1. Consider two singly linked lists of integers $S1$ and $S2$, and a C function func as given below. Suppose that the elements in $S1$ and $S2$ are *distinct*, and are placed in ascending order in their own lists. Let m and n represent the numbers of elements in $S1$ and $S2$, respectively. (20%)

```
typedef struct list_node *list_ptr;
typedef struct list_node {
                int value;
                list_ptr next;
};
list_ptr S1, S2;
list_ptr out =NULL;
func(list_ptr S1, list_ptr S2)
{
   list_ptr ptr1, ptr2, temp;

   ptr1 = S1;
   while (ptr1 != NULL) {
         ptr2 = S2;
         while (ptr2 != NULL) {
               if (ptr1->value == ptr2->value) {
                  temp = (list_ptr) malloc(sizeof(struct list_node));
                  temp->value = ptr1->value;
                  temp->next = out;
                  out = temp;
               }
               ptr2 = ptr2->next;
         }
         ptr1 = ptr1->next;
   }
}
```

(a) What is the function of func($S1$, $S2$)?

(b) What is the time complexity of func in terms of m and n?

(c) Could you design an algorithm which gives a better time complexity? Explain your algorithm. (Exact C program is NOT necessary, simply explain it.)

(d) What is the time complexity of your algorithm?

2. Consider a set of 5 keys as shown below: (20%)
{58, 50, 55, 30, 64}

   (a) Draw the binary search tree by scanning these keys from left to right.

   (b) Suppose that the probabilities of searching for items with keys 58, 50, 55, 30, 57 and 60 are 0.1, 0.25, 0.15, 0.35, 0.1 and 0.05, respectively.

      i. What is the expected number of key comparisons required for each retrieval (including unsuccessful ones)?

      ii. If the above five items are organized as a linked list and *sequential search* method is used, is it possible to construct a linked list that has an expected number of key comparisons that is smaller than the one using the previous binary search tree? Construct such a list if your answer is yes.

3. Give conditions under which *QuickSort* algorithm has the worst cast behavior in terms of time complexity. (10%)

4. (a) What is the fundamental concept behind the AVL trees? (20%)

   (b) Consider the AVL tree of Figure 1. Show the resulting AVL trees after the following elements are added each.

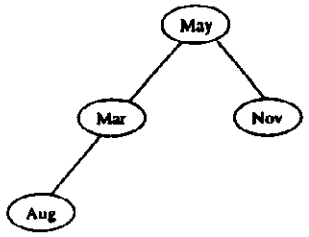

Figure 1:

   i. Apr
   ii. Jan
   iii. Dec
   iv. Feb

所　別：電機工程技術研究所　　組別：計算機組　　科目：資料結構
學程別：

5. Draw the step-by-step results of inserting the integers 1 to 9 into an empty max-heap. (10%)

6. Let $h(x) = x \pmod{10}$ be the hash function for memory spots numbered 0 to 9. Let 31, 63, 13, 69, 4, 19, 29 be the input sequence. Consider the following two collision resolutions: (20%)
   (a) close hash table using linear probing, and
   (b) close hash table using second hash function, $h'(x) = x \pmod{7}$.
   What are the results of these two methods?