CS101 – Computer Programming
Quiz for Monday Batch – 29 September 2014

Q1. Consider the following variation of mergeSort to sort the array in descending order. In the version of mergeSort discussed in class, the array is divided into two sub arrays of (almost) equal size. In the following version, the array is divided into two sub arrays of different sizes. The arrays are divided in the ratio 1/n and (n-1)/n.

The function **mergeSortedSubArrays(A, start, mid, end)** takes in input array A, where A[start], A[start+1], ..., A[mid-1] and A[mid], A[mid+1], ..., A[end-1] are sorted in descending order and when returning sorts the elements from A[start], ..., A[end-1] in descending order.

The function **selectionSort(A, start, end)** sorts the array A from start to end-1 in descending order.

```
void mergeSort(int A[], int start, int end, int n) {
      if(end == start)
            return;
      if(end – start < n) {
            selectionSort(A, start, end);
            return;
      }
      int mid = start + (end – start)/n;
      mergeSort(A, start, mid, n);
      mergeSort(A, mid, end, n);

      mergeSortedSubArrays(A, start, mid, end);
      return;
}

int main() {
      int A[11] = {1, -1, 3, 4, 6, 2, 7, 9, 0, -5, 11};
      mergeSort(A, 0, 11, 3);
      return 0;
}
```

The number of function calls of mergeSort when the above program is executed is:

A) 10
B) 11
C) 13
D) 12

Refer Q2 on the next page

Q2. Consider the following recursive implementation of **binarySearch** algorithm and choose the correct options among the following:

```
int binarySearch(int A[], int start, int end, int ele) {
     int mid = (start + end)/2;
     if(start == end-1) {
          if(ele == A[mid]) {
               cout << "Y";
               return mid;
          } else {
               cout << "N";
               return -1;
          }
     }
     if(ele == A[mid]) {
          cout << “Y”;
          return mid;
     } else if (ele < A[mid]) {
          cout << “L”;
          return binarySearch(A, start, mid, ele);
     } else {
          cout << “R”;
          return binarySearch(A, mid, end, ele);
     }
}

int main() {
 int A[15] = {4, 9, 16, 18, 22, 26, 33, 36, 37, 52, 60, 68, 88, 94, 99};
 int index = binarySearch(A, 0, 15, ELE);
}
```

Consider the folowing statements about the above program:

A) If ELE is replaced with 9, the output of the above program is LLY.

B) If the output of the above program is LLRRN, the value of ELE lies within the range (16, 18)

C) The number of function calls to binarySearch if ELE is replaced with 96 is 5.

D) LRLRY is a valid output of the above program.