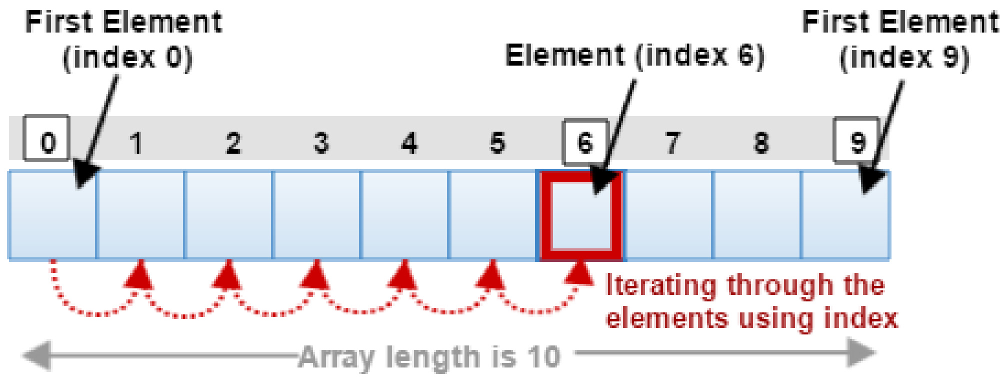
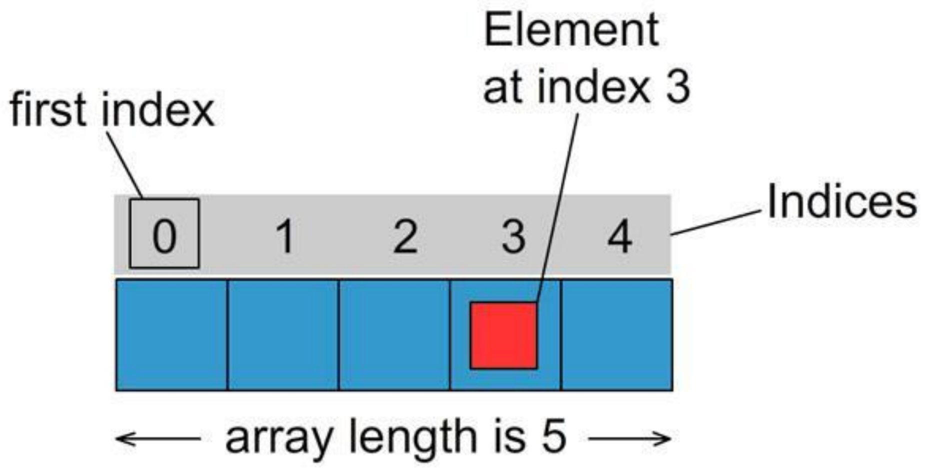
Arrays in Java

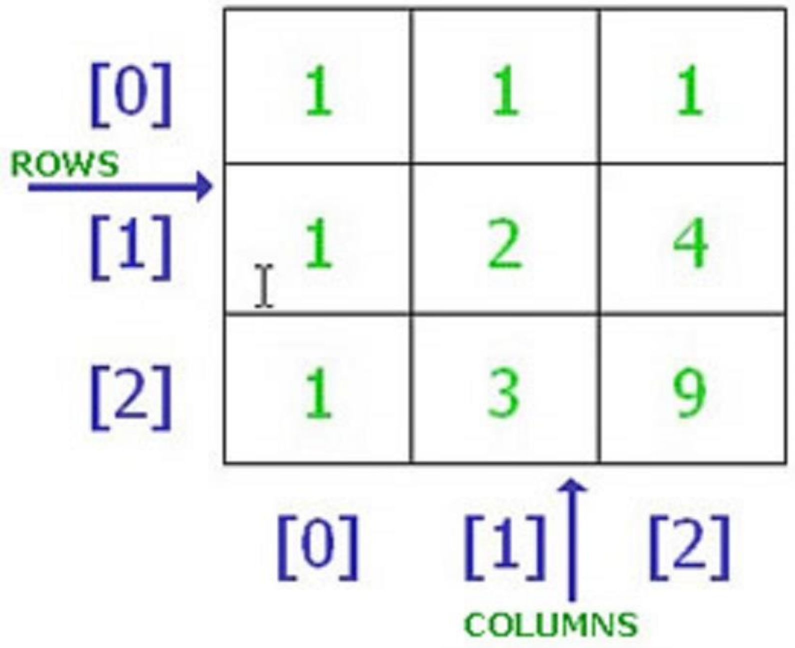
- An array is a fixed size sequential collection of elements of identical types.
- They are created with either a new operator or array initializes.
- The element in an array are indexed by the integers 0 to n- 1, where n is the size of the array



