Task 2.2.a

In [1]:
```
# For the given problem, dependent variable is "Price of the flat (z)" and
# i.e "Size of the flat in square ft(x)" & "number of bedrooms in the flat(
# let z = ax+by+c is a hyperplane which is nearest to the points given in t
# Now, using least square method for linear regression, x=(a,b,c)^T = (A^TA
# This concept is described in part 2.1
# The code is written below
```

In [2]:
```
#Importing libraries
import numpy as np
```

In [3]:
```
dataset = [[1600,3,8.2],
           [1260,2,6.6],
           [1800,4,10.3],
           [600,1,1.7],
           [850,2,3.6],
           [920,2,4.4],
           [1090,2,5.4],
           [890,2,4.8],
           [1340,3,10.5],
           [1650,2,7.4]]
# For each sublist of list dataset, 1st entry shows x value, 2nd entry show
```

In [4]:
```
# writing it in z= ax+by+c form and then convert it into matrix form
# I am writing it in matrix equation form directly

A=[]
for i in range(10):
    temp=[]
    temp.append(dataset[i][0])
    temp.append(dataset[i][1])
    temp.append(1) # for coefficient of c i.e. 1
    A.append(temp)

B=[]
for i in range(10):
    B.append(dataset[i][-1])
```

In [5]:
```
# Printing matrix A and vector B
print(A)
print(B)
```

```
[[1600, 3, 1], [1260, 2, 1], [1800, 4, 1], [600, 1, 1], [850, 2, 1], [920, 2,
1], [1090, 2, 1], [890, 2, 1], [1340, 3, 1], [1650, 2, 1]]
[8.2, 6.6, 10.3, 1.7, 3.6, 4.4, 5.4, 4.8, 10.5, 7.4]
```

In [6]:
```
#writing function for transpose of a matrix
def transpose(A,m,n):
    trans=[]
    for j in range(n):
        temp=[]
        for i in range(m):
            temp.append(A[i][j])
        trans.append(temp)
    return trans
```

In [7]:
```
trans_A= transpose(A,10,3)
print(transpose(A,10,3)) # printing transpose of matrix A
```

```
[[1600, 1260, 1800, 600, 850, 920, 1090, 890, 1340, 1650], [3, 2, 4, 1, 2, 2,
2, 2, 3, 2], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

In [8]:
```
mul = np.dot(trans_A, A) # multiplying A^T and A
inv = np.linalg.inv(mul) # inverse of A^T*A
```

```
prod= np.dot(inv,trans_A) # multiplying (A^T*A)^-1 and A^T
res = np.dot(prod,B) # finding (A^T*A)^-1 * A^T* B
```

In [9]:

```
# So, our result matrix is
print(res) # It shows the values of a,b,c
```

```
[ 0.00362953  1.67818099 -1.9252501 ]
```

Conclusion : Best fit hyperplane for the given data is z = 0.0036*x + 1.6781*y - 1.9252

###----------

To estimate the upper and lower limit of bank loan based on the requirement, I will use the above

best fit hyperplane

Here, according to requirement 950<=x<=1050 and y=2,3

For, x= 950,y=2 ----> z = 4.851

For, x= 950,y=3 ----> z = 6.5291

For, x= 1050,y=2 ----> z = 5.211

For, x= 1050,y=3 ----> z = 6.8891

It means, for 950 square ft. flat with 2 bedrooms, flat cost will be minimum ie 4.851 millions and

for 1050 square ft. flat with 3 bedrooms, flat cost will be maximum ie 6.8891 millions

Note: These are just estimates