Sinhgad College of Engineering Department of Computer Engineering

Name of Student: **PRASHANT KUMAR**

Roll No: **405B049**

PRN Number: **71813716H**

SEAT No: **B150234347**

Class: **BE** Div.: **2** Batch: **B**

Name of Laboratory: **Laboratory Practice III**

List of Assignments

S.NO.	Title of Assignment	Remark	Signature
1	Linear Regression: Find equation of best fit line for given data		
2	Decision Tree Classifier implementation		
3	K-NN classification algorithm implementation		
4	K-Means clustering algorithm implementation		
5	S-DES algorithm implementation		
6	S-AES algorithm implementation		
7	Diffie-Hellman Key Exchange algorithm implementation		
8	RSA algorithm implementation		

ASSIGNMENT No : 1

TITLE : Assignment based on Linear Regression

PROBLEM DEFINITION : The following table shows the results of a recently conducted study on the correlation of the number of hours spent driving with the risk of developing acute backache. Find the equation of the best fit line for this data.

PREREQUISITE : Basic of Python, Data Mining Algorithm

SOFTWARE REQ : Anaconda with Python 3.7

HARDWARE REQ : P4, 2GB RAM, 500GB HDD

LEARNING OBJECTIVES : Learn Linear Regression for given different Dataset

OUTCOMES : After Completion of this assignment Students are able to understand the How to find the correlation between Two Variable, How to calculate Accuracy of Linear Model and how to plot graph using matplotlib.

Teacher Signature

PAGE :
DATE : / /

Linear Regression

1. What are the four assumptions of linear regression?

Ans Four assumptions of linear regression.

(a) There is a linear relationship between the independent variable and dependent variable.

(b) The residuals are independent. It is mostly relevant when working the with time series data.

(c) The residuals have constent variance at every level of 'x'.

(d) The residuals of the model are normally distributed.

2. What is meant by dependent and independent variables?

Ans Independent variable is a variable that stands alone and isn't changed by the other variables you are trying to measure. Eg:- 'Age' of person.

Dependent variable is a variable that is dependent on other factors for the output.
Eg:- 'Test score' of an exam.

PAGE:
DATE: / /

What is difference between linear and multiple linear regressions?

Simple linear regression has only one and one dependent variable; whereas multiple regression has one dependent variable and two or more independent variable. Eg:- When we predict the same based on carpet area, ~~interior~~ alone, its linear regression and when we predict the same based on carpet area, interior design, age of building, it is multiple linear regression.

4. What is difference between regression model and estimated regression equation?

Ans Regression model is a form of predictive modelling technique which investigates the relationship between a dependent and independent variable. This technique is used for forecasting, time series modelling and finding the casual effect relationship between the variable. Regression model equation is a mathematical equation representing the relationship between 2 or 3 variables

It is given by -

$Y = a + bx + e$; where Y is dependent Variable,
x is independent variable,
b is slope of regression line,
e is error

5. What are different types of regression? What are their significance?

Ans Types of regression:

(a) Linear regression - It is used when dependent variable is linearly related to independent variables.

(b) Logistic Regression - It used Sigmoid curve to show relationship between target and dependent variable.

(c) Ridge regression - It is used when there is high correlation between independent variable.

(d) Polynomial regression - It is used to model a non-linear dataset using a linear model. It is similar to multiple linear regression but uses non linear curve.

PAGE :
DATE :

Conclusion:

Thus we learn that to how to find the trend of data using X as Independent Variable and Y is and Dependent Variable by using Linear Regression.

