【不可使用計算機】＊作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！！

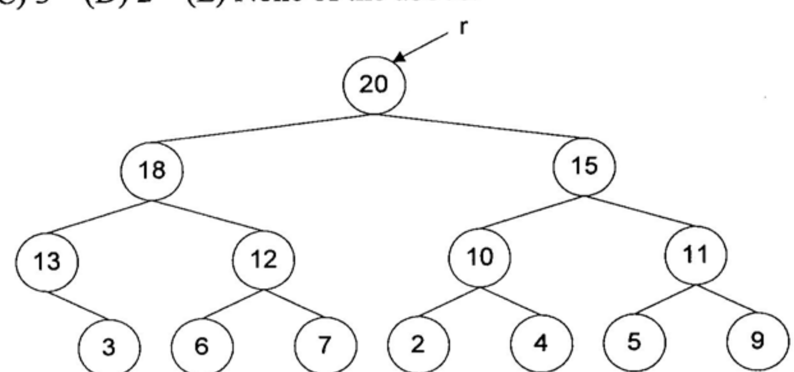
注意：每大題的小題必需全對該大題才給分。每大題5分。請使用答案卡作答

1. Suppose that we use *struct node* to represent binary trees, and use function *unknown* to process a binary tree.

```
struct node
{
  int data;
  struct node * left, * right;
};
struct node * r;
void unknown(struct node *p)
{
  struct node * q;
  if (p->left!=NULL)
    unknown(p->left);
  if (p->right!=NULL)
    unknown(p->right);
  q=p->left;
  p->left=p->right;
  p->right=q;
}
```

(1) Which order does function *unknown(r)* use to traverse the tree with root node pointed by the pointer *r*?
(A) preorder (B) inorder (C) postorder (D) level order (E) None of the above.

(2) Consider the following binary tree. After the execution of function *unknown(r)*, what is the value of the data of the rightmost node?
(A) 7 (B) 6 (C) 3 (D) 2 (E) None of the above.



(3) Suppose the binary tree is of *n* nodes. What is the time complexity of function *unknown*?
(A) O(log n) (B) O(n) (C) O(nlog n) (D) O(n^2) (E) None of the above.

2. Suppose that the values of *a, b, c, d* are 2, 4, 6, 8, respectively. Please answer the following questions.

(4) What is the result of the expression " ab+c*d-"?
(A) 36 (B) 28 (C) 44 (D) 18 (E) None of the above.

(5) What is the result of the expression "-+*abcd"?
(A) 8 (B) 10 (C) 12 (D) 14 (E) None of the above.

(6) Suppose that we use a stack to transform an infix expression with *m* operands and *n* operators. What is the size of the stack?
(A) O(n/m) (B) O(n) (C) O(m+n) (D) O(mn) (E) None of the above.

【不可使用計算機】＊作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！！

3. Considering the following program segment, please answer the following questions.

```
int a[]={8, 26, 5,77,1,61,11, 60, 15, 49, 19};
int n=11;

int ptt(int x)
{
  int k=-1;
  int i=0;
  int j=n-1;
  do  {
    k=(i+j)/2;
    if (a[k]<=x)
      i=k+1;
    else
      j=k-1;
  } while (i<=j);
  return k;
}
```

(7) What is the result of *ptt(1)*?
(A) 4 (B) 0 (C) 3 (D) 9 (E) None of the above.

(8) What is the result of *ptt(10)*?
(A) 4 (B) 0 (C) 3 (D) 9 (E) None of the above.

(9) What is the time complexity of the function *ptt*?
(A) O(log n) (B) O(n) (C) O(nlog n) (D) O(n^2) (E) None of the above.

