國立中正大學電機、通訊工程學系平時考試答案卷

考試科目：

班別：

學　　號：406415054

姓名：傅心[illegible]

自虛線以下開始作答：

8.5

1.

	i	count	做 $i=i+8$ 後的 i
1^{st} 進 while	0	0	8
2^{nd}	8	8	16
3^{rd}	16	$24 = 8\times 3 = 8\times(1+2)$	24
4^{th}	24	$48 = 8\times 6 = 8\times(1+2+3)$	32
+2 5^{th}	32	$80 = 8\times 10 = 8\times(1+\cdots+4)$	40
6^{th}	40	$120 = 8\times 15 = 8\times(1+\cdots+5)$	48
⋮			
n^{th}	$8(n-1)$	$8\times[1+\cdots+(n-1)]$	$8n$ ✓

$(n+1)^{th}$ 不會進 while ~~$8n$~~

令 $N = size = 8n$ $\Rightarrow$ 次數 $= n = \frac{N}{8}$ $\quad O(\frac{N}{8}) = O(N)$ #

2、 $\underline{size = 100}$

$n = \left\lceil \frac{100}{8} \right\rceil = 12$

+0

count 最終 $= 8\times[1+\cdots+\overset{12}{\cancel{11}}]$

$= 8\times\frac{12\times\overset{13}{\cancel{11}}}{2} = 8\times 6\times\overset{13}{\cancel{11}} = \overset{624}{\cancel{528}}$

?

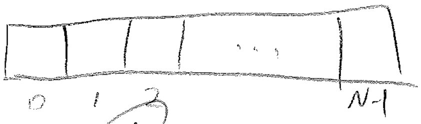
$$\begin{array}{r} 48 \\ \times\ 13 \\ \hline 144 \\ 48\ \ \\ \hline 624 \end{array}$$ ✓

$\Rightarrow$ count is $\overset{624}{\cancel{528}}$ #

105.6.10,000

3.

~~若在~~第



```
void fun ( )
{   int key , i=0, a[N] ;
    scanf ("%d", key ) ;
    while ( i < N && key ≠ a[i] )
    {      i++ ;       }

    if ( i == N )  printf (" Not found\n") ;

    else  printf (" %d is at position %d\n", key , i ) ; // position 幾
                                                           表示 a[幾]
    return ;   }
```

+3.5

若在 a[0] 找到：判斷 1 次

a[1] 2.

⋮

a[N-1] N

<ps> 找不到 N+1

平均判斷次數：

$$\frac{1+\dots+N}{N} = \frac{N(N+1)}{2N} = N+1 \text{ 省略}$$

⟹ 歸類在 O(N)