CODE :

```
import numpy as np
import pandas as py
import matplotlib.pyplot as plt

def estimate_coef(x, y):
    n = np.size(x) # number of observations/points
    m_x, m_y = np.mean(x), np.mean(y) # mean of x and y
vector
    # calculating cross-deviation and deviation about x
    SS_xy = np.sum(y*x) - n*m_y*m_x
    SS_xx = np.sum(x*x) - n*m_x*m_x
    # calculating regression coefficients
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x
    return(b_0, b_1)

def plot_regression_line(x, y, b):
    # plotting the actual points as scatter plot
    plt.scatter(x, y, color = "m",
            marker = "o", s = 30)
   # predicted response vector
    y_pred = b[0] + b[1]*x
    # plotting the regression line
    plt.plot(x, y_pred, color = "g")
    # putting labels
    plt.xlabel('x')
    plt.ylabel('y')
    plt.show() # function to show plot
def main():

data = py.read_csv("linear.csv") #reading table data from
csv file
   x = data['x']
   y = data['y']
   b = estimate_coef(x,y) # estimating coefficients
   print("Estimated coefficients:\nb_0 = {} \
    \nb_1 = {}".format(b[0], b[1]))
   plot_regression_line(x,y,b) # plotting regression line
if __name__ == "__main__":
    main()
```

Input for the code –

```
#linear.csv

x,y
10,95
9,80
2,10
15,50
```

10,45
16,98
11,38
16,93

OUTPUT :

Estimated coefficients:
b_0 = 12.584627964
b_1 = 4.58789860998

ASSIGNMENT No. 2

TITLE : Assignment based on Decision Tree classifier

PROBLEM DEFINITION : A dataset collected in a cosmetics shop showing details of customers and whether or not they responded to a special offer to buy a new lip-stick is shown in table below. Use this dataset to build a decision tree, with Buys as the target variable, to help in buying lip sticks in the future. Find the root node of decision tree. According to the decision tree you have made from previous training data set, what is the decision for the test data.

PREREQUISITE : Basic of Python, Data Mining Algorithm, Concept of Decision Tree Classifier

SOFTWARE REQUIREMENTS : Anaconda with Python 3.7

HARDWARE REQUIREMENT : P4, 2 GB RAM, 500GB HDD

Teacher Signature_____

Learning Objectives : Learn how to apply Decision Tree Classifier to find the root node of decision tree. According to the decision tree you have made from previous training data set.

OUTCOMES : After completion of this assignment students are able to Implement code for Create Decision Tree for given dataset and find the root node for same based on the given condition.

PAGE :
DATE : / /

What is Decision Tree Classifier and how that algorithm works?

Decision Tree algorithm belongs to family of Supervised learning algorithm. It can be used to solve regression and classification problem.

The goal is to create a learning simple decision rules.

It builds classification a regression models in the form of tree structure. It breaks down a data into smaller subsets while at the same time an associated decision tree is developed. The final result is a tree with decision nodes and output leaf.

2. Explain Following

(i) Information Gain :- It is based on decrease in entropy after a dataset is splitted on an attribute. Constructing a decision tree is all about finding feature that has highest information gain and hence, affects the result the most.

(ii) Gini Index :- It measures the degree of probability

PAGE :
DATE : / /

of a particular variable being wrongly classified when it is randomly chosen. The values varies between 0 and 1; 0 denotes all elements belong to a class and 1 denotes all elements are distributed across different classes.

(iii) Entropy :- It is degree of uncertainity and impurity. The aim of decision tree is to reduce the level of entropy from root node to leaf node.

3. Explain pruning of Decision Trees

Ans Pruning is the proces of reducing the size of the tree by turning some branch nodes into leaf nodes and removing leaf nodes under the original branch. It is shortening of branches of the tree. It is useful as it avoid over fitting of data during training phase.

4. Explain Gain Ratio.

Ans Information Gain Ratio is the ratio of information gain to entropy. It is used to normalize the

PAGE :
DATE : / /

of a particular variable being wrongly classified when it is randomly chosen. The values varies between 0 and 1; 0 denotes all elements belong to a class and 1 denotes all elements are distributed across different classes.

(iii) Entropy :- It is degree of uncertainity and impurity. The aim of decision tree is to reduce the level of entropy from root node to leaf node.

3. Explain pruning of Decision Trees

Ans Pruning is the process of reducing the size of the tree by turning some branch nodes into leaf nodes and removing leaf nodes under the original branch. It is shortening of branches of the tree. It is useful as it avoid over fitting of data during training phase.