4. Consider the following program. Please answer the following questions.

```
#include <stdio.h>
int data[]={8, 26, 5,77,1,61,11, 60, 15, 49, 19};
int n=10;
void a(int r)
{
  int e, j;
  e=data[r];
  for(j=2*r;j<=n;j*=2)  {
    if (j<n&&data[j]<data[j+1]) j++;
    if (e>=data[j]) break;
    data[j/2]=data[j];
  }
  data[j/2]=e;
}
int main(void)
{
  int i;
  for(i=n/2;i>=1;i--)
    a(i);
}
```

(10) What is the result of *data[10]*?
(A) 11 (B) 1 (C) 15 (D) 26 (E) None of the above

(11) What is the result of *data[3]*?
(A) 77 (B) 8 (C) 61 (D) 60 (E) None of the above

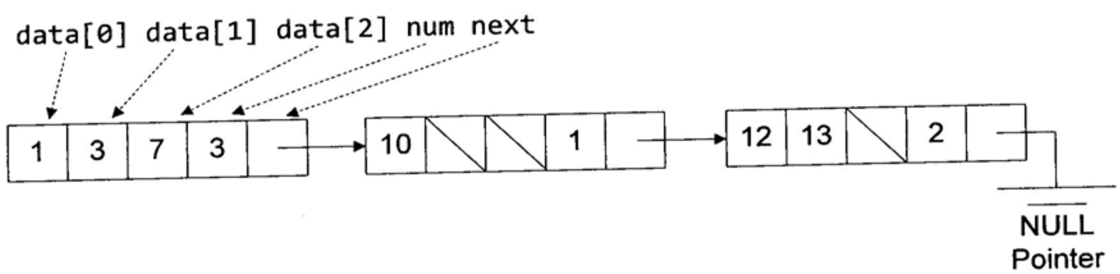
(12) What is the time complexity of the program where *n* is the number of integers in the array *data*?
(A) O(log n) (B) O(n) (C) O(nlog n) (D) O(n^2) (E) None of the above.

5. An *NCTU linked list* with order *m* is a variation of the linked list satisfying the following constraints:

- Each node is able to store at most *m* integers.
- The integers within one node are in ascending order.
- Each integer in an NCTU linked list is distinct.
- If there is a path from node p to node *q*, the integers stored in node *p* are smaller than the integers stored in node *q*.

We use the following example to show how we implement an NCTU linked list with order 3. Each node of NCTU linked list is implemented by *struct node*, where (1) *num* is used to indicate the number of integers within the node, (2) *data[0]*, *data[1]*, …, *data[num-1]* are used to store such *num* integers and (3) *next* is used to point to the next node.

```
struct node
{
  int data[3];
  int num;
  struct node * next;
};
```



Example of an NCTU linked list with order 3

We now use the above approach to implement an NCTU linked list with order *m* and *m>2*. The NCTU linked list is used to store *n* integers and each node is of *m/2* integers. Assume that *m* is even and *n* is dividable by *m*. Please answer the following questions.

(13) Suppose an integer *x* is in the NCTU linked list, what is the time complexity to find the location of the integer *x*?
(A) O(logm*n/m) (B) O(logm+n/m) (C) O(m+n/m) (D) O(logm*n/m+m) (E) None of the above.

(14) Suppose an integer *x* is not in the NCTU linked list, what is the time complexity to insert the integer *x* into the NCTU linked list?
(A) O(logm*n/m) (B) O(logm+n/m) (C) O(m+n/m) (D) O(logm*n/m+m) (E) None of the above.

(15) Suppose an integer *x* is in the NCTU linked list, what is the time complexity to delete the integer *x*?
(A) O(logm*n/m) (B) O(logm+n/m) (C) O(m+n/m) (D) O(logm*n/m+m) (E) None of the above.

6. A job priority queue is implemented using a Min-Heap in which a lower key value means a higher priority. The jobs are entered and stored in the Min-Heap as shown in the following array **Q.**

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
keyvalue	--	6	8	10	12	24	15	13	20	18	26						

(16) Next job is extracted from the job queue for execution. What is the value of Q[4] in the remaining job queue?
(A) 15 (B)18 (C) 20 (D).26 (E) None of the above

(17) After step (16) is executed, next job is extracted from the job queue for execution. What is the value of Q[5] in the remaining job queue?
(A)13 (B) 15 (C) 18 (D) 24 (E) None of the above

(18) After step (17) is executed, a new job with priority 11 is inserted into the job queue. What is the value of Q[9] in the remaining job queue?
(A)18 (B) 20 (C) 24 (D) 26 (E) None of the above

7. When a data set is too large to be accommodated in internal memory, a multi-way (say m-way or order m) search tree is used as an indexing structure and is stored on a disk. Assume the size of each node on the disk is 512 bytes. Assume each node contains key values (each key 5 bytes), and pointers to subtree nodes(each pointer 7 bytes).