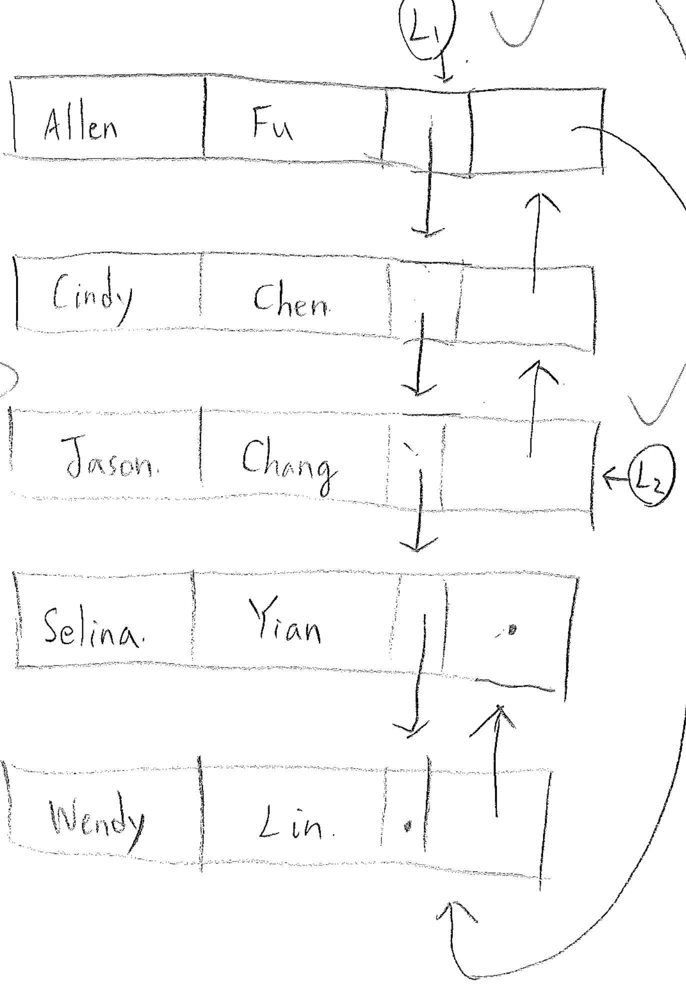
<ps.> 若把"找不到"也列入考量

平均次數：$\frac{1+\dots+(N+1)}{(N+1)}$

$$= \frac{(N+1)(N+2)}{2(N+1)} = N+2 \Rightarrow O(N)$$ 仍為

4. i.

+3



```
ii. struct node {. char FirstName [20] ;
                   char LastName [20];
                   Struct node * Link 1 ;
       ✓           struct node * Link 2;  }
```

Student ID: 40641505 4 Name: 傅心琁

Data Structures
Fall 2018 Quiz 2

1. (2 Points) Analyze the following algorithm for its complexity in the Big-O notation. Analyze the possible costs, sum up the costs, derive the formula for the costs, and categorize it to O(f(n)). Show the calculation details. The data size is N.

8 40

```
void do_something(int size)
{                        40
    int i=0, count=0;

    while (i<size) {     i= 0   count :
        count += i;        8
        i = i + 8;         16
    }                         24
    printf("count is %d\n", count);32
}
```

C=1 i= 8
2 16
3 24
4 32
5 40

2. (1 point) What is the output of the above function if the input to size is 100? $\frac{100}{8}$

3. (4 Points) Write a sequential search function in C and analyze its complexity in the Big-O notation. Show the detailed calculation of your analysis. The data size is N.

4. Consider a multi-value linked list. A sample node is given below. The first two fields are strings to store the first name and the last name of a person. The third and the fourth fields are for pointers to point to the next node. The first pointer (Link1) makes a sorted list according to the field FirstName, and the second pointer (Link2) makes a sorted list according to the field LastName.

FirstName	LastName	Link1	Link2

last

i. (1 point) List five names and draw a multi-value linked list with five nodes connected to each other.

ii. (2 points) Define the above node as a data type in C.

first

7

1	2	3	5 7	

96
× 81
5286

12.44

88
× 6
528

$\frac{n(n+1)}{2}$

8×(1+...+11)

Student ID: 406415054 **Name:** 傅心瑄

Data Structures
Fall 2018 Quiz 4

李石根

1. 1 (1 pt) Given N nodes, what is the maximum height and minimum height of a binary tree? Explain your answer.
2. 1 (1 pt) Given the height H, what is the maximum and minimum numbers of nodes in a binary tree? Explain your answer.
3. 2 (2 pts) Draw the expression tree for the following expression and write the result of performing the **Postorder** traversal.

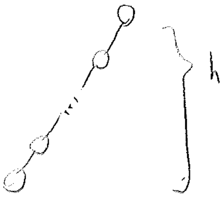
   Z – (Y * X) + (W / V * (U) – T 〈注意优先〉 ⟶ (背面)

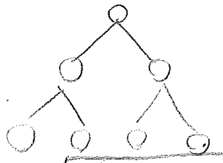
4. Use the type definition and function prototypes below and complete these three functions in C:
   - i. 2 (2 pts) initQ () which prepares the queue head by initialize Count to 0 and Front and Rear to NULL
   - ii. 2 (2 pts) enqueue () which enqueues (inserts) a data to the queue and updates the count and pointers
   - iii. 2 (2 pts) dequeuer () which dequeues (removes) the front data as the return value and updates the count and pointers

```
typedef struct list {
    int data;
    struct list * link;
    } listType;

typedef struct head {
    int count;
    listType * front;
    listType * rear;
    } qHead;

qHead * initQ (void);
void enqueue (qHead *, int);
int dequeue (qHead *);

int main(void)
{
    qHead * Q;

    Q = initQ ();
    enqueue (Q, 123);
    enqueue (Q, 98765);
    enqueue (Q, 2468);
    while (Q->count > 0) {
        printf("Content of Queue is:\n");
        printf("%d\n", dequeue (Q));
    }
}
```

1、Max: N min: $\lfloor \log_2 N \rfloor + 1$

chain-like tree 完整の binary tree.

