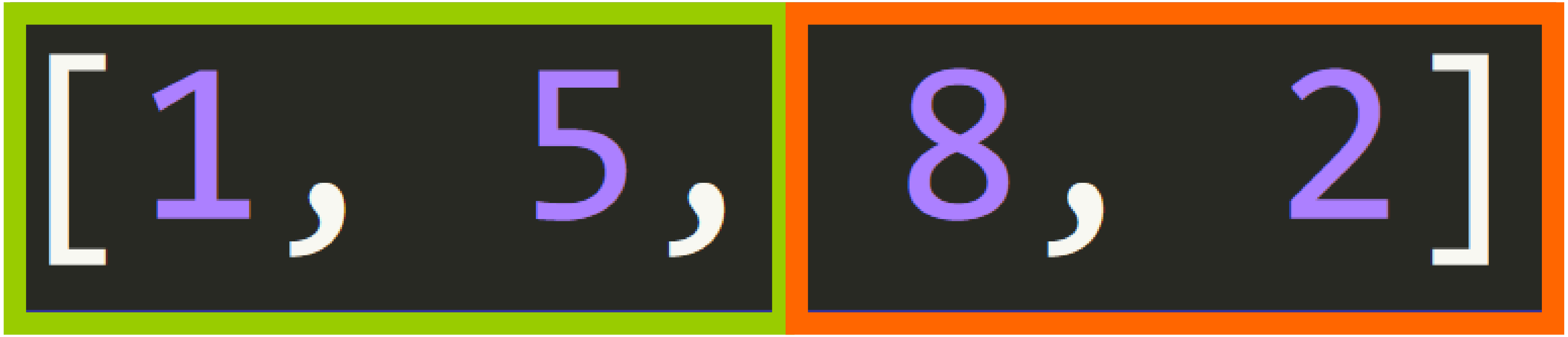
elem_selected: 1

Insertion Sort



```
[1, 5, 8, 2]
```

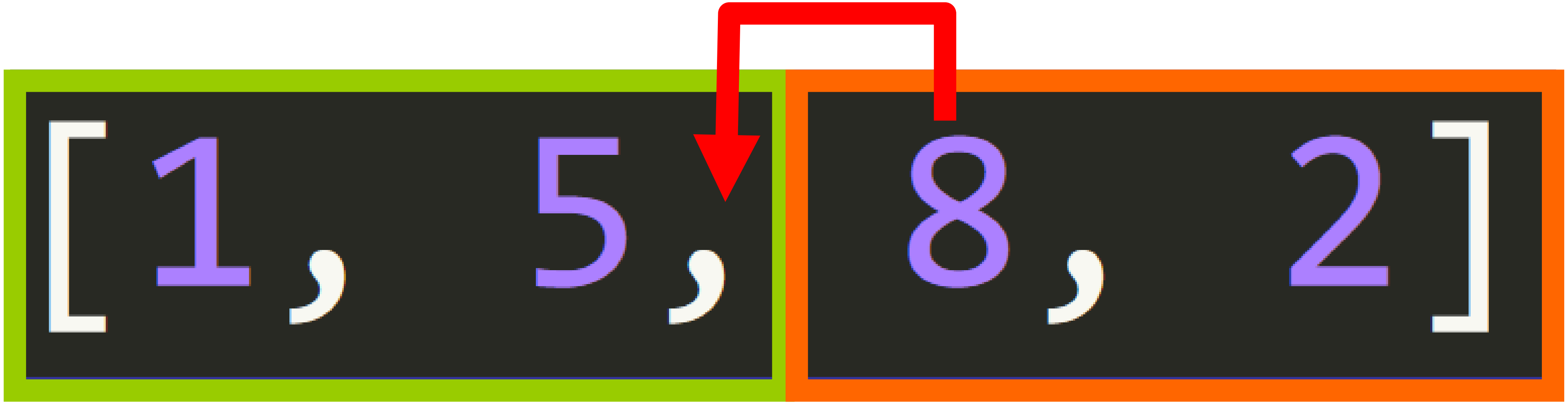
Insertion Sort



```
[1, 5, 8, 2]
```

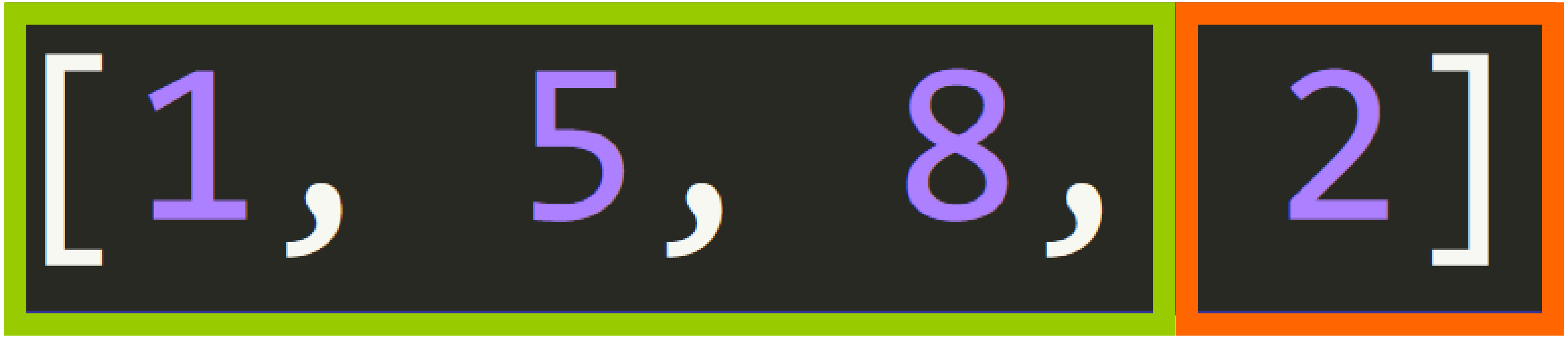