(19) What is the order m of the m-way tree ?
(A) 45 (B) 50 (C) 43 (D) 44 (E) None of the above

(20) If the height of root node is 1, what is the maximum number of keys in an m-way index tree of height h?
(A) $m(m^h-1)$ (B) m^h-1 (C) $(m^h-1)/(m-1)$ (D) $m(2^h-1)$ (E) None of the above

(21) For a data set of N records stored in an m-way balanced search tree, what is the time complexity to search a key value in terms of the number of disk accesses in the worst case ?
(A) $O(N/m \log_2 N)$ (B) $O(N \log_m N)$ (C) $O(N/m \log_m N/m)$ (D) $O(\log_m N)$
(E) None of the above

8. A symbol table is implemented using a Hash Table of table size 11 with hash function h (k) = k mod 11, where k is the key of a record.

(22) The records with keys :13, 14, 2, 11, 18, 3, 36, 28 , 21 are inserted in sequential order into the hash table using linear probing method for collision resolution. What is the total number of identifier comparisons if each key value is searched once.
(A) 14 (B)16 (C)18 (D)20 (E) None of the above

(23) As the Hash Table in step (22), if keys 25 and 24 are searched, what is the total number of identifier comparisons?
(A) 11 (B)13 (C)15 (D)17 (E) None of the above

(24) The records with keys :13, 14, 2, 11, 18, 3, 36, 28 , 21 are inserted in sequential order into Hash Table using chaining method instead for collision resolution. What is the total number of identifier comparisons if each key value is searched once.
(A) 11 (B)13 (C)15 (D)17 (E) None of the above

9. The pre-order traversal of a binary search tree (BST) is 26, 20, 15, 19, 24, 22, 48, 36, 30, 64 and its in-order traversal is 15, 19, 20, 22, 24, 26, 30, 36, 48, 64. When the element to be deleted is in a nonleaf node that has two children, the element is replaced by the largest element in its left subtree.

(25) When a key value 37 is inserted into the BST, what is the 6^{th} key value in the post-order traversal of the updated BST?
(A)24 (B)37 (C) 30 (D)26 (E) None of the above

【不可使用計算機】*作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！！

(26) After step (25) is executed, the key value 48 is deleted from the BST, what is the 7^{th} element in its pre-order traversal of the updated BST ?
(A) 26 (B) 36 (C)22 (D) 37 (E) None of the above

(27) After step (26) is executed, the key value 26 is deleted from the BST, what is the fourth element in its post-order traversal of the updated BST?
(A) 19 (B)20 (C)22 (D)24 (E) None of the above

10. Suppose you are to sort 12 numbers in the range [0, 999] using RadixSort. The input list is linked as L = [18, 203, 16, 30, 123, 521, 63, 528, 210, 216, 941, 55] .

(28) At the end of first pass of RadixSort, the 7^{th} element of the resulting chain is
(A)123 (B) 63 (C) 55 (D) 18 (E)None of the above

(29) At the end of 2^{nd} pass of RadixSort, the 7^{th} element of the resulting chain is
(A)216 (B) 63 (C) 16 (D)123 (E)None of the above

(30) At the end of 3rd pass of RadixSort, the 7^{th} element of the resulting chain is
(A)216 (B)203 (C)210 (D) 123 (E)None of the above

11.

(31) Consider the following algorithm. Algorithm Sort(A, n)
Input: Array A containing n different integer values.
Out: Array A sorted in increasing order of values.

```
For i ← n – 1 downto 1 do {
  m ← i
      (*)
      t ← A[m];   A[m] ← A[i];   A[i] ← t // Swap A[i] and A[m]
      }
  } return A
```

Which of the following instructions must be inserted at the point marked (*) so that the algorithm correctly sorts the values stored in A in increasing order of values?