Types of arrays in Java

- SIngle Dimensional array
- Multidimensional array





Types of arrays in Java

SIngle Dimensional array

Syntax

- to Declare an Array

```
dataType[] arr; (or)  dataType []arr; (or) dataType arr[];
```

- to Instantiation of an Array

```
array=new datatype[size];
```

Example 1

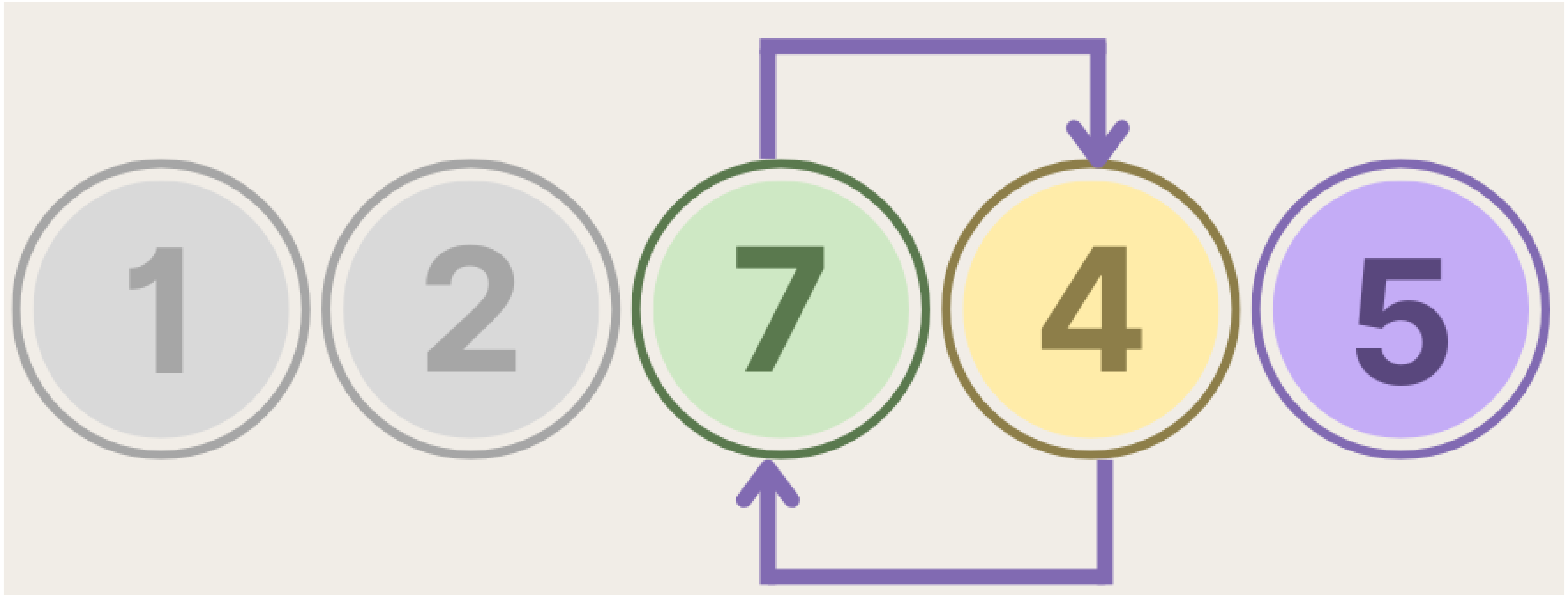
```
class Testarray{
public static void main(String args[])
{
        int [] a=new int[5]; //declaration and
instantiation
        a[0]=10; //initialization
        a[1]=20;
        a[2]=70;
        a[3]=40;
        a[4]=50;
```

```
 //printing array
```

