# AP Computer Science A

1. Log in
2. Unit 5
   - Array Review and Swapping Elements
   - Selection Sort
   - Insertion Sort
   - Bubble Sort
   - Merge Sort
   - Sequential Search
   - Binary Search

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

Original Array: 8 2 5 7 3

Check the first two elements ... they need to be swapped!

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

2 8 5 7 3

Swap is made ...

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

2 5 8 7 3

Move on, if a swap is needed, make it!

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

2 5 7 8 3

Move on, if a swap is needed, make it!

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

2 5 7 3 8

1st Pass COMPLETE!

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

1. Compare the first 2 elements (index 0 and 1), swap if needed.
2. Compare the second 2 elements (index 1 and 2), swap if needed.
3. Continue through the list.
4. Repeatedly go through list in this way until no elements are swapped.

NOTE: This process causes the largest # in the array to move to its "ending position" with each pass through the array.

Example of Bubble Sort

2 5 7 3 8

THE FIRST PASS IS COMPLETE!!!

CONTINUE PASSES TIL DONE!!!

Jan 22-8:05 AM

**Bubble Sort (sorting a list of n elements)...**

Example of Bubble Sort

After 1st Pass 2 5 7 3 8

After 2nd Pass 2 5 3 7 8

After 3rd Pass 2 3 5 7 8

DONE!!!

* Reminder: After each pass the largest item shifts to the "end"

Jan 22-8:05 AM

Bubble Sort ...

Consider the following array ...

Not in order!!!

5 3 7 2

Oct 7-12:59 PM

Bubble Sort ...

Consider the following array ...

Not in order!!!

5 3 7 2

Check first two numbers. Need change?

Oct 7-12:59 PM

Bubble Sort ...

Consider the following array ...

Not in order!!!

5 3 7 2

Check first two numbers. Need change?

YES! Swap 'em!

3 5 7 2

Oct 7-12:59 PM

Bubble Sort ...

Consider the following array ...

Not in order!!!

3 5 7 2

Check next two numbers. Need change?

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 7 2

Check next two numbers. Need change?

NO! Move On!

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 7 2

Check next two numbers. Need change?

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 7 2

Check next two numbers. Need change?

YES! Switch 'em!

3 5 2 7

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 7 2

Check next two numbers. Need change?

YES! Switch 'em!

3 5 2 7

Notice how the "largest" number has now "risen to the top" ... gone furthest to the right?

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

We now must make another "loop through" (continue until no swaps made)

3 5 2 7

Check first two numbers. Need change?

NO! Move On!

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 2 7

Check next two numbers. Need change?

YES! Swap 'em!

3 2 5 7

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 5 2 7

Check next two numbers. Need change?

YES! Swap 'em!

3 2 5 7

Check next two numbers. Need change?

NO! Move On!

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

Swap was made on last run through so ... do again!

3 2 5 7

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

Swap was made on last run through so ... do again!

3 2 5 7

Check next two numbers. Need change?

YES! Swap 'em!

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 2 5 7

Check next two numbers. Need change?

YES! Swap 'em!

2 3 5 7

No! No!

Oct 7-12:59 PM

## Bubble Sort ...

Not in order!!!

Consider the following array ...

3 2 5 7

Check next two numbers. Need change?

YES! Swap 'em!

2 3 5 7

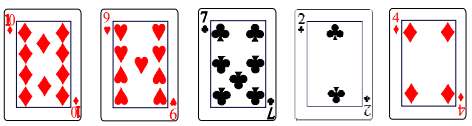
No! No!

* There was a swap on this pass so we must pass through again to find "no swaps"!

Oct 7-12:59 PM

## Bubble Sort Example with Cards ...

* Count the "testing"
* How many passes?



Jan 22-8:38 AM

## Bubble Sort ...

Here's what it looks like in JAVA ... step by step ...

```
public class bubbleSort {
   public static void sort(int a[])
   {
      boolean keepGoing=true;
      while(keepGoing)
      {
         keepGoing=false;
         for(int i=0;i<a.length-1;i++)
         {
            if(a[i]>a[i+1])
            {
               int temp=a[i];
               a[i]=a[i+1];
               a[i+1]=temp;
               keepGoing=true;
            }
         }
      }
   }
   public static void main(String[] args) {
      int[] theArray = {2,5,7,1,33,11,3};
      sort(theArray);
      for(int i=0;i<theArray.length;i++)
         System.out.print(theArray[i]+"  ");
   }
}
```

Oct 7-12:59 PM

**Things to do ...**

1. Wrap Up Unit 6 WS 01-03
2. Work on Unit 6 WS04 Arrays and Bubble Sort

Sep 26-10:58 AM