Université d'Ottawa · University of Ottawa

Faculté de génie
École d'ingénierie
et de technologie de l'information

Faculty of Engineering
School of Information Technology
and Engineering

CSI 3105 Final Exam

Date: December, 1997
Duration: 3 hours

Profs: Dr. Sylvia Boyd

NAME: (Please print) _____

STUDENT NUMBER: _____

Note:
1) This is a closed book exam, and no claculators are allowed.
2) The answers for this exam are to be written in the spaces provided.
3) A list of formulas which may be useful are given on the last page.
4) The total mark for this test is 100, with mark allocation as follows:

QUESTION	MARKS	MARKS OBTAINED
1	15	
2	10	
3	10	
4	10	
5	15	
6	15	
7	15	
8	10	
TOTAL	100	

GOOD LUCK!

C.P. 450, Succ. A
Ottawa (Ontario) K1N 6N5 Canada

P.O. Box 450, Stn. A
Ottawa, Ontario K1N 6N5 Canada

(613) 562-5826 • Téléc./Fax (613) 562-5187

1. [15 marks total]

Recall the Sequential Search Algorithm from class, which given an array S of n keys and a key x, determines if x is in S.

```
procedure seqsearch (n: integer;
                     S: array[1..n] of keytype;
                     x: keytype;
          var location: index);
begin
  location:= 1;
  while location≤n and S[location] ≠ x do
    location:= location+1
  end;
  if location>n then
    location:=0
  end
end;
```

a) [5 marks] Do a worst case analysis of this algorithm. For this analysis, use comparisons of x with keys as the basic operation being counted.

b) [10 marks] Do an average case analysis of this algorithm, under the ssumptions that we know x is in S, the keys in S are distinct, and that all positions in S are equally likely for x. Be sure to clearly explain your steps (and any formulas you use) in your analysis.

2. [10 marks total]
Recall Floyd's dynamic programming algorithm which computes the shortenst paths from each node in a weighted undirected graph to each of the other nodes. The weights are nonnegative numbers, and we are given the adjacency matrix W for the graph. Let $D^{(k)}[i,j]$ = the length of a shortest path from v_i to v_j using only nodes in the set $\{v_1, v_2, \ldots, v_k\}$ as internal nodes.

(a) [5 marks]
Write the recursive formula for $D^{(k)}[i,j]$ used by Floyd's algorithm.

(b) [5 marks]
The matrix $D^{(2)}$ is given below for an execution of Floyd's algorithm. Using this matrix, find $D^{(3)}$.

$$\begin{bmatrix} 0 & 1 & 3 & 8 & \infty \\ 1 & 0 & 2 & 7 & \infty \\ 3 & 2 & 0 & 1 & 5 \\ 8 & 7 & 1 & 0 & 2 \\ \infty & \infty & 5 & 2 & 0 \end{bmatrix}$$

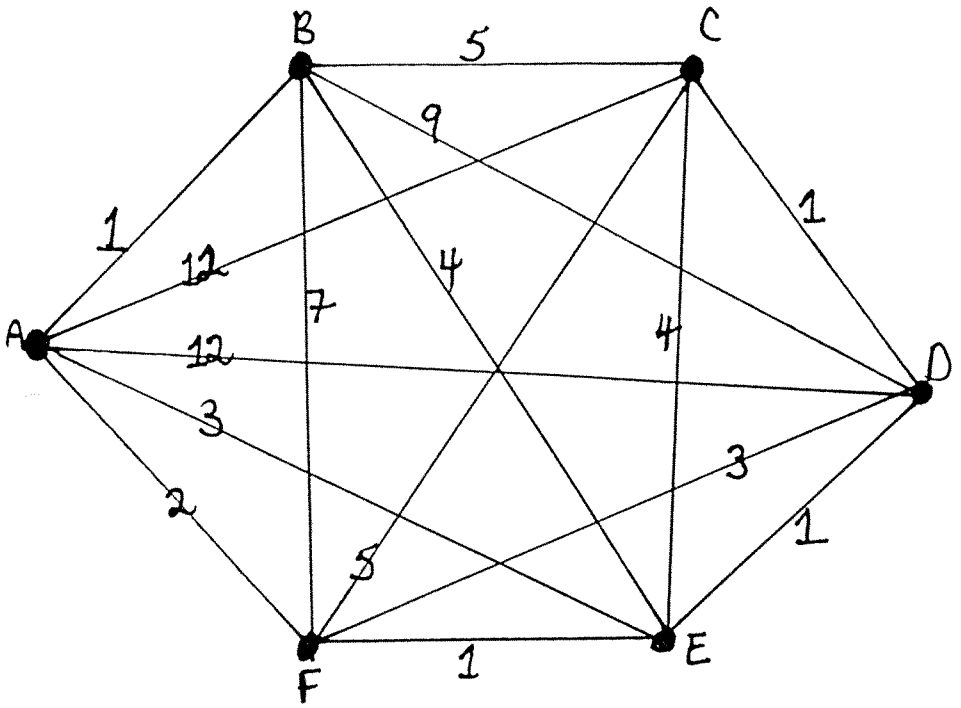
3. [10 marks]

Given all the book check out cards used in the library for the past year (each of these cards has the name of who checked out that book), describe an algorithm which would determine how many distinct people checked out at least one book over the past year. State the complexity of your algorithm, and explain how you get it.
Note: Your algorithm does not have to be in great detail, and you will be marked on its efficiency.