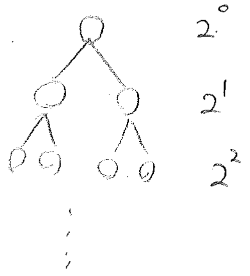


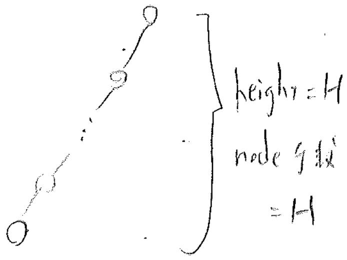


(說明在背面)

2、Max: $2^H - 1$ min: H

完整 tree chain-like tree





2^{H-1}

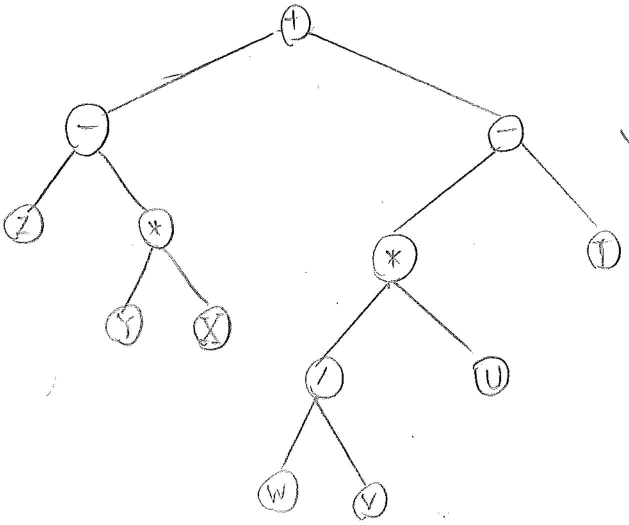
$2^0 + 2^1 + \cdots + 2^{H-1} = 2^H - 1$

4、

```
qHead * initQ ( )
{
    qHead *Q = (qHead *) malloc (sizeof (qHead));
    Q -> count = 0;
    Q -> front = NULL; Q -> rear = NULL; return Q;
}

void enqueue (qHead *Q, int data)
{
    listType * current = (listType *) malloc (sizeof (current));
    current -> data = data; current -> link = NULL;
                              定義 front 、rear
    if (Q->count == 0) { Q-> front = current; Q -> rear = current; }
    else { (Q->rear) -> link = current; Q -> rear = current; } (Q-> count)++;
                  串上去                      更新 rear
}
```

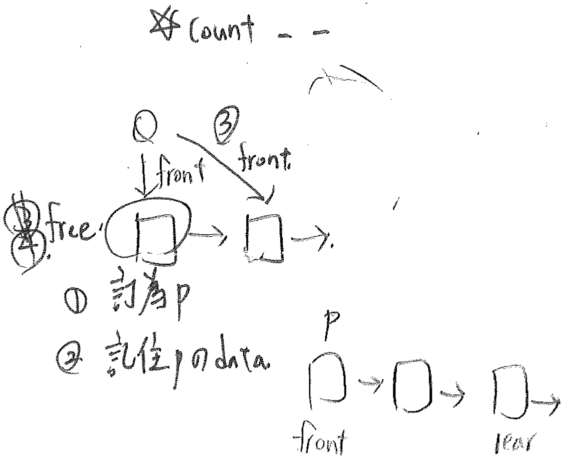
(dequeue 在背面)

3、



Z Y X * - W V / U * T - +

```
4. int dequeue ( qHead * Q )
   {    int temp ;   listType * p ;
        p= Q->front ;   temp= p->data ;
        Q-> front= p->link ;
        free (p);   (Q->count) -- ;
        return data ;
   }
```



1. min = $\lfloor \log_2 N \rfloor + 1$;

	N	H
$\lfloor \log_2 N \rfloor = 0$	1	1
$\lfloor \log_2 N \rfloor = 1$	2	2
	3	2
	4	3
$\lfloor \log_2 N \rfloor = 2$	5	3
	6	3
	7	3
	8	4
$\lfloor \log_2 N \rfloor = 3$	9	4