elem_selected: 8

Insertion Sort



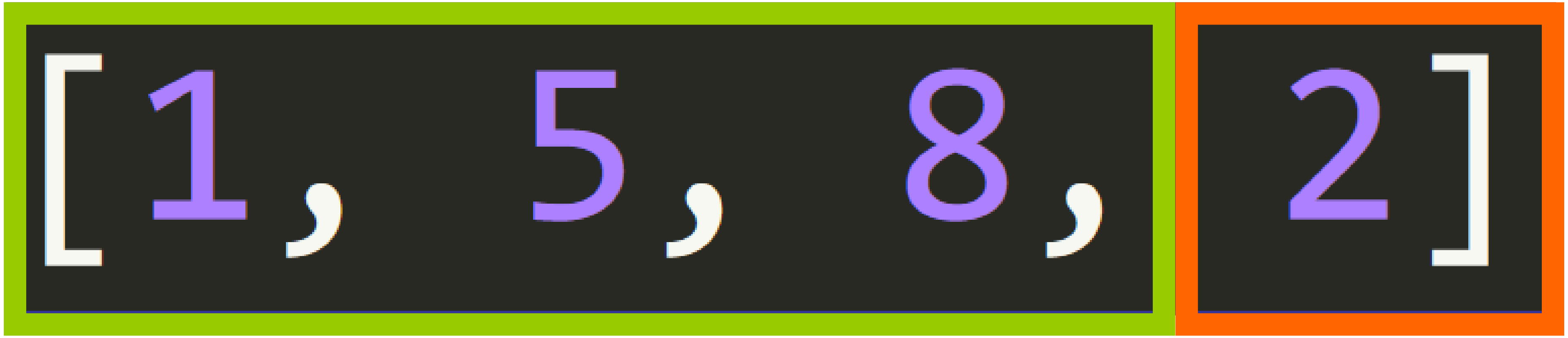
elem_selected: 8

Insertion Sort



```
[1, 5, 8, 2]
```

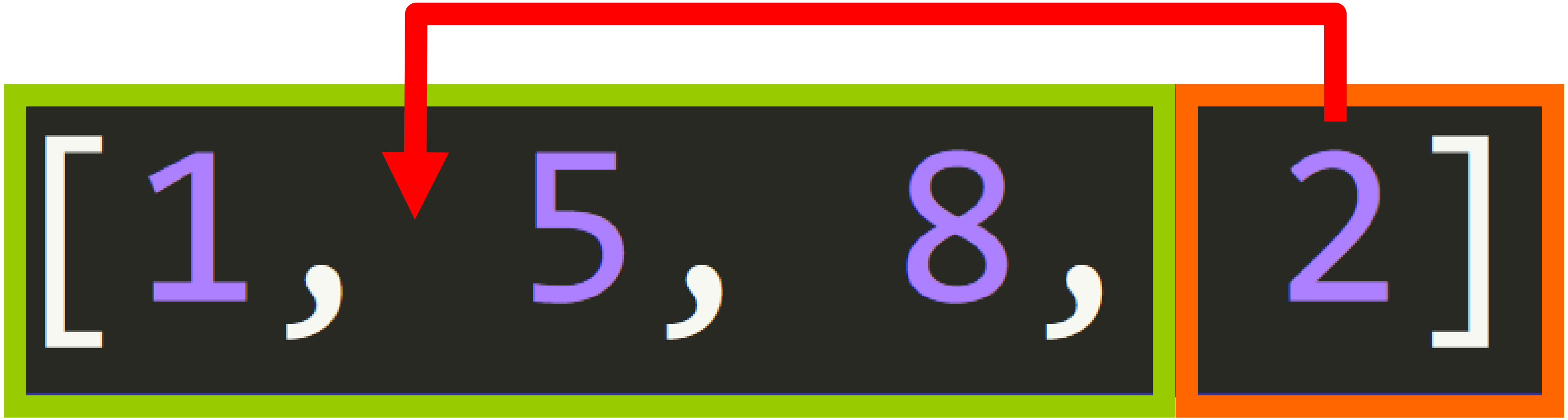
Insertion Sort



[1, 5, 8, 2]

