# Question 1

(a) Canonical LP formulation
Problem Definition:
We have been given n terminals (Red and Blue combined) for a robot's controller board. We need to be able to connect each red terminal with every blue terminal using a wire so that by just connecting a subset of red-blue pairs, we can control various robotic functions. Our main aim is to connect each red with blue with a minimum total wire length.

We need to formulate this in LP in Canonical form.
Canonical form:
Max C.x
s.t. $Ax <= b$
$x >= 0$

1) Variables
Since the wire to connect can originate from both Red and Blue blocks, Consider:

(a) $L(Ri)-$ Length of ith wire from Red blocks i = 1...k
(b) $L(Bj)-$ Length of jth wire from Blue blocks j = 1...l

(k + l = n)

2) Constraints

(a) $L(Ri) >= 0$, //Length can be 0 as well
(b) $L(Bj) >= 0$, //Length can be 0 as well
(c) $L(Ri) + L(Bj) >= distance(Ri, Bj)$

Since In canonical form the constraints need to be less than or equal, to maximize a function. Constraint (c) can be re-written as:
$-L(Ri) - L(Bj) <= -distance(Ri, Bj)$

3) Objective function

Minimize: $\sum_{i=1}^{k} L(Ri) + \sum_{j=1}^{l} L(Bj)$
Since In canonical form we maximize the objective function, we can re-write our Objective function as:

Maximize: $-(\sum_{i=1}^{k} L(Ri) + \sum_{j=1}^{l} L(Bj))$

(b) Dual from the LP of the above part
Duality:
$-\sum_{i=1}^{k} L(Ri) - \sum_{j=1}^{l} L(Bj) <= y1 * (-L(Ri), -L(Bj)) <= -distance(Ri, Bj) * y1$

1) Objective function:
Min $-distance(Ri, Bj) * y1$

2) Constraints

(a) $y1 >= 0$
(b) $-y1 >= -1$
(c) $-y1 >= -1$

# Question 2

(a) Description

Given: We have an oracle function, that takes an LP as an input and determines whether there exists a feasible solution or not.

We have a linear program whose optimum solution (Min ($C^T.x$) lies within the range [M, -M].

We need to prove that, Given the oracle, we can get the optimum value of the Linear program within a range of [$\epsilon$ to $-\epsilon$] (an error rate of $+-\epsilon$), in $O(Log(M/\epsilon))$ oracle calls.

Consider our original LP, Now According to the hint provided, we can have a new sub LP within a range [i to j] from our original range with our optimum value $<= z$ by adding constraints like ($C^T.x <= i$) and ($C^T.x >= j$). If we pass this new LP into our oracle and the solution is feasible, then we can say that there is an optimum value less than z in the range till i else the optimum value lies above the range from i. We will follow the concept of binary search to localize the range and reduce it by half after every call to oracle until we find the final range of solution within [$-\epsilon$ to $\epsilon$]. There can be three cases, The optimum range lies in the lower half, or it lies in the upper half, or it lies in between the halves (here we take the lower half, as this is a minimization problem).

We will keep on dividing the original range into two halves until we reach $2^k$, Where we have $M/2^k$ parts. At our target we will get range $< 2 * \epsilon$. On calculation, This will result in a time complexity of $O(log(M/2 * \epsilon)$

Consider the following algorithm.

**Algorithm 1:** Get the optimum value within the range [$-\epsilon$ to $\epsilon$]

```
/* Here's my algorithm.
Assume modifyLP to give a new LP with provided range
callOracle gives LP feasibility */
```

```
GetOptVal(LP, i, j){
    if range between | -epsilon to epsilon | then
        return range
    end

    mid = (i + j)/2

    newLp = modifyLP(LP, i, mid)
    result = callOracle(newLp)

    if(result == feasible) then
        GetOptVal(newLp, i, mid)
    else
        GetOptVal(newLp, mid+1, j)
    end
}
```