Searching and Sorting in One dimensional array

- **Search** : linear search vs. binary search (requires the array elements to be sorted)
  - return the subscript of the array element that match the value that is being searched for
  - return -1 if the value is not there

linear search in array

```
int LinearSearch (const int a[], int aSize, int toFind)
{
    // Look through all items, starting at the front.
    for (int i = 0; i < aSize; i++)
        if (a[i] == toFind)
            return i;

    // You've gone through the whole list without success.
    return -1;
}
```

Binary search in array

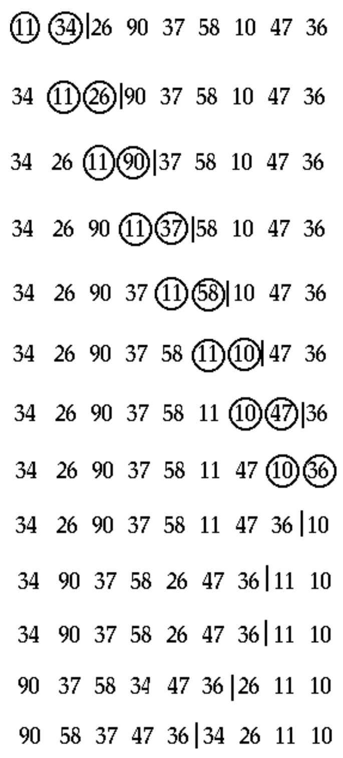
```
 int BinarySearch(int a[], int aSize, int toFind)
{
   int start = 0;                          //the search starts with index 0
   int last = aSize -1;                    //last is the last array index

   while (start <= last)                   //while there is still a place to look.
   {
       int middle = (start + last) / 2;    //Look here first
       if (toFind == a[middle])            //Found item. Quit.
            return middle;
       if (toFind > a[middle])        //Look in the last half
            start = middle + 1;
       else                                //OR look in the first half
            last =  middle - 1;
   }

   //the element wasn't found
   return -1;
}
```

- **Sorting**

bubble sort (The xSort Applet : http://math.hws.edu/TMCM/java/xSortLab/)

```
        void BubbleSort (int list[], int listSize)
         {
            bool sorted= false;       //is the list sorted?

            //start last at the last array element
            int last = listSize - 1;
            int i;                    //used as a loop index

            while ( !sorted )
            {
                 //assume the list is in order
                 sorted = true;
                 for (i = 0; i < last; i++)
                {
                      if (list[i] < list[i+1])
                      {
                         //swap two elements
                         Swap (list[i], list[i+1]);

                         //the list wasn't already sorted
                         sorted = false;
                      }
                }
                last--;
            }

void Swap (int  &value1,  int  &value2)
{
            int    tmp;

            tmp = value1;
            value1 = value2;
            value2 = tmp;

            return;
}
```

Example


11 34 | 26 90 37 58 10 47 36
34 11 26 | 90 37 58 10 47 36
34 26 11 90 | 37 58 10 47 36
34 26 90 11 37 | 58 10 47 36
34 26 90 37 11 58 | 10 47 36
34 26 90 37 58 11 10 | 47 36
34 26 90 37 58 11 10 47 | 36
34 26 90 37 58 11 47 10 36
34 26 90 37 58 11 47 36 | 10
34 90 37 58 26 47 36 | 11 10
34 90 37 58 26 47 36 | 11 10
90 37 58 34 47 36 | 26 11 10
90 58 37 47 36 | 34 26 11 10