(A) for j ← 0 to n – 1 do
    if A[j] > A[m] then m ← j
(B) for j ← i + 1 to n – 1 do
    if A[j] > A[m] then m ← j
(C) for j ← 0 to i – 1 do
    if A[j] > A[m] then m ← j
(D) for j ← i + 1 to n – 1 do
    if A[j] < A[m] then m ← j
(E) for j ← 0 to n – 1 do
    if A[j] < A[m] then m ← j

(32) Assume that quicksort is being used to sort an array A with values 4 2 8 7 3 1 5 6. After the first invocation to the partition algorithm the array looks like this:
2 3 1 4 7 5 8 6
Which value could have been chosen as the pivot?
(A) 2 (B) 3 (C) 4 (D) 7 (E) 8

(33) Sorting 6 elements with a comparison sort requires at least how many comparisons in the worst case.
(A) 6 (B) 7 (C) 8 (D) 9 (E) 10

12.

(34) A divide-and-conquer algorithm typically makes use of

(A) hashing
(B) recursion
(C) iteration through looping
(D) floating-point division
(E) pipeline

(35) Consider the following C++ function that uses a divide and conquer approach to calculate the sum of a range of numbers.

```
int sum(int i, int j) {
        if (i == j) {
            return i;
        }
        else {
        int mid = (i+j) / 2;
            int result = sum(i, mid) + sum(mid+1, j);
            return result;
    }
}
```

Which of the following lines of code from the above function divides this problem into sub-problems?

(A) int result = sum(i, mid) + sum(mid+1, j);
(B) int mid = (i+j) / 2;
(C) if (i == j) {
(D) return result;
(E) int sum(int i, int j) {

(36) What is the runtime overhead of a divide-and-conquer algorithm that recursively processes two equal halves of a problem that each have an overhead of O(n)?

(A) $O(n \log n)$ (B) $O(\log n)$ (C) $O(n^2)$ (D) $O(n^3)$ (E) $O(n)$

13. For each of the questions below, choose one of the possible answers.

(37) Amount of time required by the dynamic programming algorithm for finding the optimal parenthesization of a sequence of *n* matrices

(A) $O(n \log n)$ (B) $O(\log n)$ (C) $O(n^2)$ (D) $O(n^3)$ (E) $O(n)$

(38) Solution to the recurrence $T(n) = 4T(n/2) + 3$

(A) $O(n \log n)$ (B) $O(\log n)$ (C) $O(n^2)$ (D) $O(n^3)$ (E) $O(n)$

(39) Worst-case running time of RANDOMIZED-QUICKSORT

(A) $O(n \log n)$ (B) $O(\log n)$ (C) $O(n^2)$ (D) $O(n^3)$ (E) $O(n)$

14. Consider the following three problems. Assume that the only operations allowed on the data are

- comparing the values of two floating-point numbers and identifying the larger value;
- comparing the distance between two array entries (the absolute value of the difference between the two array entries) with the distance between two other array entries;
- swapping two entries in the array.

Further assume that each allowed operation has unit cost. What are the worst-case optimal asymptotic running times for algorithms that solve these problems?

【不可使用計算機】*作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！！

(40) Nearest Neighbors: Given an unsorted array of n floating-point numbers as input, return two of the numbers that are closest in value to each other.
(A) O(n log n) (B) O(log n) (C) O(n^2) (D) O(n^3) (E) O(n)

(41) Farthest Neighbors: Given an unsorted array of n floating-point numbers as input, return two of the numbers that are farthest in value from each other.
(A) O(n log n) (B) O(log n) (C) O(n^2) (D) O(n^3) (E) O(n)

(42) Given a floating-point number to find a closest value in a sorted array of n floating-point numbers.
(A) O(n log n) (B) O(log n) (C) O(n^2) (D) O(n^3) (E) O(n)

15. A hiker faces the Knapsack problem. There are 7 items to be packed into the knapsack, each with value v_i and weight w_i as shown in the following table.

i	1	2	3	4	5	6	7
v_i	8	6	15	4	2	5	9
w_i	2	3	5	4	4	2	6

The knapsack, which is initially empty, can hold a maximum weight of 15, so some item(s) must be left behind. The optimality criterion is to maximize the total value of the items that are placed in the knapsack. The hiker fills the knapsack one item at a time, using an algorithm that is greedy on value density, where the value density of an item is its value/weight ratio.