Declaration, Instantiation and Initialization of Java Array

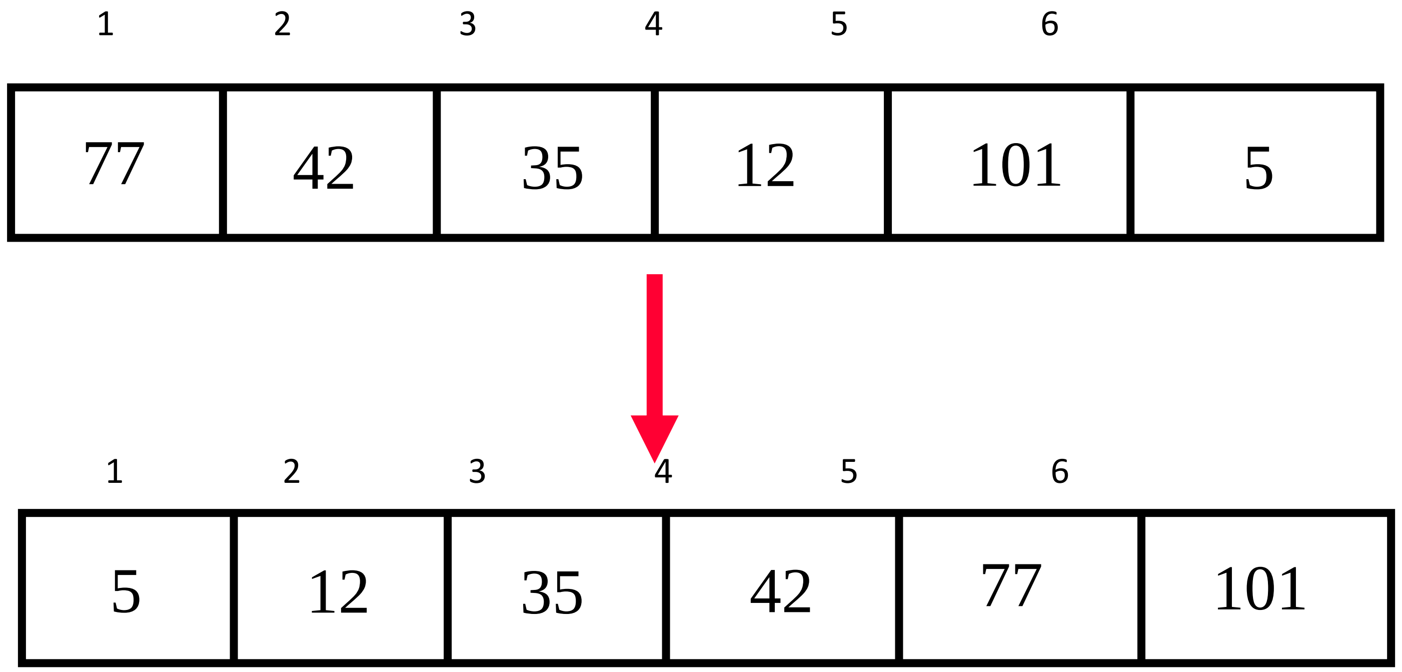
```
int a[]={33,3,4,5};//declaration, instantiation and initialization

class Testarray1{
public static void main(String args[]){

 int a[]={33,3,4,5};//declaration, instantiation and initialization

//printing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);

}}
```

Bubble Sort



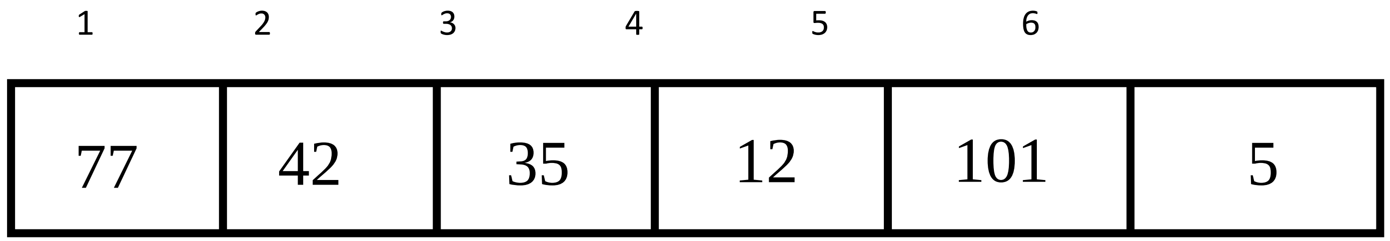
Sorting

Sorting takes an unordered collection and makes it an ordered one.