4. Explain Gain Ratio.

Ans Information Gain Ratio is the ratio of information gain to entropy. It is used to normalize the

PAGE :
DATE : / /

information gain of an attribute against how much entropy it has.

Gain ratio = Information Gain / Entropy

5. State advantages and disadvantages of Decision Tree

Ans Advantages →

(a) Compared with other algorithm, decision tree require less effort for data pre processing.

(b) It does not require normalisation of data.

(c) It does not require Scaling of data.

(d) Missing values do not affect the problem of building a decision tree to some extent.

Disadvantages →

(a) Small change in data can cause large structure of decision tree.

(b) It require higher time to train the model.

(c) It is inadequate for applying regression and predicting continuous values.

PAGE :
DATE :

CONCLUSION :

In this way we learn that to how to create Decision Tree based on given decision, Find the Root node of the tree using Decision tree Classifier.

CODE :

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.externals.six import StringIO
import pydotplus as pdd
from IPython.display import Image
from sklearn.tree import export_graphviz

data=pd.read_csv("sales.csv")
data
data.describe()

print(data['Buys'].value_counts())
le=LabelEncoder();
#data=data.apply(le.fit_transform)
x=data.iloc[:,:-1] #-1 means don't take last column
x=x.apply(le.fit_transform)
#Store labels in Y
y=data.iloc[:,-1]
classifier=DecisionTreeClassifier(criterion='entropy')
classifier.fit(x,y)
#Predict value for the given Expression
#[Age < 21, Income = Low,Gender = Female, Marital Status =
Married]
test_x=np.array([1,1,0,0])
pred_y=classifier.predict([test_x])
print("Predicted class for input [Age < 21, Income =
Low,Gender = Female, Marital Status = Married]\n", test_x," is
",pred_y[0])
#method to generate graph
dot_dat=export_graphviz(classifier,out_file=None,feature_names
=x.columns,class_names=["No","Yes"])
graph=pdd.graph_from_dot_data(dot_dat)
graph.write_png("tree.png")
Image(graph.create_jpg())
```

Input for the code –

```
#sales.csv

ID,Age,Income,Gender,MaritialStatus,Buys
1,<21,High,Male,Single,No
2,<21,High,Male,Married,No
3,21-35,High,Male,Single,Yes
4,>35,Medium,Male,Single,Yes
5,>35,Low,Female,Single,Yes
6,>35,Low,Female,Married,No
7,21-35,Low,Female,Married,Yes
8,<21,Medium,Male,Single,No
```

9,<21,Low,Female,Married,Yes
10,>35,Medium,Female,Single,Yes
11,<21,Medium,Female,Married,Yes
12,21-35,Medium,Male,Married,Yes
13,21-35,High,Female,Single,Yes
14,>35,Medium,Male,Married,No

OUTPUT :

Yes 9
No 5

Name: Buys, dtype: int64

Predicted class for input [Age < 21, Income = Low,Gender = Female, Marital Status = Married]

[1100] is Yes

Ruff & Tuff
Date
Page

ASSIGNMENT No : 3

TITLE : Assignment based on KNN Classification

PROBLEM DEFINITION : In the following diagram let blue circles indicate positive examples and orange squares indicate negative examples. We want to use KNN algorithm for classifying the points. If K = 3, find the class of the point (6,6). Extend the same example for Distance-Weighted KNN and Locally weighted Averaging.

PRERE QUISITE : Basic of Python, Data Mining Algorithm, Concept of KNN Classification

SOFTWARE REQUIREMENTS : Anaconda with Python 3.7

HARDWARE REQUIREMENTS : P4, 2GB RAM, 500 GB HDD

LEARNING OBJECTIVES : Learn How to Apply KNN Classification for Classify Positive and Negative Points in given example.

Teacher Signature

OUTCOMES: After completion of this assignment Students are able to Implement code for KNN Classification for Classifying Positive and Negative Points.

Teacher Signature

PAGE :
DATE : / /

KNN

1. Explain KNN Algorithm.

Ans KNN is supervised machine learning. It uses the similarity between the new data point and available datapoint. It puts the new point in the category that is most similar to available categories.

It is mainly used for classification problems. It is lazy learning algorithm. It does not learn from training data immediately instead stores the dataset and at the time of classification, it performs necessary operation to get the output category of dataset.

2. What is nearest neighbour classification?

Ans Working -

(a) Select the 'k' no. of neighbours

(b) Calculate distance of unknown point from each of the available points.

(c) Take 'k' neighbours whose distance is minimum from unknown point.

(d) Get mode for the type of category of these neighbours.

PAGE :
DATE : / /

(e) The resultant category is assigned to the unknown data point.

3. What is Distance function in KNN and how distance is calculated?

Ans Distance metric used distance function to provide a relationship metric between each elements in the dataset. Euclidean distance, Minkowski distance, Manhattan distance are commonly used distance function.

Euclidean distance - $d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$

Manhattan distance - $d(x,y) = \sum_{i=1}^{n} |x_i - y_i|$

Minkowski distance - $d(x,y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$

PAGE :
DATE : / /

4. Which libraries / functions / packages in python are used in KNN implementation?

Ans Sc-kit learn "from Sklearn.neighbors import KNeighborClassifier" is used.

5. What are different types of regression? What are their significance?

Ans The value of 'K' is non-parameter and usually, it is value is K = Sqrt(N)/2 where N is no. of samples in your training dataset. A small noise of k means the noise will have a higher influence on the result and large value of k is computationally expensive. Sometimes, an odd number is chosen for smaller projects.

CONCLUSION: In this way we learn KNN classification to predict the general and Distance weighted KNN for Given data point in term of Positive or Negative.

CODE :

