Lecture 29

- Covers
  - Searching arrays
  - Sorting arrays

- Reading: Savitch 6.3, 6.4

- Searching arrays

Example

- We want to write a class that contains an array of integers and allows us to manipulate it safely
- It will need
  - A constructor
  - A method to read in values
  - A method to change a value
  - A method to retrieve a value
  - Methods to find the index of certain values
  - A method to sort the array

Define the class

```
public class IntArray
{

}
```

Declare the attribute

```
public class IntArray
{
    int[ ] values;

}
```

Declare the constructor(s)

```
public IntArray(int size)
{
        values = new int[size];
}
```

Q: What happens if size is not a positive integer?

Declare the constructor(s)

```
public IntArray(int size)
{
    if (size > 0)
    {
        values = new int[size];
    }
    else
    {
        values = new int[0];
    }
}
```

Define the methods

- A method to read in values

```
public void readValues( )
{
    System.out.println("Enter " + values.length + " integers");
    for (int i = 0; i < values.length; ++i)
    {
        values[i] = keyboard.nextInt( );
    }
}
```

Define the methods

- A method to change a value

```
public boolean setValue(int index, int value)
{
    boolean successful = false;

    if (index >= 0 && index < values.length)
    {
        values[index] = value;
        successful = true;
    }
    return successful;
}
```

Define the methods

- A method to retrieve a value

```
public int getValue(int index)
{
    return values[index];
}
```

Searching methods

- To find a value in an unsorted array, we must start at the beginning and iterate through the array looking for the search key
- To search for the value 8, start by looking at position 0, then 1, then 2 (where 8 is found)

a→	15	12	8	10	3	4	9	11

Searching methods

- To search for the value 2, start by looking at position 0, then 1, then 2, then 3, then 4, then 5, then 6, then 7
- As there are no more elements in the array, the search was unsuccessful

a→	15	12	8	10	3	4	9	11

Find first occurrence of a value

```
public int findFirstOccurrence(int searchValue)
{
    int index;
    for (index = 0; index < values.length; ++index)
    {
        if (values[index] == searchValue)
            break;
    }
    return index;
}
```

Alternative findFirstOccurrence

```
public int findFirstOccurrence(int searchValue)
{
    int index;
    for (index = 0; index < values.length &&
                    values[index] != searchValue; ++index);
    return index;
}
```

Alternative findFirstOccurrence

```
public int findFirstOccurrence(int searchValue)
{
    bool found = false;
    int indexOfFirstOccurrence = -1;
    int index = 0;
    while (!found && index < values.length)
    {
        if (values[index] == searchValue)
        {
            found = true;
            indexOfFirstOccurrence = index;
        }
    }
    return indexOfFirstOccurrence;
}
```

Class exercise

- Write a method to find the last occurrence of a value in the array

- Sorting arrays

Sorting the array

- Sometimes we wish to sort an array so that the values in the array are organised in a specific order

Sorting

- Unsorted array

a→	15	12	8	10	3	4	9	11

- Sorted array

a→	3	4	8	9	10	11	12	15

How do we sort it into numerical order?

- Many sort algorithms have been developed, for example
  - Selection sort
  - Insertion sort
  - Bubble sort
  - Quick sort
  - Merge sort

Selection sort

- The array values are in any order and we want to rearrange them into order from minimum to maximum (ascending order)
- So we repeatedly select the minimum element from the remaining unsorted (back) section of the array and make it the last element in the sorted (front) section of the array

Selection sort

	0	1	2	3	4	5	6	7
initial unsorted array :	15	12	8	10	3	4	9	11
select 3 {a[4] swap with a[0]}	**3**	12	8	10	15	4	9	11
select 4 {a[5] swap with a[1]}	**3**	**4**	8	10	15	12	9	11
select 8 {a[2] swap with a[2]}	**3**	**4**	**8**	10	15	12	9	11
select 9 {a[6] swap with a[3]}	**3**	**4**	**8**	**9**	15	12	10	11
select 10 {a[6] swap with a[4]}	**3**	**4**	**8**	**9**	**10**	12	15	11

Selection sort

	0	1	2	3	4	5	6	7
select 11/a[7] swap with a[5]	3	4	8	9	10	11	15	12
select 12/a[7] swap with a[6]	3	4	8	9	10	11	12	15
a[7] is already sorted	3	4	8	9	10	11	12	15

Class exercise

- Problem
  - Show the steps in using selection sort to sort the following array

0	1	2	3	4
7	10	3	2	8

- Solution

Selection sort - the logic

- Pseudocode

LOOP FOR position from 0 to size of array – 1 (i.e. final position)

Find index of minimum element in array from position to size - 1

Swap minimum element and current element

ENDLOOP

Selection sort

- Refinement
- Solution 1
- Problems?

```
METHOD Find index of minimum element(
                    in   : array, initial index, final index
                    out : index of min)
  min = large number
  LOOP FOR position from initial to final index
      IF a[position] < min THEN
          index of min = position
          min = a[position]
      ENDIF
  ENDLOOP
  Return index of min
ENDMETHOD
```

