CS 1713 Exam 2 Solutions — Fall 2008

1a) (5 points)

```
public int linearSearch(String[] array, Comparable target) {
   for (int i=0;i<array.length;i++)
      if (array[i].equals(target))
         return i;
   return -1;
}
```

1b) (15 points)

```
public int binarySearch(String[] array, Comparable target) {
   int low = 0;
   int high = array.length - 1;
   while (low <= high) {
      int mid = (low + high)/2;
      if (array[mid].equals(target))
         return mid;
      else if (array[mid].compareTo(target) < 0)
         low = mid + 1;
      else
         high = mid - 1;
   }
   return -1;
   }
}
```

2) (20 points)

```
   public static void insertionSort (int[] list) {
      for (int index = 1; index < list.length; index++) {
         insertItem(list, index);
      }
   }

   public static void insertItem(int[] list, int index) {
       int key = list[index];
       int position = index;
       while (position > 0 && (key < list[position-1]))   {
          list[position] = list[position-1];
          position--;
       }
       list[position] = key;
    }
```

3) (5 points) a) 35 seconds b) 250 seconds c) about 3.3 seconds (1,000,000 elements requires about 20 times through the loop so 4 million requires about 22 times. 3*22/20 = 5.3.)

4) (10 points)

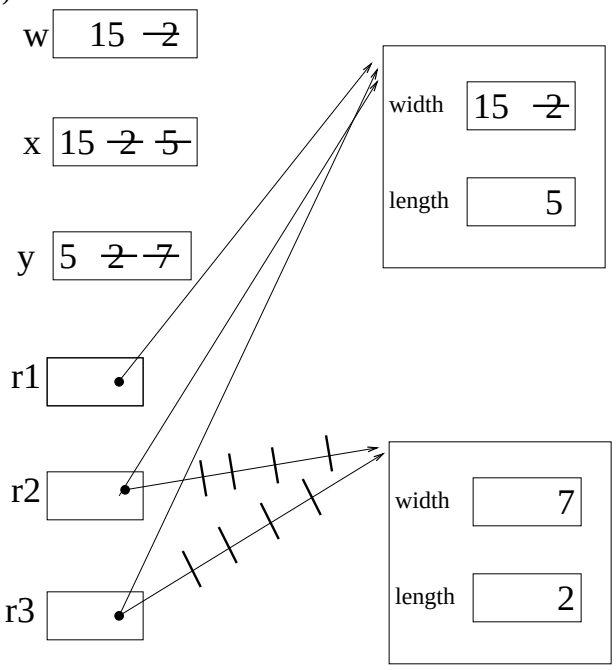
```
public boolean checkTwice(String s1, String s2) {
  int ind = s1.indexof(s2);
  if (ind == -1)
    return false;
  s1 = s1.substring(ind +1);
  if(s1.indexof(s2) == -1)
     return false;
  s1 = s1.substring(ind);
  return (s1.indexof(s2) == -1);
```

5) (12 points) a) 'x' b) 5 c) -1 d) 19 e) "ne is CS 2123" f) "xt o"

6) (8 points)

```
public boolean isSorted(double[] list) {
   for (int i=0;i<list.length - 1;i++)
      if (list[i] > list[i+1])
         return false;
   return true;
}
```

7) (8 points)



8) (20 points)

Selection Sort

0	1	2	3	4	5	6	7
24	**14**	**37**	**13**	**48**	**23**	**66**	**17**
13	14	37	24	48	23	66	17
13	14	37	24	48	23	66	17
13	14	17	24	48	23	66	37
13	14	17	23	48	24	66	37
13	14	17	23	24	48	66	37
13	14	17	23	24	37	66	48
13	14	17	23	24	37	48	66

Insertion Sort

0	1	2	3	4	5	6	7
24	**14**	**37**	**13**	**48**	**23**	**66**	**17**
14	24	37	13	48	23	66	17
14	24	37	13	48	23	66	17
13	14	24	37	48	23	66	17
13	14	24	37	48	23	66	17
13	14	23	24	37	48	66	17
13	14	23	24	37	48	66	17
13	14	17	23	24	37	48	66