⇒ 可观察出 $H = \lfloor \log_2 N \rfloor + 1$

Student ID: 406415054 **Name:** 傅[illegible]

Data Structures
Fall 2018 Quiz 5

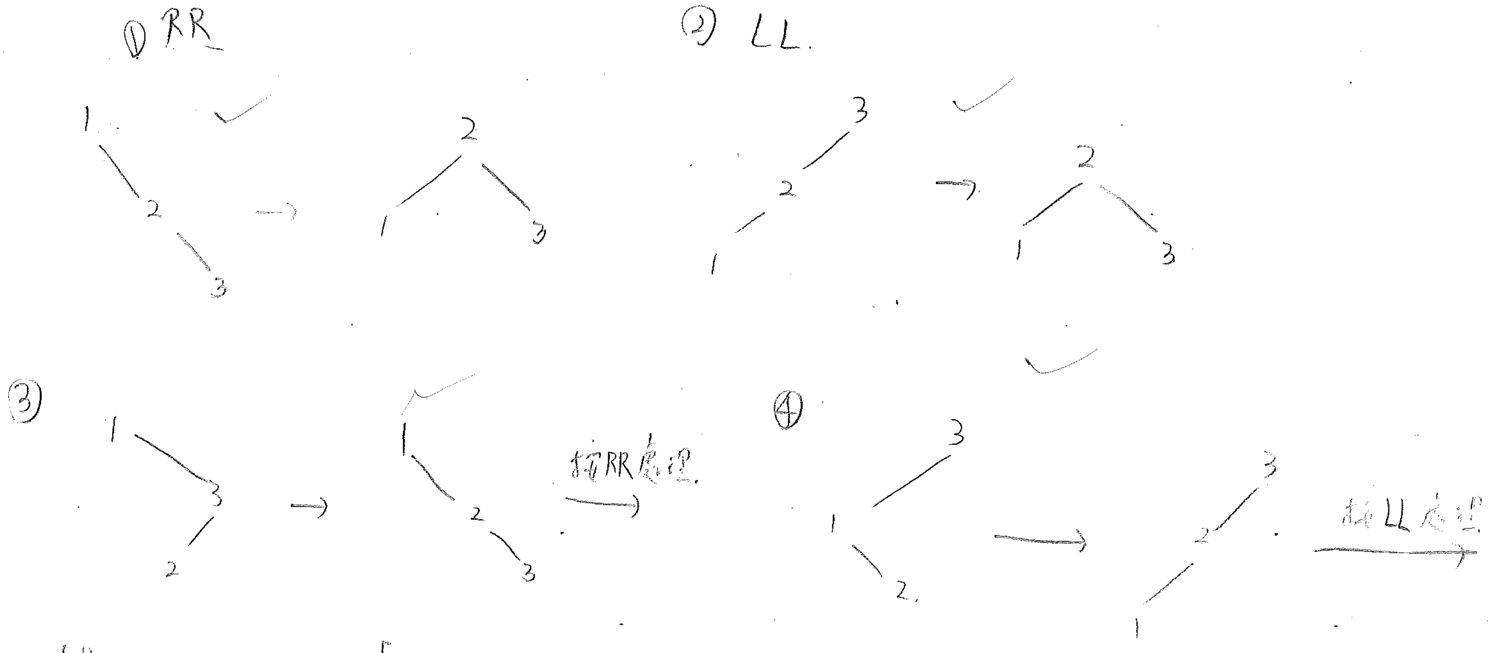
10

v 1. 2 (2 pts) Define the AVL tree. How is a binary search tree (BST) updated to the AVL tree?
v 2. 1 (1 pt) Define the data structure to the node in a BST using C.
v 3. 3 (3 pts) Using C and the data structure above, write the code to the insert function of a BST.
✓ 4. 1 (1 pt) Explain how the delete function of a BST works. why need.
5. 3 (3 pts) Write the search algorithm of a BST. Analyze its complexity and give the result in the Big-O notation. Show detail calculation with your explanation. code. 要分析. (~~好~~,坏,均)

1. BST: 每个node底下最多2个 subtrees, 且中序排列时会由小至大. 且右子 > 左子 ??

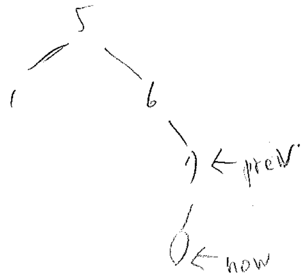
AVL: 一种特殊の BST, 其 |左subtree高度 − 右 subtree高度| ≤ 1 ✓
若 > , 则調整

update 方法: 当 ... 不成立时, 作調整.



