C 13.2 Answer:

|  |
| --- |
| Since L is in P which is polynomial time, and it's reducible by language M. Assume that we have algorithm A(x) that will return yes if the input is equal to 5.  The piece of algorithm will be :  if A(x) = yes then  return 5  else return 10 |

Q: Show that the MST decision problem is polynomial-time reducible to the Subset Sum problem.

Answer:

|  |
| --- |
| Algorithm MST2SS(G,k)  Input:  Output:  T<-MST(G)  sum <- 0  for each e of T.edges() do  sum <- sum + weight(e)  S<-new empty sequence  S.insertLast(2)  if sum <= k then  return (S,2)  return (S,1) |

Q: Show the shortest path decision problem is polynomial-time reducible to the MST decision problem. **Hint**: convert the shortest path problem to a decision problem, then reduce to MST problem.

Answer:

|  |
| --- |
| Algorithm SP2MST(G,k)  G' <- new Graph  u<-G'.insertVertex("u")  v<-G'.insertVertex("v")  e<-G'.insertEdge(u,v,2)  p<-shortestPath(G')  sum<- 0  for each e of P do  sum <- sum + weight(e)  if sum <= k then  return (G',2)  else  return (G',1) |

R 13.1 Answer:

|  |
| --- |
| He has shown that L is reducible to NP-complete. That means L can be anywhere in entire NP region even in P. He has not proven anything abt the location of L. So, solving L in polynomial time does not prove anything because L may be already in P and all problems in P are already solvable in polynomial time.  So, to prove P=Np, he has to show one of the followings  a) L is an NP complete problem or M is also reducible to L  b) L is an NP hard problem.  If he was able to prove any of the above, all NP problems could be reduced to L which can be solved in polynomial time. |

R 13.3 Answer:

|  |
| --- |
| F = a OR b AND C NOT a   1. Randomly assign T/F to the variable in the formula 2. Algorithm check\_SAT(f)   Assign input variable into the gate and will give yes/no. |

R 13.13 Answer:

|  |
| --- |
| SS1(S,max,min),SS2(S,T)  (S,T)  return (S,T,T)  Making 100: 23, 47, 22,8 |