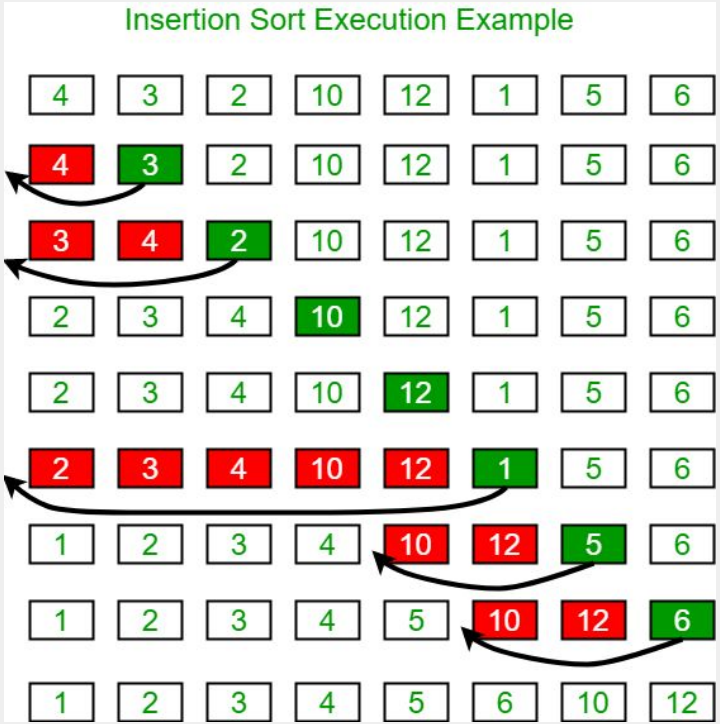
Insertion sort

How insertion sort works?

Insertion sort is a simple sorting algorithm that works the way we sort playing cards in our hands.



Algorithm explained

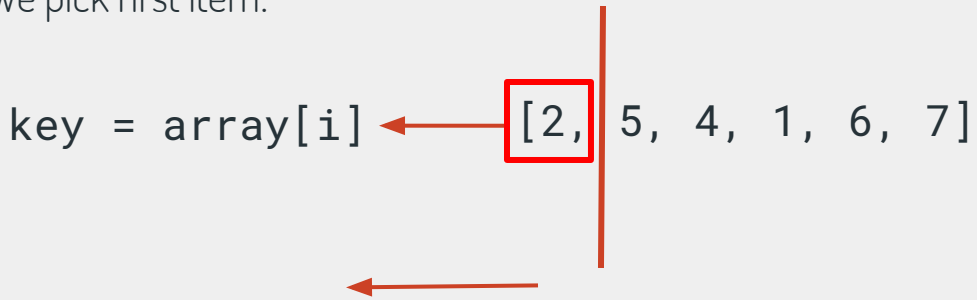
Given array of integers:

```
[2, 5, 4, 1, 6, 7]
```

We have to go through the array:

```
for (i = 0, i < array.length; i++)
```

We pick first item:

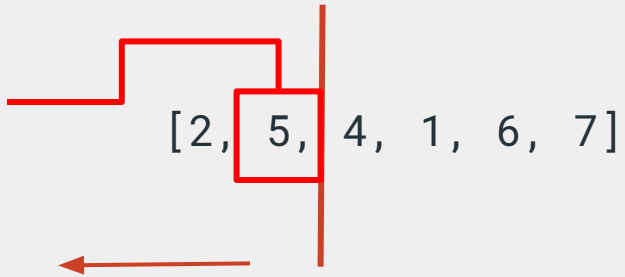


Everything on the left is already sorted

Algorithm explained

We pick second item:

```
key = array[i]
```




```
i = 1
```

Now we have to go through all elements on the left to **insert** our number

So let's define an index for a new loop:

```
j = i - 1
```

```
j = 0
```

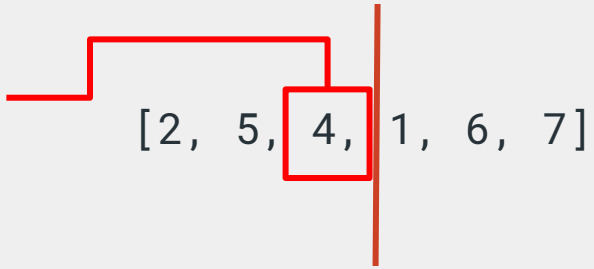
We can define our loop with check for swap:

```
while (j >= 0 && array[j] > key)
```

Check is here because of the fact that there is no need to go through sublist on the left because the first element of the list is lower than key.

We pick third item:

```
key = array[i]
```




```
for (i = 0, i < array.length; i++)
    key = array[i]
    j = i - 1
    while (j >= 0 && array[j] > key)
```

i = 2

key = 4

j = 1

Now **4 < 5** so we can now enter our loop and implement loop body:

If **4 < 5** we can swap them:

```
array[j + 1] = array[j]
```

and decrement index of inner array:

```
j = j - 1
```

j = 0

Is it done?

```
for (i = 0, i < array.length; i++)
        key = array[i]
        j = i - 1
        while (j >= 0 && array[j] > key)
            array[j + 1] = array[j]
            j = j - 1
```

Current state:

```
[2, 5, 5, 1, 6, 7]
```

We have to **insert** key:

```
array[j + 1] = key
```

```
i = 2
key = 4
j = 1

j = 0

array[1] = 4
```

Exercise

1. Program takes an argument from a command line - that argument will be a name of a file containing numbers to be sorted.
2. Program reads that file and insert all numbers to array.
3. Program sort numbers with our own insertion sort algorithm.
4. Program creates an output file (prefix: "sorted_") with sorted numbers.
5. Program displays information about execution time of every operation: read data from file and insert them to array; sort numbers; write output file.
6. Try to sort already sorted file - how long does it take?
7. Modify your program - check execution times for LinkedList and ArrayList .
8. Modify your program - check execution times for your insertion sort and built in sort method: `Array.sort(int[])`

FILES

There are 7 files - containing 1 000, 10 000, 50 000, 100 000, 500 000, 1 000 000 and 3 000 000 numbers. Please start with 1000 numbers file first!