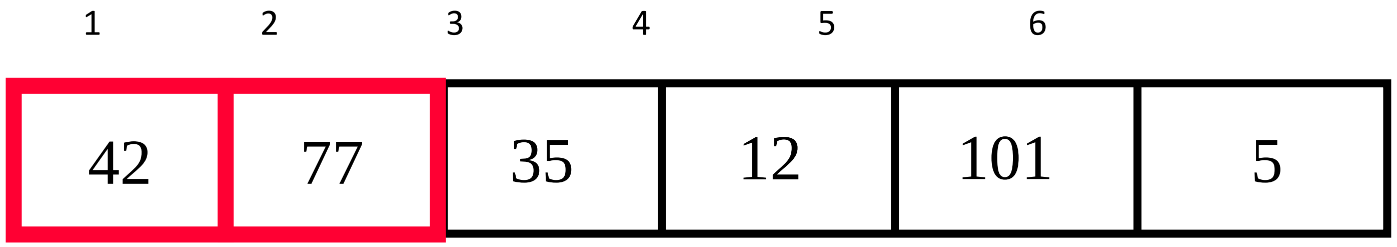
Traverse a collection of elements

- Move from the front to the end
- "Bubble" the largest value to the end using pair-wise comparisons and swapping



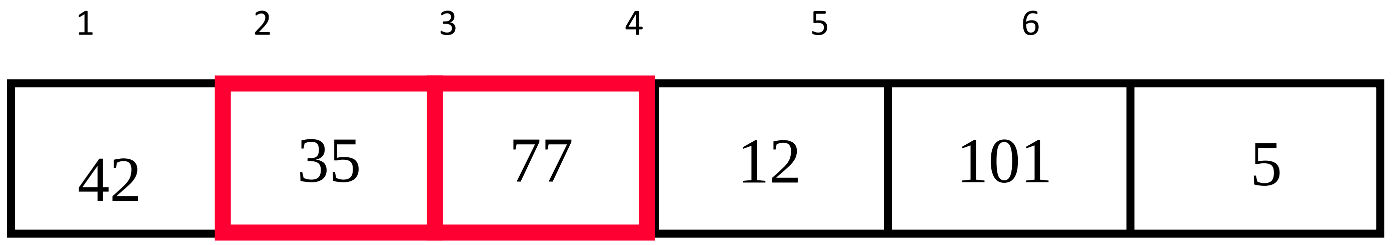
Traverse a collection of elements

- Move from the front to the end
- “Bubble” the largest value to the end using pair-wise comparisons and swapping



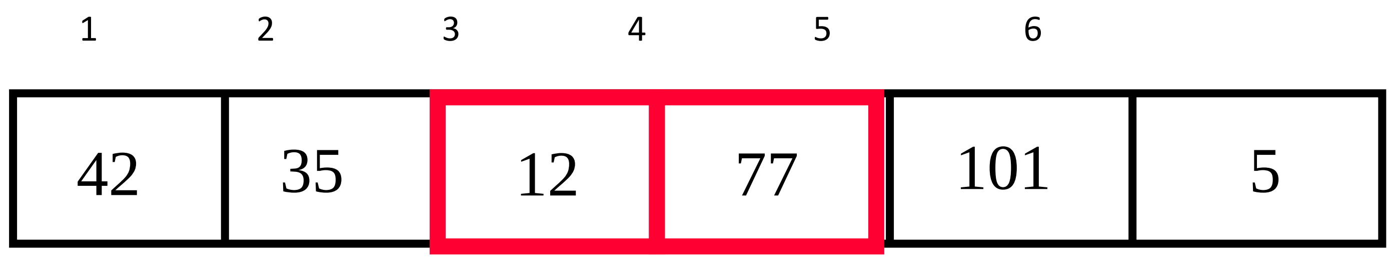
Traverse a collection of elements

- Move from the front to the end
- "Bubble" the largest value to the end using pair-wise comparisons and swapping



