CSI 3105 Assignment 4 Solutions
Fall, 2014

1. If the graph is not connected, then that means that at some stage of the algorithm, **min** will remain at value ∞ after the for-loop that checks each vertex for being nearest to Y. So we can use the pseudocode for Prim's algorithm from the textbook pretty much the way it is, but put in a check for this situation:

{first part of pseudocode, the same except that the procedure would also output a Boolean " connected", then…changes in RED}

```
.
.
.
connected := true;
repeat n-1 times or until connected:= false
     min:= ∞  ;
     for i:= 2 to n do
        if 0 <= distance[i] < min then
            min:= distance[i];
            near:= i;
        end;
    if (min= = ∞  ) then connected := false;
end;
if connected == true then
  {the rest the same as the pseudocode in the text}
```

2. If the women do the asking:

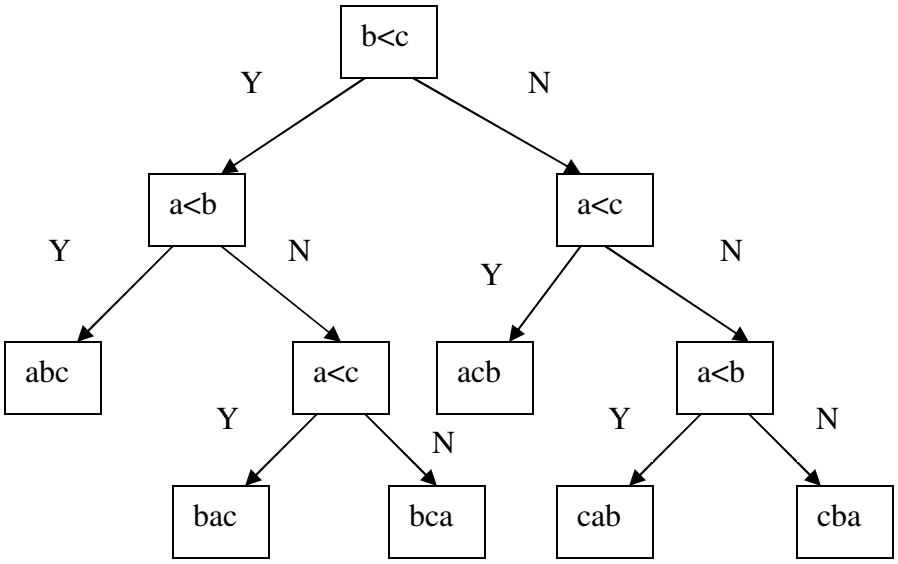
   A-1, B-4, C-5, D-3, E-2

3.

Stage	Job Considered	Time slot allotted	Schedule
1	3	3	_ _ 3 _
2	7	1	7 _ 3 _
3	6	None	7 _ 3 _
4	1	4	7 _ 3 1
5	4	None	7 _ 3 1
6	2	2	7 2 3 1
7	8	None	7 2 3 1
8	5	None	7 2 3 1

Final Schedule: 7 2 3 1 Profit: 170

4. a)



b) W(3) = depth of tree = 3.

c) The fewest number of comparisons = the length of a shortest root to leaf path =2.

5. a)
Here is one solution (I mention two other possible solutions at the end of this question): Suppose the variables are x_1, x_2, x_3,..., x_n in the 3-SAT expression and there are k clauses. For each clause make a duplicate clause, and add a literal x_{n+1} to the original copy of the clause and the negation of x_{n+1} to the copy of the clause. This gives an instance of 4-SAT with 2k clauses.

i) The work involved in this algorithm TRAN is essentially to copy the input twice with a new literal in each clause. This can clearly be done in polynomial time in the size of the input.

ii) Let y be the instance for 4-SAT created by TRAN from x, the instance of 3-SAT. If the answer for y is yes, then that means there is a T/F assignment A to the variables that make each the whole expression true, which means this assignment gives at least one true literal in each clause. We claim that this same assignment for the variables (excluding the added variable x_{n+1}) gives a yes answer for the instance x for 3-SAT.

Proof: Consider any clause S in the instance x of 3-SAT, and the corresponding clauses C and its duplicate C' in y. We have that if S consists of the 3 literals a, b and c, i.e.
$S = (a \vee b \vee c)$, then

–

C = $(a \vee b \vee c \vee x_{n+1})$ and C' = $(a \vee b \vee c \vee x_{n+1})$.

Case 1:
Suppose variable x_{n+1} has value true in the assignment A. Then at least one of a, b and c must be true in order for C' to be true. Thus we have at least one true literal in S.
Case 2:
Suppose variable x_{n+1} has value false in assignment A. Then at least one of a, b and c must be true in order for clause C to be true. Thus we have at least one true literal in S.

In both Case 1 and Case 2, we have that the same assignment as A (excluding the added variable x_{n+1}) will end up giving at least one true literal per clause in x, and thus be a an assignment for the 3-SAT instance x that makes the whole expression true. Hence the answer for x is also yes.

iii) Let x be an instance for 3-SAT for which the answer is yes. Let B be a T/F assignment for the variables of x that make the whole expression x true. This means for every clause, there is at least one true literal. The clearly means that the same assignment (with the new variable x_{n+1} set to true or false, it doesn't matter) gives at least one true literal for every clause in y = TRAN(x), and so the answer is yes for y as well.

b) [2 marks]

$$(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_5)$$

Two alternate solutions for part a):

ALTERNATE SOLUTION 1:
For each clause take one of the literals in that clause and repeat it to get an instance of 4-SAT.

ALTERNATE SOLUTION 2:

Add a new literal $\bar{x}_{new}$ to all clauses, and then add one extra clause which is
$(x_{new} \vee x_{new} \vee x_{new} \vee x_{new})$