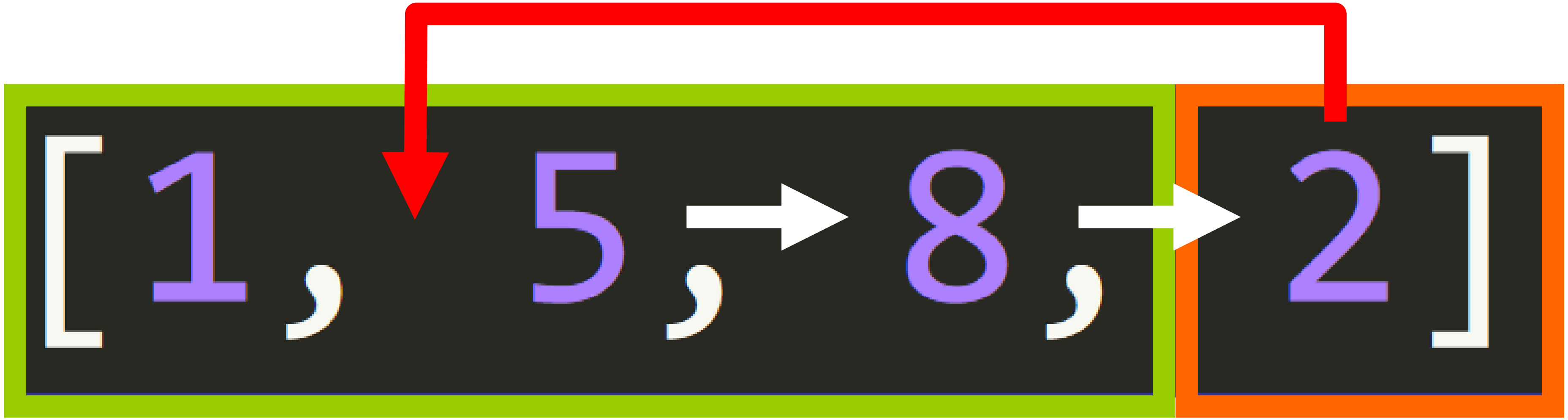
elem_selected: 2

Insertion Sort



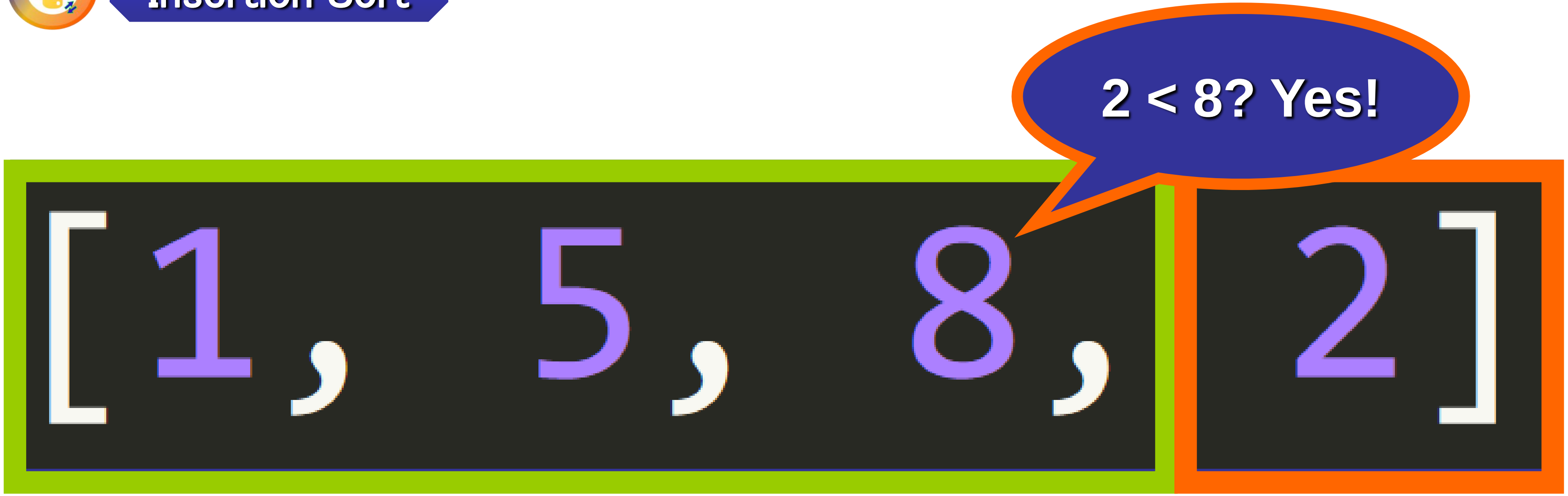
elem_selected: 2

Insertion Sort



elem_selected: 2

Insertion Sort