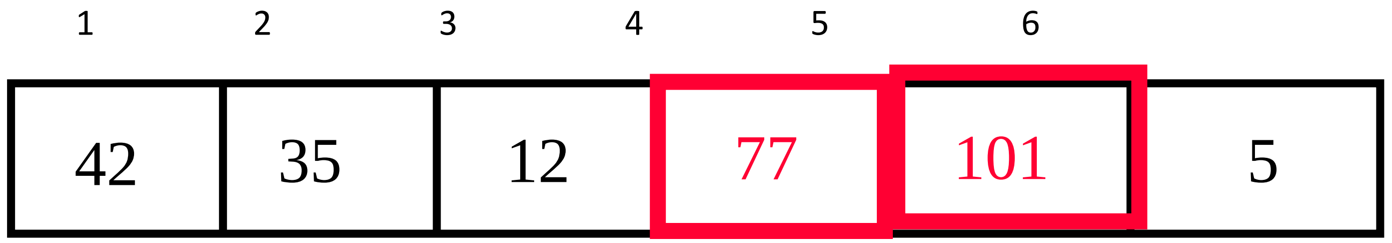
Traverse a collection of elements

- Move from the front to the end
- “Bubble” the largest value to the end using pair-wise comparisons and swapping



Traverse a collection of elements

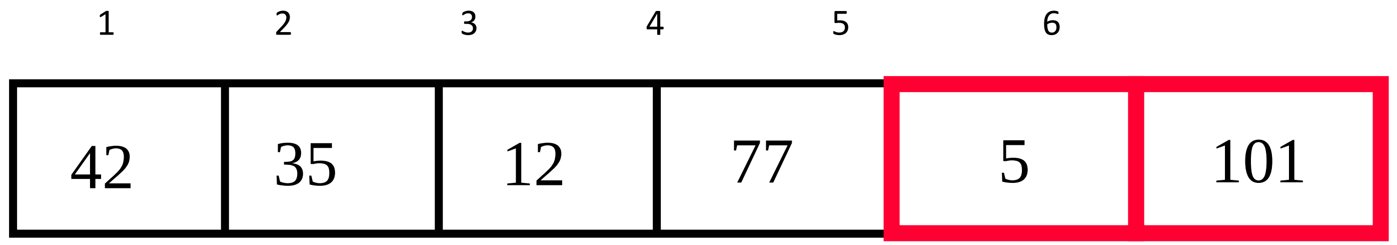
- Move from the front to the end
- "Bubble" the largest value to the end using pair-wise comparisons and swapping



No need to swap

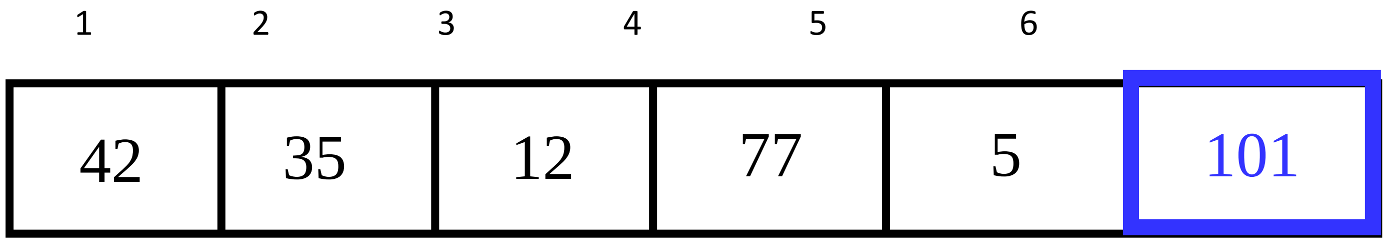
Traverse a collection of elements

- Move from the front to the end
- “Bubble” the largest value to the end using pair-wise comparisons and swapping