```
import numpy as np
import pandas as pd
import math

x=np.array([[2,4],[4,2],[4,4],[4,6],[6,2],[6,4]])
y=np.array([0,0,1,0,1,0])
#  0=negative 1=positive class
def eucledian_distance(x1,y1,x2,y2):
    return math.sqrt((x1-x2)**2+(y1-y2)**2)
def chooseK(arr):
    print("Size of array :",arr.shape[0])
    k=round(math.sqrt(arr.shape[0]))
    if(k%2==0):
k=k+1;

    #k should be odd so that classfication can be done
properly(No chance of 50%-50% classification)
    print("Choosen value of K : ",k)
    return k;
chooseK(x)

def classifyPoint(x,y,point,k):
    inputSize=x.shape[0];
    distance=[]; #for string eucledian distance
    for i in range(inputSize):
distance.append(eucledian_distance(point[0],point[1],x[i][0],x
[i][1]));
    mergedList=list(zip(distance,y));
    mergedList.sort(); #sort according to increasing distance
    freq0=0; #Freq of group 0 (negative)
    freq1=0; #Freq of group 1 (positive)
    for i in range(int(k)): #Iterate for k neighbours
        if(mergedList[i][1]==0):
            freq0=freq0+1;
        elif (mergedList[i][1]==1):
            freq1=freq1+1;
    if(freq0>freq1):
        return 0;
    else:
```

```
return 1;

def main():
    print("Input X coordinate");
    x_co=int(input())
    print("Enter Y coordinate")
    y_co=int(input())
    pointt=(x_co,y_co)
    print(pointt)
    k=chooseK(x);
    label="--"
    if(classifyPoint(x=x,y=y,point=pointt,k=k)==0):
        label="Negative";
    else:
        label="Positive";
    print("Point {} belongs to {} class".format(pointt,label))
    print (classifyPoint(x=x,y=y,point=pointt,k=k))
main()
```

Output :

```
 ('Size of array :', 6)
('Choosen value of K : ', 3.0)

Input X coordinate
6
Enter Y coordinate
6
(6, 6)
('Size of array :', 6)

