Name _____ Period_____

1. Consider the following code segment from an insertion sort program.

```
for (int j = 1; j < arr.length; j++)
{
     int insertItem = arr[j];
     int k = j - 1;
          while (k >= 0 && insertItem < arr[k])
          {
               arr[k + 1] = arr[k];
               k--;
          }
     arr[k + 1] = insertItem;
     /* end of for loop */
}
```

Assume that array arr has been defined and initialized with the values `{5, 4, 3, 2, 1}`.

What are the values in array arr after each pass of the for loop (i.e., when j = 1, when j = 2, when j = 3, when j = arr.length-1)?

2. Consider the following sort method. This method correctly sorts the elements of array data into increasing order.

```
public static void sort(int[] data)
{
     for (int j = 0; j < data.length - 1; j++)
     {
          int m = j;
          for (int k = j + 1; k < data.length; k++)
          {
               if (data[k] < data[m])
               /* Compare values */
               {
                    m = k;
               }
          }
     int temp = data[m];
     /* Assign to temp */
     data[m] = data[j];
     data[j] = temp;
     /* End of outer loop */
     }
```

(a) Assume that sort is called with the array {6, 3, 2, 5, 4, 1}. What will the value of data be after each pass of the outer loop (i.e., when j = 1, when j = 2, when j = 3, when j = arr.length-1)?

(b) Assume that sort is called with the array {1, 2, 3, 4, 5, 6}. How many times will the expression indicated by /* Compare values */ and the statement indicated by /* Assign to temp */ execute?

3. Given the following array as input, illustrate how the Mergesort algorithm performs. To illustrate the Mergesort's behavior, start with the dividing of the array until the end condition of the recursive function is met and then show how the merge is performed.

3 8 4 10 1 5 6 9

4. Given the following array [10, 5, 3, 9, 22, 24, 28, 27, ?] and assuming that Quicksort will be used to sort this array in ascending order,

(a) Select a value for the last element of the array (indicated by "?") such that the partitioning performed by Quicksort is most balanced. Explain why this makes Quicksort perform efficiently.

(b) Show the results of the first two rounds of the Quicksort algorithms based on the number you have chosen.

5. The following code fragment does a sequential search to determine whether a given integer `value` is stored in an array `a[0] … a[n – 1]`. What should replace `/* boolean expression */` so that the algorithm works as intended?

```
int i = 0;
while( /* boolean expression */ )
{
     i++;
}
if(i == n)
     return -1;      //value not found
else
     return i;      //value found at location i
```

6. Refer to the code below to answer the following

```
private int[] a;

/** Does binary search for key in array a[0] … a[a.length-1].
*  sorted in ascending order.
*  @param key the integer value to be found
*  Postcondition:
*  - index has been returned such that a[index] == key
*  - If key not in a, return -1.
*/
public int binSearch(int key){
     int low = 0;
     int high = a.length - 1
     while(low <= high){
          int mid = (low - high) / 2;
          if(a[mid] == key)
               return mid;
          else if(a[mid] < key)
               low = mid + 1;
          else
               high = mid - 1;
     }
     return -1;
}
```

A binary search will be performed on the following list,

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
4	7	9	11	20	24	30	41

(a) How many iterations will be required to determine that 27 is not in the list?

(b) If the key value searched is 27, what is the search interval (a[?] … a[?]) for each pass through the while loop?

(c) What will be stored in y after executing the following?

```
int y = binSearch(4)
```

(d) If the test for the while loop is changed to

```
while(low < high)
```

the `binSearch` method does not work as intended. Which value(s) in the given list will not be found?

7. For each of the following sets, how many iterations will be required to find a key value using an iterative binary search algorithm, (Note: $10^3 \sim 2^{10}$)

(a) 1000 elements

(b) 2000 elements

(c) 30,000 elements

(d) 600 elements

(e) 1 million elements

8. An array of integer values is to be searched for a prime number. Once a prime number is found the algorithm will return the value of the prime number. If no prime number is found -1 will be returned.

Consider the examples below,

Array 1:	Array 2:	Array 2:
4 | 6 | 8 | 7	4 | 6 | 3 | 7	4 | 6 | 9 | 2
Returns 7	Returns 3	Returns -1

Write the method findPrimes which accepts a one-dimensional array of integer values and returns the first prime number found or returns -1 if no prime numbers are found.