```
[1, 5, 8, 2]
```

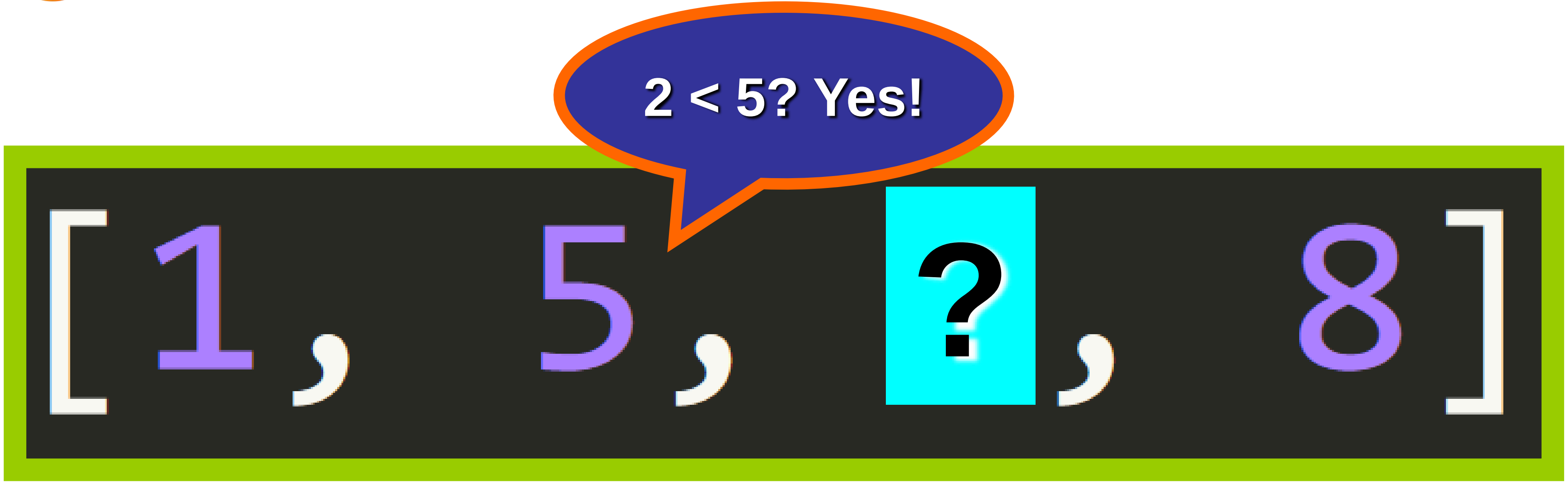
elem_selected: 2

Insertion Sort

```
[1, 5, ?, 8]
```

elem_selected: 2

Insertion Sort



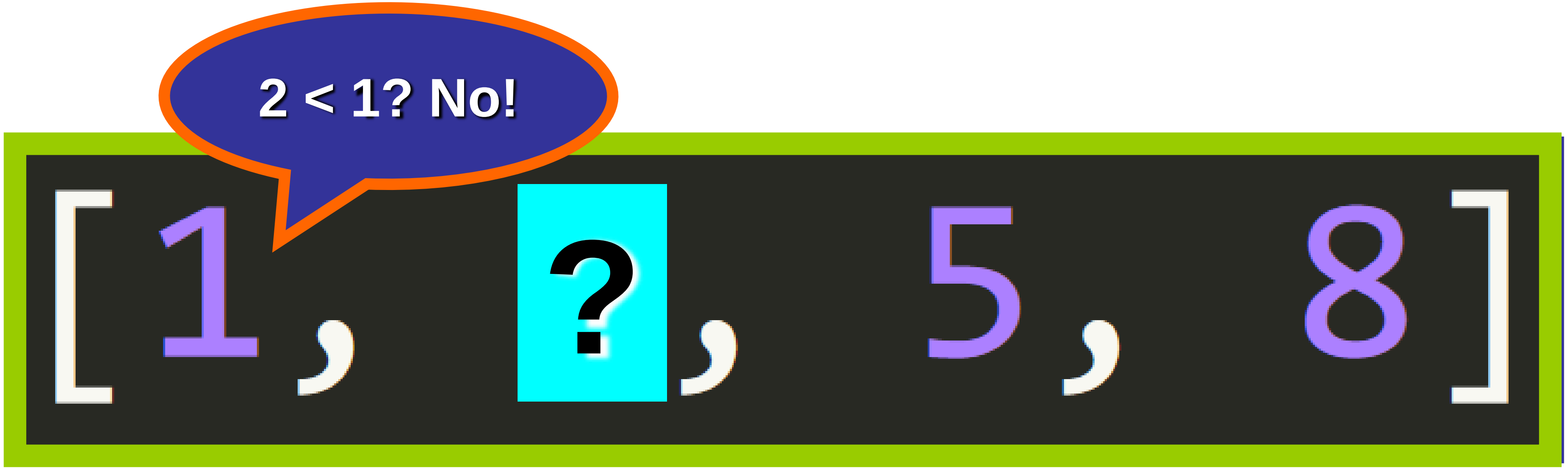
elem_selected: 2

Insertion Sort

```
[1, ?, 5, 8]
```

elem_selected: 2

Insertion Sort



elem_selected: 2

Insertion Sort

Sorted!

```
[1, 2, 5, 8]
```

Time to Practice!