('Choosen value of K : ', 3.0)
Point (6, 6) belongs to Negative class
0
```

ASSIGNMENT No : 4

TITLE : Assignment based on K-Means Clustering

PROBLEM DEFINITION : We have given a collection of 8 points. P1 = [0.1, 0.6] P2 = [0.15, 0.71] P3 = [0.08, 09] P4 = [0.16, 0.85] P5 = [0.2, 0.3] P6 = [0.25, 0.5] P7 = [0.24, 0.1] P8 = [0.3, 0.2]. Perform the K-mean clustering with initial Centroids as m1 = P1 = Cluster #1 = C1 and m2 = P8 = Cluster #2 = C2.

PREREQUISITE : Basic of Python, Data Mining Algorithm, Concept of K-Mean Clustering

SOFTWARE REQUIREMENT : P4, 2 GB RAM, 500 GB HDD

HARDWAR REQUIREMENT : Anaconda with Python 3.7

LEARNING OBJECTIVES : Learn How to Apply KMean Clustering for giving datapoints.

OUTCOMES : After Completion Students are able to implemend for K-mean clustering with initial Centroids.

Teacher Signature_____

PAGE :
DATE : / /

K-Mean

1. How do you solve K mean?

Ans Working →

(a) Choose the number of clusters 'k'.

(b) Select k random points from the data as Centroids

(c) Assign all points to the closest cluster centroid.

(d) Recompute the centroid of newly formed clusters.

(e) Repeat step 3 and 4 until there is no change in new centroid position.

2. What is K-means algorithm with example?

Ans K-means clustering is an unsupervised learning algorithm that is used to assign a cluster to an unknown data point based on it's similarity with other clusters. The goal is to determine the intrinsic grouping in a set of unknown data.

Eg - Based on age, height and weight of a person, divide them into two possible clusters of 'male' and 'female' and then categorize any new person based on its similarity with the above clusters.

PAGE :
DATE : / /

3. What is K-means good for?

Ans Advantages →

(a) Relatively simple to implement.

(b) Scales to large datasets.

(c) Guarantees convergence.

(d) Easily adapts to new examples.

4. What is K in data mining?

Ans In data mining, K-means intends to partition 'n' into 'k' clusters in which each object belongs to the cluster with nearest mean. It is used to classify the data points into 'k' different cluster, each having some distinct feature and representing a group different from the others.

5. What is K in K-means? How is K-means different from KNN?

Ans K is the total number of clusters to be formed in the given unlabelled dataset.

PAGE :
DATE : / /

Difference between KNN and K-Means →

a) KNN is supervised whereas K-Means is unsupervised machine learning.

b) KNN is used for classification or regression whereas K-Means is used for clustering.

c) KNN is lazy learner whereas K-Means is an eager learner.

d) KNN performs better of all data have same scale which is not true for K-Means.

Conclusion: In this way learn K Mean Clustering Algorithm.

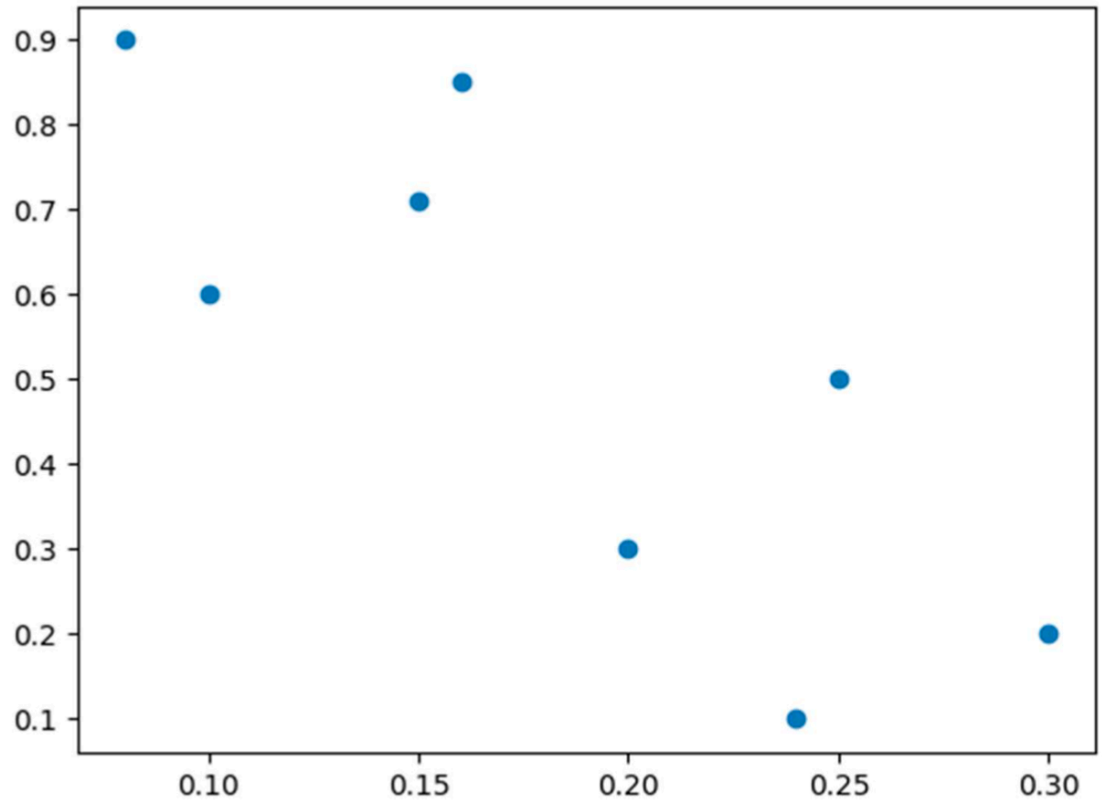
CODE :