(43) Assume fractions of items cannot be packed. When this algorithm is used, what is the total value of the items that are packed?
(A) 28 (B) 30.5 (C) 34 (D) 37 (E) 38.5

(44) Assume the fractions of items can be packed. When this algorithm is used, what is the total value of the items that are packed?
(A) 28 (B) 30.5 (C) 34 (D) 37 (E) 38.5

(45) Total values are Optimal or Not optimal for both non-fraction / fraction cases
(A) Optimal/Optimal
(B) Not optimal/Not optimal
(C) Optimal/Not Optimal
(D) Not Optimal/optimal
(E) None of above

16. Consider an undirected graph G with N vertices and M edges, where each edge (u, v) has a positive weight w(u, v). Let P=v_0-v_1-v_2-...-v_k be a simple path from v_0 to v_k. Define the distance of P as $w(P)= \max_{i=0 \text{ to } k-1}\{ w(v_i, v_{i+1})\}$. We say that a path P is optimal from vertex s to vertex t, if w(P) is the smallest one among all paths from s to t.

(46) Which of the followings is correct?
(A) The optimal path has the so called optimal substructure.
(B) The optimal path between two vertices is unique.
(C) The problem can be solved with a greedy algorithm.
(D) The problem can be solved with dynamic programming.
(E) The Floyd-Warshall algorithm is the most efficient method for this problem.

(47) Suppose there are 5 vertices, $v_1, v_2, \ldots, v_5$ and the weight between v_i and v_j is ((i-j)*(i-j) mod 5)+1, where $i \neq j$. Which of the followings is correct?

(A) There is no solution for this input.
(B) The weight of the optimal solution between any pair of vertices is 2.
(C) There is a pair of vertices which has the optimal solution as 5.
(D) The optimal path between any pair of vertices is unique.
(E) The optimal path and the shortest path between v_2 and v_4 are the same path.

(48) Continuing the previous question, redefine the weight between v_i and v_j as ((i-j)*(i-j) mod 4)+1, where $i \neq j$. What is the weight of the optimal path between v_1 and v_4?

(A) 1 (B) 2 (C) 3 (D) 4 (E) 5.

17. For each positive integer N, find three integers a, b, c for which N= Bino(a, 3) + Bino(b, 2) + Bino(c, 1), where a> b > c >=0 and Bino(m, n) = m!/((m-n)!n!). I.e., Bino(m, n) is the well known binomial coefficient and it is 0 when m < n. So N can be represented as abc. For example, let N=3, then a=3, b=2, c=1 and N can be denoted as 321.

(49) Let N=10, then which is correct about the corresponding abc?

(A) a=4 (B) a+b=6 (C) b+c=2 (D) b=2 (E) a+b+c=7.

(50) Let N=18, then what is correct about the corresponding abc?

(A) a+b+c=12 (B) a+b=10 (C) b+c=7 (D) a=6 (E) b=4.

(51) Which of the followings is wrong about this representation of N?

(A) There can be more than one digit for a.
(B) The representation can be found by a greedy algorithm.
(C) We can use it as a number system to represent positive integers.
(D) This representation can be generalized as N= Bino(a, 4) + Bino(b, 3) + Bino(c, 2) + Bino(d, 1), where a > b > c > d >=0.
(E) The representation may not be unique for some positive N.

18. Consider the below C code fragment, where MAXV, MAXDEG and MAXINT are properly defined constant positive integers.

(52) What can the procedure *mystery()* find ?

(A) shortest path tree (B) all pairs of shortest paths (C) minimum cut (D) minimum spanning tree (E) a maximum matching.

(53) If we change the statement (*) into "*if (distance[w] > (distance[v]+weight))}*" and the statement (**) into "*distance[w]=distance[v]+ weight;*", then what can the procedure *mystery()* find ?

(A) shortest path tree (B) all pairs of shortest paths (C) minimum cut (D) minimum spanning tree (E) none of the above.

(54) What is the most accurate time complexity of the procedure *mystery()*, if there are n vertices and m edges?

(A) O(mlog n) (B) O(m) (C) O(mn^2) (D) O(n log n) (E) O(n^2 +(m+n)).