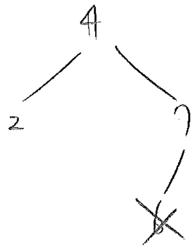
2.
```
typedef struct NODE {
    int data;
    struct NODE * l;
    struct NODE * r;
} node;
```

3.
```
void insert (node * root, int data)
初始化 { node * ptr = malloc (sizeof ( node)) ; node * prev, * now;
       ptr -> data = data; ptr -> l = ptr -> r = NULL;

若根空  if (root == NULL) { root = ptr ; }
        else {   now = root ;
            while (now != NULL)
prev = seek  {   prev = now ;
prev · seek      if (ptr -> data > now -> data) { now = now -> r ; }
一搭一唱          else { now = now -> l ; }
             }
             if (ptr -> data < ~~seek~~ prev -> data) { prev -> l = ptr ; }   插入
             else { prev -> r = ptr ; }
背面还有
```



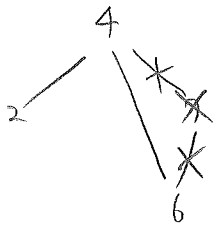
}

4. case 1: 刪 leaf node：直接刪



✓

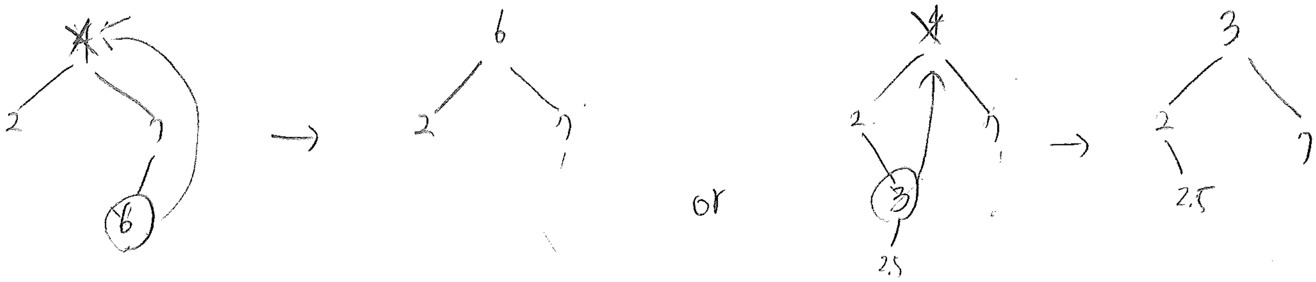
case 2: 刪 leaf-like node：繞線
(直接繞过被刪者)



✓

case 3: 其餘：找右 subtree の min or 左 subtree の Max 来取代
(∵找最鄰近者取代，如此一来整体の中序排列会保持合法)

✓



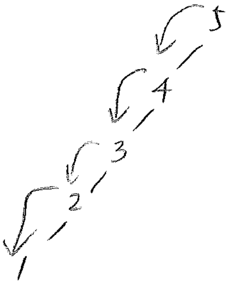
图中，被圈起の替代節点，须刪除，其方法比照 case 1 or 2.

5.
```
node * search (node * root , int data)
{   while (root != NULL && data != root -> data) ← 注意顺序
    { if (data > root -> data) { root = root -> right ; }
        else  { root = root -> left ; }
    }
    if (root == NULL) { printf ("Not found. "); }
    return root ;
}
```

✓

最坏: chain-like tree.

有 N 个 node ⇒ 找 N 次



視同 sequential search

⇒ O(N) #

平均: 原本有 N 个 node.