Algorithm

Insertion Sort

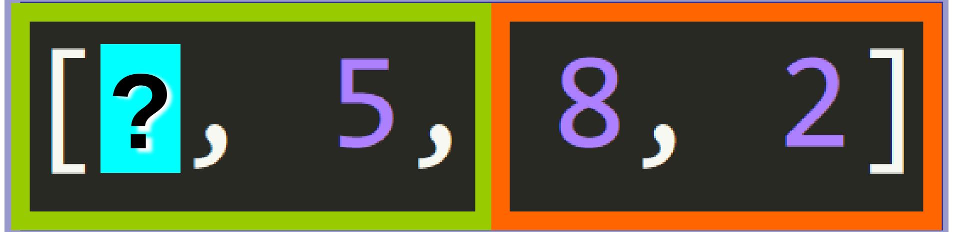
Example | Code

[5, 1, 8, 2]

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 1

```
>>> insertion_sort([5, 1, 8, 2])

=========> Starting Insertion Sort

---> Outer loop. Iteration #1 (i = 1)
Sorted portion: [5]
Unsorted portion: [1, 8, 2]

We need to find the correct spot for: 1.
1 is the first element in the unsorted portion.
Now let's compare 1 with the elements of the sorted portion.
Let's find where it belongs...

-> Inner loop
Is the element selected 1 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 1
Moving 5 from index 0 to index 1 (see below)
Old list: [5, 1, 8, 2]
New list: [5, 5, 8, 2]
See how 5 is now at index 1

Bingo!
We've found the right location for 1: index 0
The list is now: [1, 5, 8, 2]
```



```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 1

```
>>> insertion_sort([5, 1, 8, 2])

=========> Starting Insertion Sort

---> Outer loop. Iteration #1 (i = 1)
Sorted portion: [5]
Unsorted portion: [1, 8, 2]

We need to find the correct spot for: 1.
1 is the first element in the unsorted portion.
Now let's compare 1 with the elements of the sorted portion.
Let's find where it belongs...

-> Inner loop
Is the element selected 1 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 1
Moving 5 from index 0 to index 1 (see below)
Old list: [5, 1, 8, 2]
New list: [5, 5, 8, 2]
See how 5 is now at index 1

Bingo!
We've found the right location for 1: index 0
The list is now: [1, 5, 8, 2]
```

[1, 5, 8, 2]

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 1

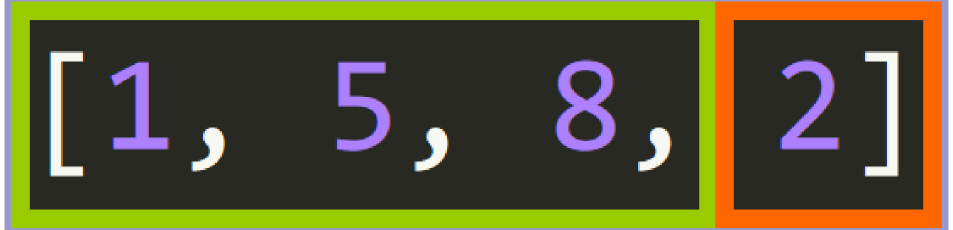
```
>>> insertion_sort([5, 1, 8, 2])

=========> Starting Insertion Sort

---> Outer loop. Iteration #1 (i = 1)
Sorted portion: [5]
Unsorted portion: [1, 8, 2]

We need to find the correct spot for: 1.
1 is the first element in the unsorted portion.
Now let's compare 1 with the elements of the sorted portion.
Let's find where it belongs...

-> Inner loop
Is the element selected 1 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 1
Moving 5 from index 0 to index 1 (see below)
Old list: [5, 1, 8, 2]
New list: [5, 5, 8, 2]
See how 5 is now at index 1

Bingo!
We've found the right location for 1: index 0
The list is now: [1, 5, 8, 2]
```

```
[1, 5, 8, 2]
```

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 8

```
---> Outer loop. Iteration #2 (i = 2)
Sorted portion: [1, 5]
Unsorted portion: [8, 2]

We need to find the correct spot for: 8.
8 is the first element in the unsorted portion.
Now let's compare 8 with the elements of the sorted portion.
Let's find where it belongs...

Is the element selected (8) smaller than 5?
No, it isn't! We need to stay where we are, at index 2.
The element 8 should be there.

