Course Name： **Data Structures and Algorithm Analysis B**

Question 1 Matching (20×1 point = 20 points)

(1) I	(2) A	(3) O	(4) G	(5) C
(6) B	(7) D	(8) K	(9) J	(10) N
(11) L,M	(12) E	(13) R	(14) F	(15) T
(16) F	(17) T	(18) Z	(19) X	(20) Y

Question 2 On Big-O (8×1.5 points = 12 points)

(21) $O(N^2 \log N)$

(22) $O(N^8)$

(23) $O(\log N)$

(24) $O(N)$

(25) $O(N^2 \log N)$

(26) $O(N^2 \log^2 N)$

(27) $O(N^{1/2})$

(28) $O(N^{1/4})$

Question 3 Big-Oh and Run Time Analysis (4×3 points = 12 points)

(29) $O(n^3)$

(30) $O(N)$

(31) $O(N^2)$

(32) $O(N^5)$

Question 4 On Quick-sort (2×5points = 10 points)

(33) a[low..high] = [F A E J T O R S X U M]

(Remark: It may have mutiple correct answers. The key points are **J** should be in the 4th position (2 points); **[F A E]** in any order should be on the left of **J** (1.5 points), **[T O R S X U M]** in any order should be on the right of **J** (1.5 points).)

(34) a[low..high] = [J A E F M U S T O R X]

(Remark: It may have mutiple correct answers. The key points are **M** should be in the 5th position(2 points), **[J A E F]** in any order should be on the left of **M** (1.5 points), **[U S T O R X]** in any order should be on the right of **M** (1.5 points).)

Question 5 h-sorting in Shell Sort (3×4 points = 12 points)

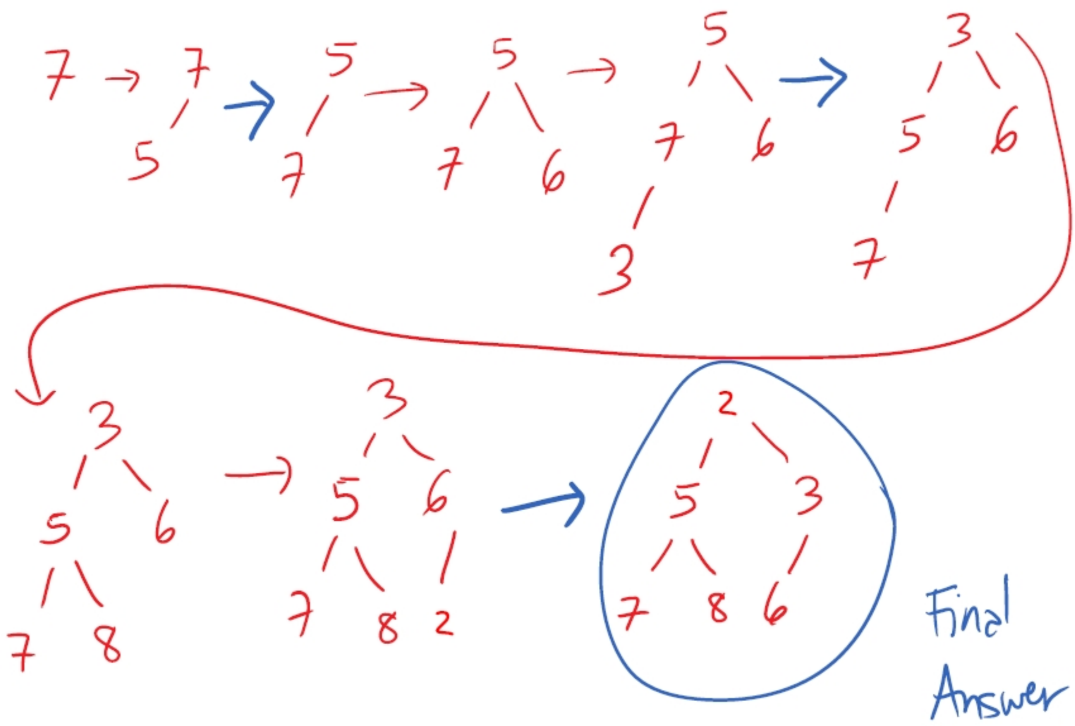
(35) 13-sort: A M S T M I D T E R M E X J U

(36) 4-sort: A I D E E J M T M M S T X R U

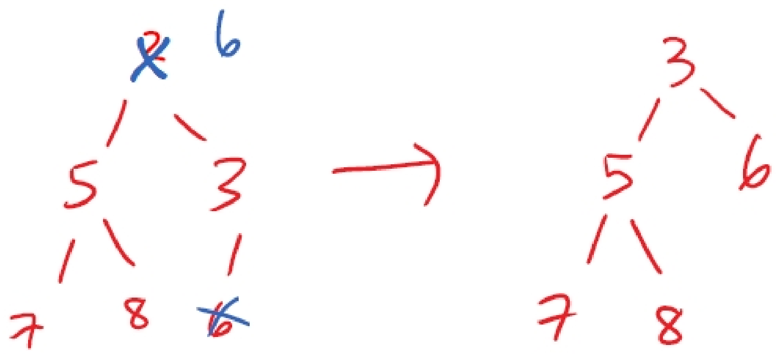
(37) 1-sort: A D E E I J M M M R S T T U X

Question 6 On Binary Min Heaps (7 + 3 = 10 points)

(38) (*7 points)* The constructing procedure of the binary min tree:



(39) (*3 points)* The operation procedure of a deleteMin from the binary min tree in the result of question **(38)** :



Question 7 Programming (3 * 10 points = 30 points)

(40) (*10 points*) The reference program:

```
public static int[] insertionSort (int[] a) {
   for (int i = 1; i < a.length; i++) {
      // YOUR CODE HERE
      for (int j = i; j > 0; j--)
         if (a[j] < a[j-1]) {
            int temp = a[j-1];
            a[j-1] = a[j];
            a[j] = temp;
         } else break;
      // …
   } // end of the for-loop
   return a;
}
```

(41) (*10 points*) The reference program:

```
public static void selectionSort (int[] a) {
   for (int i = 0; i < a.length; i++) {
      int idx = i;  // index_of_least
      int j;
      for (j = i+1; j < a.length; j++) {
         // YOUR CODE HERE #1
         if (a[j] < a[idx])
            idx = j;
         // …
      } // end of the inner for-loop

      // YOUR CODE HERE #2
      int temp = a[i];
      a[i] = a[idx];
      a[idx] = temp;
      // …
   } // end of the outer for-loop
}
```

(42) (*10 points*) Two reference programs are given below:

```
private static <E extends Comparable<E>>
int binarySearch (E x, E[] a, int low, int high) {
   // Tips: use x.compareTo(y) to compare 2 Es

   // YOUR CODE HERE
   if (low > high) return -1;

   int lo = low, hi = high;
   while (lo <= hi) {
      int mid = (lo + hi) / 2;
      int comp = x.compareTo(a[mid]);
      if (comp == 0) return mid;
      if (comp <  0) hi = mid - 1;
      else           lo = mid + 1;
   }

   return -1;
   // …
}
```

```
private static <E extends Comparable<E>>
int binarySearch (E x, E[] a, int low, int high) {
   // Tips: use x.compareTo(y) to compare 2 Es

   // YOUR CODE HERE
   if (low > high) return -1;
   int mid = (low + high) / 2;
   int comp = x.compareTo(a[mid]);
   if (comp == 0) return mid;
   if (comp <  0) return binarySearch( x, a, low, mid-1);
   else           return binarySearch( x, a, mid+1, high);
   // …
}
```