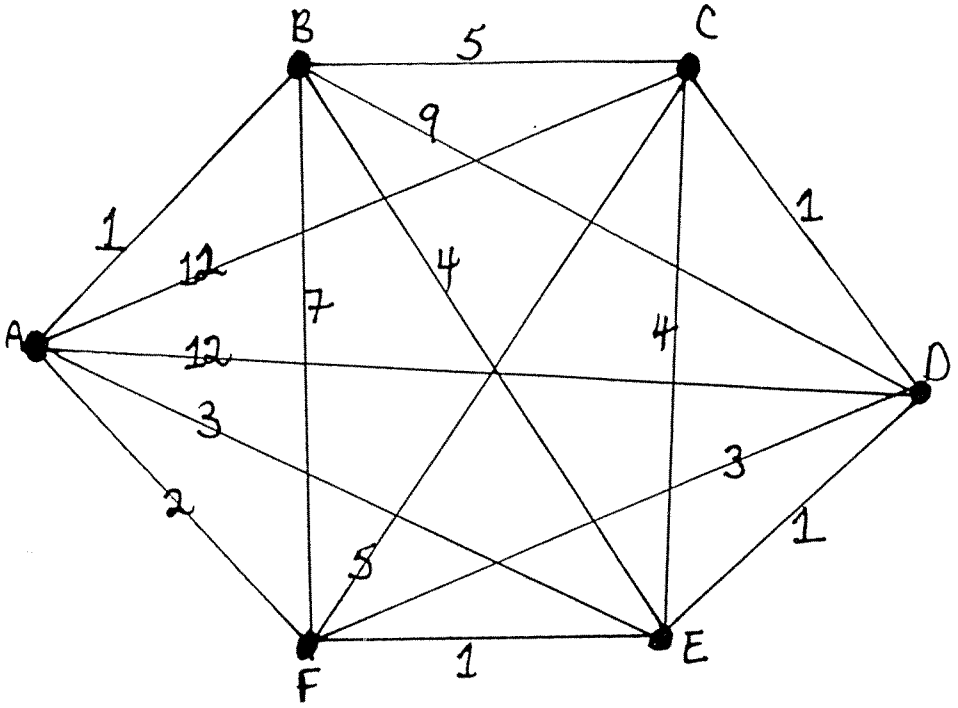
4. [10 marks]
Prove that any algorithm which uses comparisons to sort n keys must do at least $\lceil \lg n! \rceil$ key comparisons. For your proof you may use the fact that for any binary tree of depth d with k leaves, we must have $k <= 2^d$.

5. [15 marks]

a) For the graph G shown below, find the minimum cost spanning tree T using Prim's algorithm. Start the algorithm at node A. You do not have to show the details of the algorithm, but draw the tree you find directly on G, and number the edges in the tree in the order that they are found by the algorithm. Also state the cost of this tree.



b) For the same graph G, use the greedy Nearest Neighbour heuristic discussed in class to find a TSP tour for G. Start the heuristic at node A. State the cost of the tour found, and show it on G.



c) State the relationship the answers from a) and b) have with the cost of an optimal TSP tour.

6. [15 marks]

a) Given a problem Q that you wish to show is NP-complete, state the two things you must prove about Q in order to show this.

b) Define what is meant when we say an algorithm is a pseudopolynomial algorithm. Name one pseudopolynomial algorithm from class.

c) Give three examples of problems from class that are known to be NP-complete. Give two examples of problems from clas that are know to be in the class P.

7. [15 marks]

Suppose we wish to check if a given node u in a directed graph D is a source node for D. (Recall that we defined a source node to be one which has an edge directed out to every other node, and no edges are directed into u.) For each of the following data structures, briefly describe the necessary steps for checking u and the complexity. For your analysis use n as the number of nodes and m as the number of edges.

a) Adjacency matrix

b) Adjacency list (the out-neighbours are listed)

b) Adjacency list (the out-neighbours are listed)

8. [10 marks]
For each of the following two algorithms, state the complexity of the algorithm. You do NOT need to show your work for this quesiton, just state the complexity (I will only be marking your final answer).

a)

```
MaxSoFar := 0.0
for L := 1 to N do
    for U := L to N do
        Sum := 0.0
        for I := L to U do
            Sum := Sum + X[I]
        /* Sum now contains the
             sum of X[L..U] */
        MaxSoFar := max(MaxSoFar, Sum)
```

Note: State the complexity in terms of N

b)

```
recursive function MaxSum(L, U)
    if L > U then      /* Zero-element vector */
        return 0.0
    if L = U then      /* One-element vector */
        return max(0.0, X[L])

    M := ⌊(L + U)/2⌋    /* A is X[L..M], B is X[M + 1..U] */
    /* Find max crossing to left */
        Sum := 0.0; MaxToLeft := 0.0
        for I := M downto L do
            Sum := Sum + X[I]
            MaxToLeft := max(MaxToLeft, Sum)
    /* Find max crossing to right */
        Sum := 0.0; MaxToRight := 0.0
        for I := M + 1 to U do
            Sum := Sum + X[I]
            MaxToRight := max(MaxToRight, Sum)
    MaxCrossing := MaxToLeft + MaxToRight

    MaxInA := MaxSum(L, M)
    MaxInB := MaxSum(M + 1, U)
    return max(MaxCrossing, MaxInA, MaxInB)
```

Note: Given an array X=[1..N], state your complexity in terms of N for a call MaxSum(1,N)

FORMULAS WHICH MAY BE USEFUL:

$$\sum_{i=1}^{n} i^2 = \frac{2n^3+3n^2+n}{6} \qquad \sum_{i=0}^{k} 2^i = 2^{k+1}-1 \qquad \sum_{i=0}^{k} \frac{1}{2^i} = 2-\frac{1}{2^k}$$

$$\sum_{i=1}^{k} i2^i = (k-1)2^{k+1}+2 \qquad \sum_{i=2}^{n} \frac{1}{i} = \ln n \qquad \sum_{i=1}^{n} \lg i \geq n\lg n - 1.5n$$