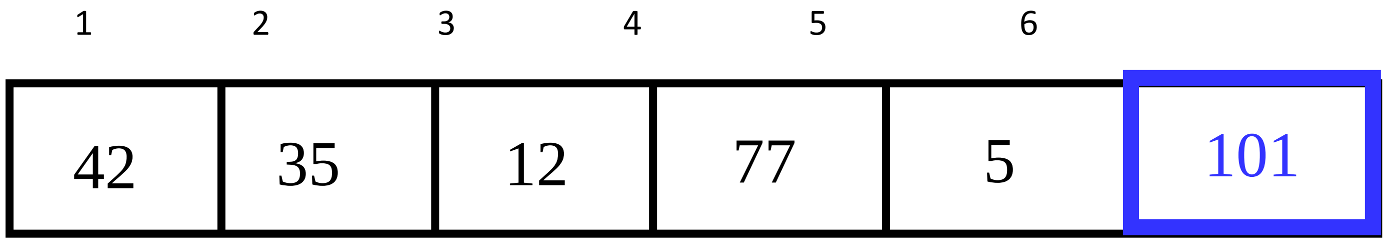
Traverse a collection of elements

- Move from the front to the end
- "Bubble" the largest value to the end using pair-wise comparisons and swapping



Largest value correctly placed

- Notice that only the largest value is correctly placed
- All other values are still out of order
- So we need to repeat this process



Largest value correctly placed

The "Bubble Up" Algorithm

```
index <- 1
last_compare_at <- n-1
loop
  exitif(index > last_compare_at)
  if(A[index] > A[index + 1]) then
    Swap(A[index], A[index + 1])
  endif
  index <- index + 1
endloop
```

LB

```
Procedure Swap(a, b)
 t <- a
  a <- b
  b <- t
endprocedure
```

Already Sorted Collections?

- What if the collection was already sorted?
- What if only a few elements were out of place and after a couple of “bubble ups,” the collection was sorted?
- We want to be able to detect this and “stop early”!

1	2	3	4	5	6
5	12	35	42	77	101

Using a Boolean “Flag”

- We can use a boolean variable to determine if any swapping occurred during the “bubble up.”

- If no swapping occurred, then we know that the collection is already sorted!

- This boolean “flag” needs to be reset after each “bubble up.”

Types of arrays in Java

Multidimensional array

Syntax

- to Declare an Array

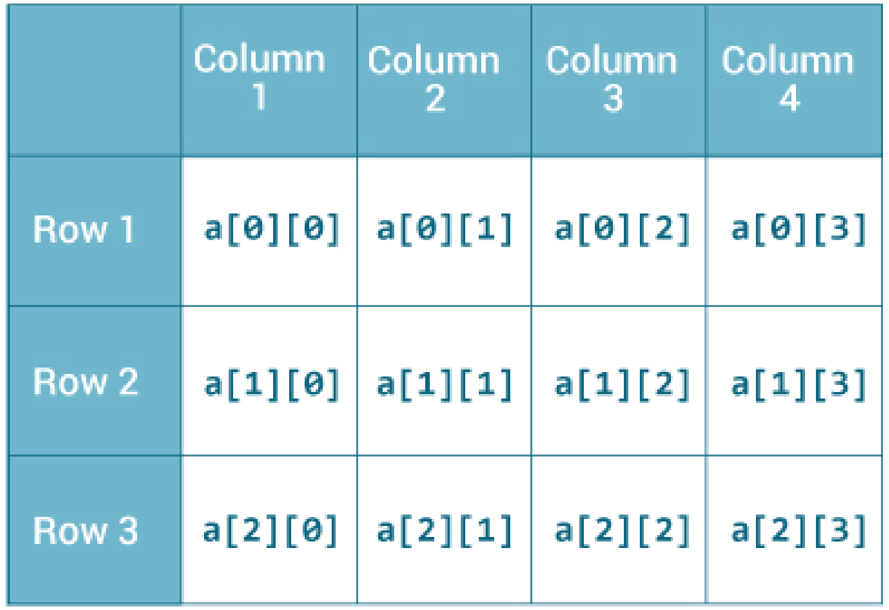
  dataType[][] array; (or) dataType array[][];

  (or) dataType []array[];

- to Instantiation of an Array

  int[][] arr=new int[3][3];//3 row and 3

column



	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Example 1

```
class Testarray{
        public static void main(String args[]){
            //declaring and initializing 2D array
        int arr[][]={{1,2,3},{2,4,5},{4,4,5}};
            //printing 2D array
        for(int i=0;i<3;i++)
        {
         for(int j=0;j<3;j++)
         {
          System.out.print(arr[i][j]+" ");
         }
```