搜寻1次後: $\frac{N}{2}$ 个 node

2 $\frac{N}{2^2}$

3 $\frac{N}{2^3}$

⋮

k $\frac{N}{2^k}$

when $\frac{N}{2^k} = 1$ ⇒ 找到

$N = 2^k$

$k = \log_2 N$

⇒ $O(\log N)$ #

Student ID: 40615054 **Name:** 傅心綺

Data Structures
Fall 2018 Quiz 6

大小 N+1

✓ 1. 1 (1 pt) Explain the property of a heap (max heap) and how this can be implemented in an array.
✓ 2. 1 (1 pt) Consider a heap in an array X[0..N]. Where is the parent located for the node X[i]? Where are the children?
✓ 3. 2 (2pts) Consider an array of size 6 below. Complete the content of array in each phase of heap construction. The shaded area is the unprocessed data, and the white area is the heap in each phase.

~~i=0~~ Phase 0

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
10	101	50	33	85	120

i=0 ~~i=1~~ Phase 1

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
10	101	50	33	85	120

i=1 Phase 2

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
101	10	50	33	85	120

i=2 Phase 3

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
101	10	50	33	85	120

i=3 Phase 4

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
101	33	50	10	85	120

Phase 5

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
101	85	50	10	33	120

Phase 6

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
120	85	101	10	33	50

✓

加入: check with parent (up)
刪根: check with larger child (down)
up

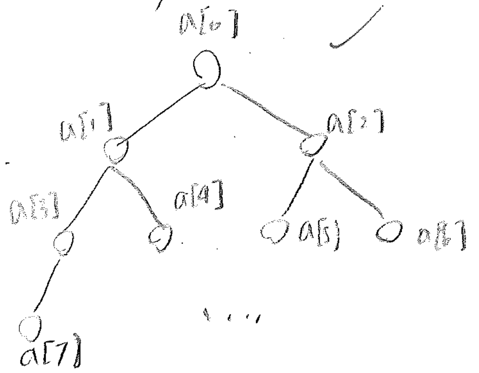
✓ 4. 1 (1 pt) Draw the above heap according to the content of the array in Phase 6.
5. 3 (3 pts) Using C to write the code for constructing a heap using the process defined in Question 3. → 非 heap sort
6. 1 (1 pt) Assume you have N numbers. How can you use a heap to find the k largest numbers? Explain and use pseudo code to write your algorithm. → heap sort

delete down ward

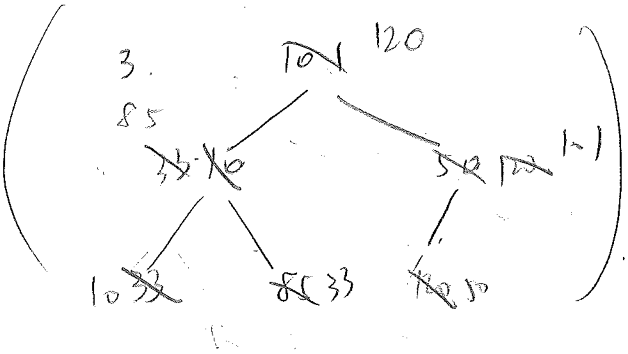
1. for all subtrees, value of root = MAX
∵ Complete / nearly complete tree.
∴ 適合用 array

2. parent: $X[\frac{i-1}{2}]$

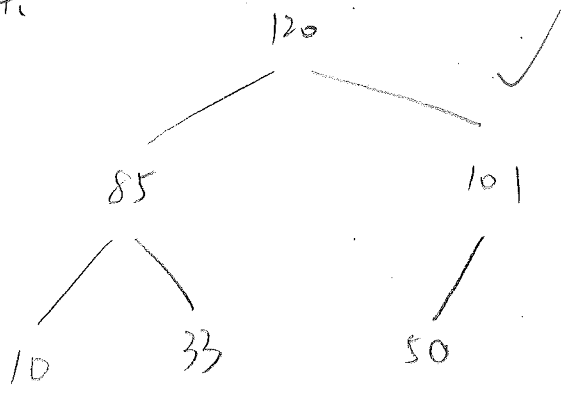
children $X[2i+1]$, $X[2i+2]$



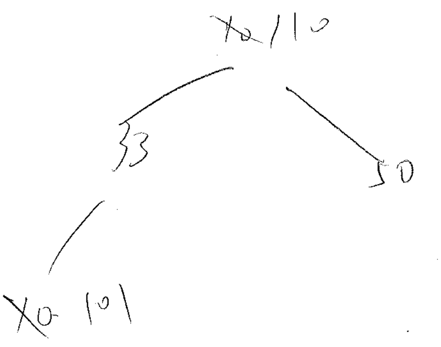
✓



4.