Bingo!
We've found the right location for 8: index 2
The list is now: [1, 5, 8, 2]
```



```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

```
---> Outer loop. Iteration #2 (i = 2)
Sorted portion: [1, 5]
Unsorted portion: [8, 2]

We need to find the correct spot for: 8.
8 is the first element in the unsorted portion.
Now let's compare 8 with the elements of the sorted portion.
Let's find where it belongs...

Is the element selected (8) smaller than 5?
No, it isn't! We need to stay where we are, at index 2.
The element 8 should be there.

Bingo!
We've found the right location for 8: index 2
The list is now: [1, 5, 8, 2]
```

```
[1, 5, 8, 2]
```

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 2

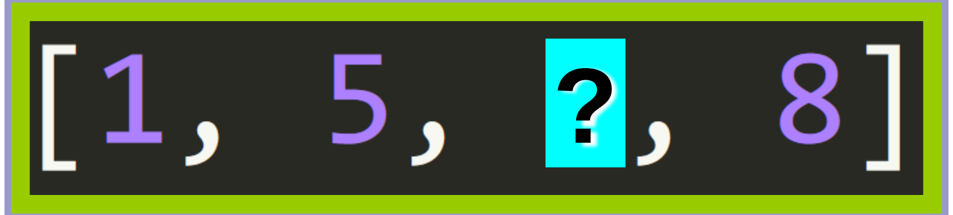
```
---> Outer loop. Iteration #3 (i = 3)
Sorted portion: [1, 5, 8]
Unsorted portion: [2]

We need to find the correct spot for: 2.
2 is the first element in the unsorted portion.
Now let's compare 2 with the elements of the sorted portion.
Let's find where it belongs...

-> Inner loop
Is the element selected 2 smaller than 8?
Yes, it is! So we need to move 8 to the right to make room for 2
Moving 8 from index 2 to index 3 (see below)
Old list: [1, 5, 8, 2]
New list: [1, 5, 8, 8]
See how 8 is now at index 3
```

```
[1, 5, ?, 8]
```

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 2

```
---> Outer loop. Iteration #3 (i = 3)
Sorted portion: [1, 5, 8]
Unsorted portion: [2]

We need to find the correct spot for: 2.
2 is the first element in the unsorted portion.
Now let's compare 2 with the elements of the sorted portion.
Let's find where it belongs...

-> Inner loop
Is the element selected 2 smaller than 8?
Yes, it is! So we need to move 8 to the right to make room for 2
Moving 8 from index 2 to index 3 (see below)
Old list: [1, 5, 8, 2]
New list: [1, 5, 8, 8]
See how 8 is now at index 3
```



```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 2

```
-> Inner loop
Is the element selected 2 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 2
Moving 5 from index 1 to index 2 (see below)
Old list: [1, 5, 8, 8]
New list: [1, 5, 5, 8]
See how 5 is now at index 2

Is the element selected (2) smaller than 1?
No, it isn't! We need to stay where we are, at index 1.
The element 2 should be there.

Bingo!
We've found the right location for 2: index 1
The list is now: [1, 2, 5, 8]
The list is now sorted!
```

[1, ?, 5, 8]

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

elem_selected: 2

```
-> Inner loop
Is the element selected 2 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 2
Moving 5 from index 1 to index 2 (see below)
Old list: [1, 5, 8, 8]
New list: [1, 5, 5, 8]
See how 5 is now at index 2

Is the element selected (2) smaller than 1?
No, it isn't! We need to stay where we are, at index 1.
The element 2 should be there.

Bingo!
We've found the right location for 2: index 1
The list is now: [1, 2, 5, 8]
The list is now sorted!
```

```
[1, 2, 5, 8]
```

```python
def insertion_sort(lst):
    for i in range(1, len(lst)):
        elem_selected = lst[i]

        while i > 0 and elem_selected < lst[i-1]:
            lst[i] = lst[i-1]
            i -= 1

        lst[i] = elem_selected
```

Sorted!
☺

```
-> Inner loop
Is the element selected 2 smaller than 5?
Yes, it is! So we need to move 5 to the right to make room for 2
Moving 5 from index 1 to index 2 (see below)
Old list: [1, 5, 8, 8]
New list: [1, 5, 5, 8]
See how 5 is now at index 2

Is the element selected (2) smaller than 1?
No, it isn't! We need to stay where we are, at index 1.
The element 2 should be there.

Bingo!
We've found the right location for 2: index 1
The list is now: [1, 2, 5, 8]
The list is now sorted!
```

Time to Practice!