Selection sort

- Refinement
- Solution 2

```
METHOD Find index of minimum element(
                    in   : array, initial index, final index
                    out : index of min)
  min = a[initial index]
  LOOP FOR position from initial index+1 to final
      IF a[position] < min THEN
          index of min = position
          min = a[position]
      ENDIF
  ENDLOOP
  Return index of min
ENDMETHOD
```

findIndexOfMinimum

```
private int findIndexOfMinimum( int start, int end )
{
    int indexOfMinimum = start;
    for ( int i = start + 1; i <= end; i++ )
    {
        if ( a[ i ] < a[ indexOfMinimum ])
        {
            indexOfMinimum = i;
        }
    }
    return indexOfMinimum;
}
```

** We are presuming that the base type of array a is int*

Class exercise

- Problem
  - For the array below step through

    findIndexOfMinimum(1, 5)

	0	1	2	3	4	5
values :	2	10	4	7	3	6

- Solution

Swapping over array elements

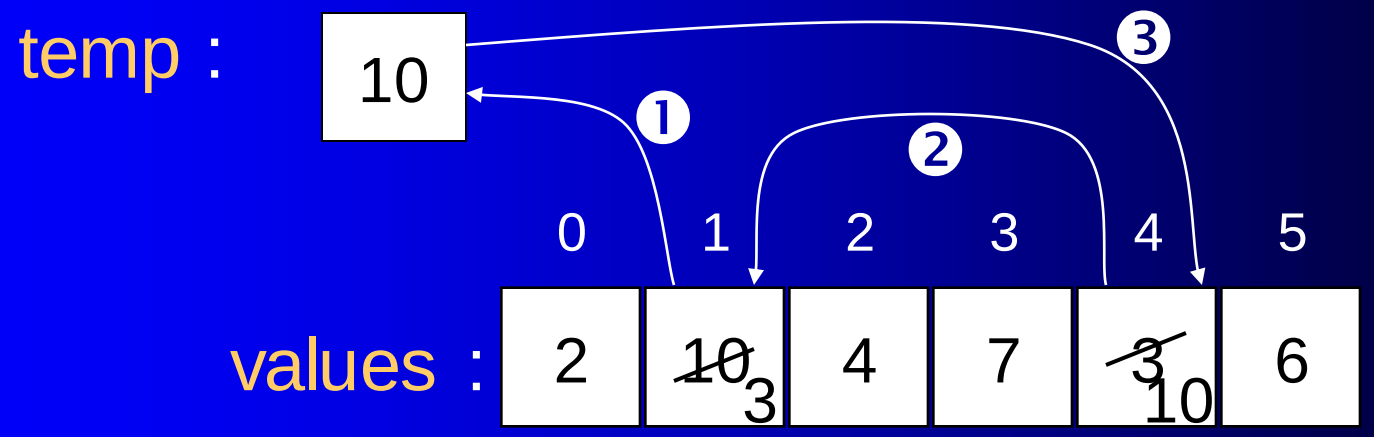
- Having found the minimum array element between start and end, we need to swap them over. How?
- E.g. swap the values at index 1 and index 4
- Requires a temporary store

temp :

	0	1	2	3	4	5
values :	2	10	4	7	3	6

Swapping over array elements

- Copy the first value into the temporary store
- Copy the second value into the position of the first value
- Copy the temporary value into the position of the second value



Solution

- Swap over the values in the array at index1 and index2

```
private void swapValues(int index1, int index2)
{
    int temp;
    temp = values[index1];
    values[index1] = values[index2];
    values[index2] = temp;
}
```

Selection sort

LOOP FOR position from 0 to size of array – 1 (i.e. final position)
Find index of minimum element in array from position to size - 1
Swap minimum element and current element
ENDLOOP

- Java code

```
public void sortArray( )
{
    int currentMin;
    for (int i = 0; i < values.length; ++i)
    {
        currentMin = findIndexOfMinimum(i, values.length - 1);
        swapValues (i, currentMin);
    }
}
```

Selection sort

- Back to the high-level pseudocode

LOOP FOR position from 0 to size of array – 1 (i.e. final position)
Find index of minimum element in array from position to size - 1
Swap minimum element and current element
ENDLOOP

- Any problems with this alternative?

LOOP FOR position from 0 to size of array – 2
Find index of minimum element in array from position to size - 1
Swap minimum element and current element
ENDLOOP

Class exercise

- Using the findIndexOfMinimum and swapValues methods, write a sort method that orders the elements in descending order
- For example, given the values

  12 3 4 54 34 23 -43 34 23 12

  after sorting they would have the order

  54 34 34 23 23 12 12 4 3 -43

Solution

Examples

- Sort an array of integers
- Sort the employee list by id
- Sort the employee list by name

Next lecture

- Multidimensional arrays