編號： 204

國立成功大學 106 學年度碩士班招生考試試題

系　　所：製造資訊與系統研究所

考試科目：程式設計

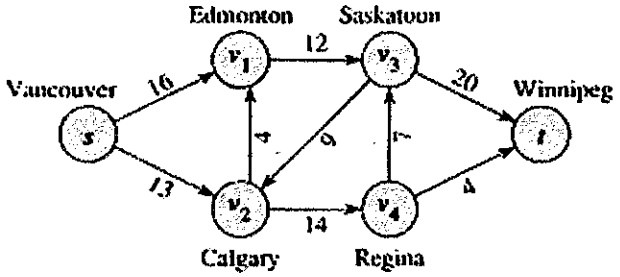
考試日期：0213，節次：2

※ 考生請注意：本試題不可使用計算機。　請於答案卷(卡)作答，於本試題紙上作答者，不予計分。

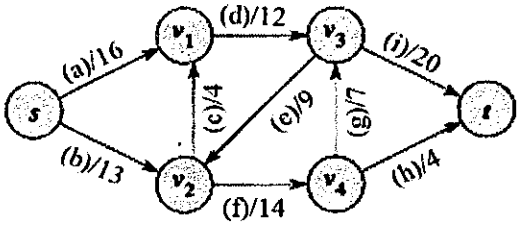
1. (15%) Describe the following data structure.
   (a) (3%) Stack
   (b) (3%) Queue
   (c) (3%) Linked List
   (d) (3%) Binary tree
   (e) (3%) Heap

2. (10%) For the set of {1, 4, 5, 10, 16, 17, 21} of keys, draw a binary search trees of height 2.

3. (20%) A flow network $G = (V, E)$ for the lucky Puck Company's trucking problem. The Vancouver factory is the source s, and the Winnipeg warehouse is the sink t. The company ships pucks through intermediate cities, but only $c(u, v)$ crates per day can go from city u to city v. Each edge is labeled with its capacity. What's the maximum value of flow in this graph? Please fill the values of [(a), (b), (c) ... (i)].



(a)



(b)

4. (20%) Illustrate the operation of SORT on the array A = {59, 31, 40, 6, 70, 40} and describe the sorting algorithm you used (ex: INSERTION-SORT, BUBBLE SORT....etc).

5. (5%) A CPU scheduling algorithm determines an order for the execution of its scheduled processes. Given n processes to be scheduled on one processor, how many possible different schedules are there? Give a formula in terms of n.

6. (10%) How to swap two variables without using a temporary variable. Hint: Use XOR ⊕.

XOR truth table

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

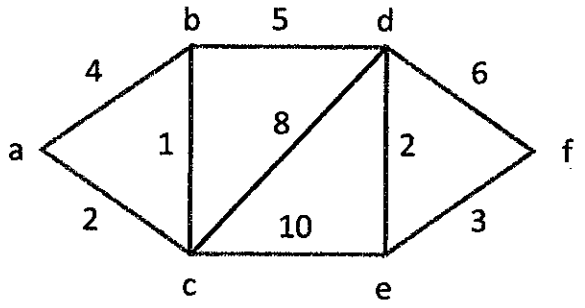
國立成功大學 106 學年度碩士班招生考試試題

系　　所：製造資訊與系統研究所

考試科目：程式設計

考試日期：0213，節次：2

7. (20%) Illustrate the operation of Dijkstra's shortest path algorithm on the graph starting from "a" to "f".




```
1   function Dijkstra(Graph, source):
2
3       create vertex set Q
4
5       for each vertex v in Graph:             // Initialization
6           dist[v] ← INFINITY                  // Unknown distance from source to v
7           prev[v] ← UNDEFINED                 // Previous node in optimal path from source
8           add v to Q                          // All nodes initially in Q (unvisited nodes)
9
10      dist[source] ← 0                        // Distance from source to source
11
12      while Q is not empty:
13          u ← vertex in Q with min dist[u]    // Node with the least distance will be selected first
14          remove u from Q
15
16          for each neighbor v of u:           // where v is still in Q.
17              alt ← dist[u] + length(u, v)
18              if alt < dist[v]:               // A shorter path to v has been found
19                  dist[v] ← alt
20                  prev[v] ← u
21
22      return dist[], prev[]
```