中原大學 100 學年度 碩士班 入學考試

3 月 19 日 15:30~17:00　　通訊工程碩士學位學程

誠實是我們珍視的美德，
我們喜愛「拒絕作弊，堅守正直」的你！

科目：資料結構與演算法

□可使用計算機，惟僅限不具可程式及多重記憶者　　■不可使用計算機

1. [15%] provide one application that prefers the data structure or the algorithm described below.

(a) Give an example that prefers **stack** to queue. [3%]

(b) Give an example that prefers **doubly linked list** to single linked list. [3%]

(c) Give an example that prefers **merge sort** to quick sort. [3%]

(d) Give an example that prefers **heap** to AVL-tree. [3%]

(e) Give an example that prefers **B-tree** to hash table. [3%]

2. [10%] given the binary tree with post-order traversal ***fechgdba*** and in-order traversal ***fecabhdg***,

(a) Show the binary tree. [5%]

(b) Represent the binary tree as a unique string (i.e., every binary tree has its own string). [5%]

3. [15%] Fill in the blanks (assumption: the array *A*[*1*…] has kept *N* scores in ascending order).

Algorithm 1: InsertArray(*A*, *x*)
Purpose: Insert score *x* into the array *A*

```
{   for (i = 1; i <= N; i++)
        if (A[i] > x)   break;
    j = N;
    while (j >= i)
    {       (a)  請勿在題目卷上作答 ;
        j = j - 1;   }
    A[i] = x;
    N = N + 1;
}
```

Algorithm 2: DeleteArray(*A*, *x*)
Purpose: Delete score *x* from the array *A*

```
{   for (i = 1; i <= N; i++)
        if (A[i] == x)   break;
    if (i <= N)
    {       j = i + 1;
        while (j <= N) and (A[j] == x)
            j = j + 1;
        while (j <= N)
        {       (b)  請勿在題目卷上作答 ;
            i = i + 1;
            j = j + 1;
            N = N - 1;      }     }
}
```

Algorithm 3: CountArray(*A*)
Purpose: Compute the number of distinct scores in the array *A*

```
{   pre = -1;   // assume all nonnegative scores!
    count = 0;
    for (i = 1; i <= N; i++)
        if (A[i] > pre)
        {   count = count + 1;
            (c)  請勿在題目卷上作答 ;     }
    return count;
}
```

Algorithm 4: ReverseArray(*A*)
Purpose: Make the array *A* in descending order

```
{   i = 1;
    j = n;
    while (  (d)  請勿在題目卷上作答 )
    {   temp = A[i];
        A[i] = A[j];
        (e)  請勿在題目卷上作答 ;
        i = i + 1;
        j = j - 1;   }
}
```

中原大學 100 學年度 碩士班 入學考試

3 月 19 日 15:30~17:00　　通訊工程碩士學位學程

誠實是我們珍視的美德，
我們喜愛「拒絕作弊，堅守正直」的你！

科目：資料結構與演算法

□可使用計算機，惟僅限不具可程式及多重記憶者　　■不可使用計算機

4. [15%] Fill in the blanks (assumption: the linked list *L* is sorted in ascending order of the score).

Algorithm 1: SearchList(*L*, *x*)
Purpose: Find the node with score *x* in *L*

```
{   pre = null;
    cur = L;
    flag = false;
    while (cur ≠ null)
        if ((*cur).score < x)
        {       pre = cur;
                (a)   請勿在題目卷上作答 ;   }
        else  break;
    if (cur ≠ null)
        if ((*cur).score == x)
                flag = true;
    return (pre, cur, flag);
}   // cur points to the first node with score >= x!
```

Algorithm 2: DeleteList(*L*, *x*)
Purpose: Delete score *x* from *L*

```
{   (pre, cur, flag) = SearchList(L, x);
    if (flag == false)   return;
    temp = cur;
    cur = (*cur).link;
    while (cur ≠ null)
        if ((*cur).score == x)
                cur = (*cur).link;
        else  break;
    if (pre == null)
        L = cur;
    else
            (b)   請勿在題目卷上作答 ;
    free(temp);   // remove the unused nodes
}
```

Algorithm 3: InsertList(*L*, *x*)
Purpose: Insert a node with score *x* into *L*

```
{   (pre, cur, flag) = SearchList(L, x);
    y = NewNode();   // create a new node!
    (*y).score = x;
    (*y).link = cur;
    if (pre == null)
        (c)  請勿在題目卷上作答 ;
    else
        (*pre).link = y;
}
```

Algorithm 4: ReverseList(*L*)
Purpose: Make *L* in descending order

```
{   if (L == null)          return;
    if ((*L).link == null)  return;
    pre = null;
    cur = L;
    do {   L = (*cur).link;
           (d)   請勿在題目卷上作答 ;
           pre = cur;
           cur = L;
    } while (cur ≠ null );
    (e)   請勿在題目卷上作答 ;
}
```

5. [15%] Heap is a useful data structure in search and sort. Answer the following questions about it:

(a) List the necessary properties of a heap structure. [3%]

(b) Write a C/C++ program to store *N* numbers as a heap. [6%]

(c) Write a C/C++ program to sort the *N* numbers in the heap. [6%]

6. [10%] given two strings A and B, answer the following questions:

(a) Write a C/C++ program to find the longest common substring between A and B. [6%]

(b) Express the time complexity of your program. [4%]

中原大學 100 學年度 碩士班 入學考試

3 月 19 日 15:30~17:00　　通訊工程碩士學位學程

誠實是我們珍視的美德，
我們喜愛「拒絕作弊，堅守正直」的你！

科目：資料結構與演算法

□可使用計算機，惟僅限不具可程式及多重記憶者　　■不可使用計算機

7. [10%] Huffman codes are widely used for data compression. Suppose we have a 1000-character data file that we wish to store compactly. The text file contains only 6 distinct characters, i.e., {A, B, C, D, E, F}, and the corresponding frequencies are given in the following table.

character	A	B	C	D	E	F
frequency	0.18	0.05	0.15	0.22	0.32	0.08

(a) Illustrate how to build Huffman's tree and derive your Huffman codes. [5%]

(b) Use your Huffman codes to encode the word "FACE". [2%]

(c) How many bits are required to store the data file if Huffman codes are used? [3%]

8. [10%] given a computer network (assumption: two computers have at most one connection line) in which each connection line is associated with the (estimated) transmission time,

(a) List two data structures that are typically used to represent the network. [4%]

(b) Design an algorithm to find the fastest path from one computer to another. [6%]

(Note: Write your algorithms in the form of pseudo-codes together with detailed comments.)