【不可使用計算機】＊作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！！

```
#include <stdio.h>
typedef struct{
        int v;    int weight; } edge;
typedef struct{
            edge edges[MAXV +1][MAXDEG];
            int degree[MAXV+1];
            int n; // n stores the vertex number.
            int m; //m stores the edge number.
} graph;
int parent[MAXV];
int mystery(graph *g, int start)
{     int i, j;
      int in[MAXV];
      int distance[MAXV];
      int v, w, weight, dist;
      for(i=1;i<= g->n;i++) {
                        in[i]=FALSE;
                        distance[i]=MAXINT;
                        parent[i]=-1;}
      distance[start]=0;
      v=start;
      while (in[v]==FALSE){
               in[v]=TRUE;
               for (i=1; i<= g->degree[v];i++){
                    w=g->edges[v][i].v;
                    weight=g->edges[v][i].weight;
(*)                     if ((distance[w] > weight)&&(in[w]==FALSE)){
(**)                                     distance[w]=weight;
                                         parent[w]=v;}}
               v=1;
               dist=MAXINT;
               for (i=2;i<=g->n;i++)
                  if ((in[i]==FALSE) && (dist > distance[i])){
                      dist = distance[i];
                      v=i; } } }
```

19. Problem I: Given a graph G = (V, E), is there a minimum-degree spanning tree T of maximum degree two, where the minimum-degree means that the maximum degree is minimized?

Problem II: Given an undirected graph and a positive integer K, is there a path of length at most K, where each edge has weight 1 and each vertex is visited exactly once?

Problem III: Given an undirected graph and a positive integer K, is there a path of length at least K, where each edge has weight 1 and each vertex is visited at most once?

(55) Which of the following statements is wrong?

(A) A problem is NP-complete, if it belongs to the class NP and all the other members in NP can be reduced to it in polynomial time.

(B) Problem I belongs to NP.

(C) Problem III belongs to NP.

(D) Problem III is NP-complete.

(E) If we change the graph in Problem III to directed graph, then it belongs to P.

(56) Which of the followings is wrong?

(A) If a spanning tree has the maximum degree as 2, then it is a Hamiltonian path.

(B) If Problem III can be solved in polynomial time, then we can find a Hamiltonian path in polynomial time.

(C) Problem I can be solved with the Prim's algorithm..

(D) A graph can have more than one spanning tree.

(E) If P=NP, then Problem II can be solved in polynomial time.

(57) Continuing the previous question, which of the following is wrong?

(A) Problem II can be solved with the Floyd-Warshall algorithm.

(B) An algorithm for Problem III can be used to find the longest path of a graph.

(C) If there is an algorithm that can find the longest path in a graph in polynomial time, then Problem III can be solved in polynomial time.

(D) Problem III can be reduced to Problem II by making each weight negative and thus can be solved with the Bellman-Ford algorithm.

(E) If Problem III can be solved in polynomial time, then P=NP.

20. This question is about the max flow problem.

(58) Which of the following statements is wrong?

(A) By the Ford-Fulkerson algorithm we can find the maximum flow.

(B) Given a flow network G=(V, E), Edmonds-Karp algorithm has time complexity $O(|V||E|^2)$.

(C) The time complexity of Ford-Fulkerson algorithm depends on the capacity.

(D) If each edge has a different capacity, then there exists a unique minimum cut.

(E) The maximum flow is equal to the capacity of a minimum cut.

(59) Which statement is wrong for a flow network G=(V, E)?

(A) If f is a maximum flow in G, then the corresponding residual network contains no augmenting path.

(B) For any cut (S, T), the capacity of the cut is not smaller than the value of the flow crossing this cut.

(C) The value of any flow f in G is bounded above by the capacity of any cut of G.

(D) If all edges of G have different capacities, then there exists a unique flow f that gives the maximum flow.

(E) The capacity of each edge of G can be any non-negative number.

(60) Let G=(V, E) be a bipartite graph, where $V = L \cup R$. Which statement is wrong about finding a maximum bipartite matching?

(A) It can be solved by constructing a corresponding flow network and finding the maximum flow.

(B) The corresponding flow network can be obtained by adding two vertices s, t and edges from s to vertices in L, and edges from vertices in R to t.

(C) The capacity of each edge in the corresponding flow network is set to 1.

(D) The maximum flow of the corresponding flow network is always integral and the flow value of each edge is integral as well.

(E) The cardinality of a maximum matching of G is equal to the maximum flow of the corresponding flow network.