✓



```
5.  void ~~heap-sort~~ (int A[6])        ?
    {  int i;

       for (i = 1; i<6; i++)
       {
                                          ✓
            upward (A, i, A[i]);
       }
    }

    void upward (int A[6], int loc, int data)
    {   ~~int i=0~~; int done =0;
        A[loc] = data;
                                         子          父
        while (loc >0 && done == 0)
check   {   if A[loc] < A[(loc-1)/2]
with            done = 1;
parent
            else { swap &A[loc], &A[(loc-1)/2] ];
                   loc = (loc-1)/2;
                 }
        }

    }

    void swap (int *b, int *c)
    {  int temp;
       temp = *b;
       *b = *c;   *c = temp;
    }
```

(尋找子節)

repeat k times ✓

6. 將N个數建成一heap，把它の根の值取出，刪除根 (利用downward法 update)

```
int downward ( heap[], N data)
{  int i, j, done ← FALSE
   data ← A[0]
   A[0] ← A[N]
   N ← N-1        i←0
   while (i*2+1 <= ~~k~~N && done ← FALSE)
   {  j = larger( heap, i*2+1, i*2+2)
check   if(A[i] > A[j] )
with       父      子
larger     done ← TRUE
child
      else { swap (&A[i] &A[j] )
             i = j;
           }
   }
}

void search (int A[N])
{  loop (k次)
   {  printf ("%d \n", downward
      (A, N, data) ;
   }
```

Student ID: 406415054 **Name:** 傅心旋

Data Structures
Fall 2018 Quiz 7

Q6.7 code 未

1. 2 (2pts) Consider an array of size 6 below to perform **Insertion Sort**. Complete the content of array after completing a process in each phase of sorting. The shaded area is the unprocessed data, and the white area is the sorted sublist.

跑前

	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
Phase 0	101	10	50	33	85	100
Phase 1	10	101	50	33	85	100
Phase 2	10	50	101	33	85	100
Phase 3	10	33	50	101	85	100
Phase 4	10	33	50	85	101	100
Phase 5	10	33	50	85	100	101
Phase 6	10	33	50	85	100	101

無斜線部份,是一 ordered list

每次均把key往前放至無斜線區

無斜線區中, >key者:右移

<key者:不動

空出の位置,填入key

跑後

2. 3 (3 pts) Using C to write the code for performing **Insertion Sort** using the process defined in Question 1. 背面

3. 2 (2pts) Consider an array of size 6 below to perform **Selection Sort**. Complete the content of array after completing a process in each phase of sorting. The shaded area is the unprocessed data, and the white area is the sorted sublist.

	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]
Phase 0	101	10	50	33	85	100
Phase 1	10	101	50	33	85	100
Phase 2	10	33	50	101	85	100
Phase 3	10	33	50	101	85	100
Phase 4	10	33	50	85	101	100
Phase 5	10	33	50	85	100	101
Phase 6	10	33	50	85	100	101

每次由斜線區中,挑出min,往前放至白色區の最末格の下一格

宣告　　**Student ID:**　　swap測試　　**Name:**

4. 3 (3 pts) Using C to write the code for performing **Selection Sort** using the process defined in Question 3.

Q2 & Q4

```
void insertion (int A[6], int size) //Q2
{   int k=1; int i, key;
    while (k < size)
    {   key=A[k];  i=k;
        while (i>0 && A[i-1] < key)
        {   A[i] = A[i-1];    要跳出
            i = i-1;
        }
        A[i] = key;
        k++;   ✓
    }
}
```

```
int main ( )
{

int A[6] = {101, 10, 50, 33, 85, 100};
//  selection( A, 6 );
    insertion ( A, 6 );
    return 0;

}
```

輪流
註解掉

```
void selection (int A[], int size) //Q4
{   int k=0;  int i, min, minIndex;
    while (k<size-1)
    {  min = A[k];  minIndex= k;  i=k+1;
       while (i < size)
       {   if (A[i] < min)
           {   min=A[i];
               minIndex = i;    ✓ 不可跳出
           }  i++;
       }
       swap (A[k ], A[minIndex]);
       k++;
    }
}
void swap (int b, int c )
{   int temp;
    temp = b;
    b=c;  c= temp;
}
```

双loop、用行為想

called by ref?