```
//===  SORTING and other ALGORITHMS ==============

// Bubble Sort //

       for(int p=0; p < numsSize; p = p+1)
        {
            for(int x=0; x < numsSize-1; x = x+1)
            {
                if(nums[x] > nums[x+1])
                {
                    int temp = nums[x];
                    nums[x] = nums[x+1];
                    nums[x+1] = temp;
                }
            }
        }

//================================================

// creates a list of numbers and randomly scrambles the order //

        for(int x=0; x < numsSize; x = x+1)
        {
            nums[x] = x;
        }

        for(int s = 0; s < numsSize/2; s = s+1)
        {
            int a = (int)(Math.random()*numsSize);
            int b = (int)(Math.random()*numsSize);
            int temp = nums[a];
            nums[a] = nums[b];
            nums[b] = temp;
        }
//===============================================

//  Selection Sort //

        for(int f = 0; f < numsSize-1; f = f+1 )
        {   int smallest = nums[f];
            int pos = f;
            for(int x = f+1; x < numsSize; x = x+1)
            {
                if(nums[x] < smallest)
                {
                    smallest = nums[x];
                    pos = x;
                }
            }
            int temp = nums[f];
            nums[f] = nums[pos];
            nums[pos] = temp;
        }

//===============================================

// Reverse a list //

         for(int x=0; x <= max/2 ; x = x+1)
        {
            int temp = nums[x];
            nums[x] = nums[max-x];
            nums[max-x] = temp;
        }
```

```
// Mr Mulkey's Mystery sort//

      for(int a=0; a < numsSize; a=a+1)
        {
            for(int b=0; b < numsSize; b=b+1)
            {
                if(nums[a]<nums[b])
                {
                    int temp = nums[a];
                    nums[a] = nums[b];
                    nums[b] = temp;
                }
            }
        }

//===============================================

// Insert a new number nn into a sorted list //

        int temp = 0;
        while(nn>=nums[temp] && temp<numsSize)
        {
            temp = temp+1;
        }
        for(int x=numsSize-1; x >= temp ; x=x-1)
        {
            nums[x+1] = nums[x];
        }
        nums[temp] = nn;
        numsSize = numsSize+1;

//==============================================

// calculate the average and print all numbers that are smaller than average//

        double total = 0;
        for(int x=0; x < numsSize; x = x+1)
        {
            total = total + nums[x];
        }

        int count = 0;
        for(int x=0; x < numsSize; x=x+1)
        {
            if(nums[x] < total/numsSize)
            {  count = count + 1; }
        }
        return count;

//==============================================

//  Search for a number in a list //

        boolean found = false;
        for(int x = 0; x < numsSize; x=x+1)
        {
            if(nums[x] == target)
            {  found = true;  }
        }
        return found;
```

```
//==============================================

//  Print all the values that are between 100 and 999

       for(int x=0; x < numsSize; x = x+1)
       {
          if( nums[x]>=100 && nums[x]<1000 )
          {
              IBIO.output(nums[x]);
          }
       }
```