**Problem Set 1**

A)

a)

**Pseudo code :-**

PARTITION (r,L)

cubesContainingR = { }

for (l1 = 0 to L-1)

for(l2 =0 to L-1)

for(l3 = 0 to L-1)

l = (l1,l2,l3)

if r belongs to Cl

add l to cubesContainingR

end for

end for

end for

return cubesContainingR

end PARTITION

PARTITION-ALL(R,L)

INPUT – R: set of K points in C

L: positive integer

Rl[L][L][L][K] <- {0}

for (r in R)

cubesContaining\_r <- PARTITION(r,L)

for (l in cubesContaining\_r)

Rl[l.x][l.y][l.z][indexof(r)] <- 1

end for

end for

for (l in Cl)

write l to file

for(r in R)

if (Rl[l.x][l.y][l.z][indexof(r)] == 1)

write r to file

end for

end for

end PARTITION-ALL

**Algorithm Analysis :-** For each of l1,l2 and l3 which ranges from **0 to L-1,** check whether r belongs to Cl and if so then save it. So there exists a total of **L3** combinations for each l1,l2,l3 .so there exists **L3** opeatations and hence the complexity of the algorithm is **O(L3).**

b)

**Pseudo Code:-**

BUILD-MATRIX (L, l, P, Q)

INPUT – L: positive integer

l: (l1,l2,l3)

P: set of N points in R3

Q: set of M points in R3

Check if points in P lie on the boundary of Cl

Check if points in Q lie on the boundary of Sl

for (q in Q)

for(p in P)

Aij = sin(|q – p|)/(|q – p|)

end for

end for

for (i = 1 to N)

for (j = 1 to N)

ATA[i][j] <- 0

for (k in M)

A­TA[i][j] <- ATA[i][j] + AT[i][k] \* A[k][j]

end for

end for

end for

return ATA

end BUILD-MATRIX

**Algorithm Analysis :-**

A which is an M X N matrix and AT will be an N X M matrix. So, ATA is an N X N matrix and to calculate each element of ATA, it requires a total of M multiplications and M-1 additions. So, the total no. of operations required for the calculation of entire matrix ATA are (2M-1) X N2 . So, the complexity of the algorithm is **O(N2M).**