```
import numpy as np
import matplotlib.pyplot as plt
import math

x = np.array([0.1,0.15,0.08,0.16,0.2,0.25,0.24,0.3])
y = np.array([0.6,0.71,0.9,0.85,0.3,0.5,0.1,0.2])
plt.plot(x,y,"o")
plt.show()
def eucledian_distance(x1,y1,x2,y2):
    return math.sqrt((x1-x2)**2+(y1-y2)**2)
def manhattan_distance(x1,y1,x2,y2):
        return math.fabs(x1-x2)+math.fabs(y1-y2
def returnCluster(m1,m2,x_co,y_co):
    #if we use manhattan_distance then clusters are classified
more correctly..
    distance1=manhattan_distance(m1[0],m1[1],x_co,y_co)
    distance2=manhattan_distance(m2[0],m2[1],x_co,y_co)
    if(distance1<distance2):
        return 1;
    else:
        return 2;
#initial centroids for cluster1 nd cluster 2
m1=[0.1,0.6]
m2=[0.3,0.2]
#difference and iteration is for controlling iteration
difference = math.inf
threshold=0.02
iteration=0;
while difference>threshold: #use any one condition #iteration
one is easy
    print("Iteration ",iteration, " : m1=",m1, " m2=",m2)
    cluster1=[];
    cluster2=[];
    #step1 assign all points to nearest cluster
    for i in range(0,np.size(x)):
        clusterNumber=returnCluster(m1,m2,x[i],y[i])
        point=[x[i],y[i]]
        if clusterNumber==1:
```

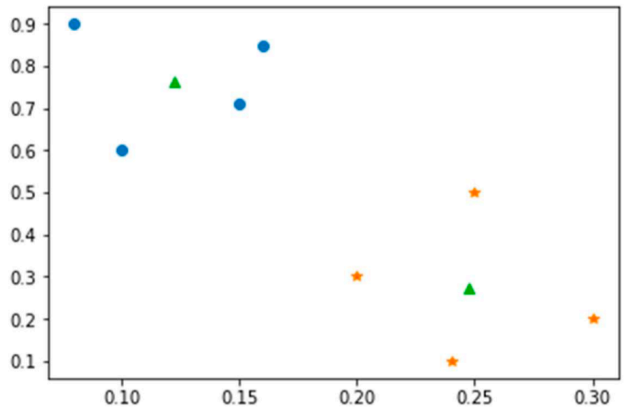
```
        cluster1.append(point);
    else:
        cluster2.append(point)
print("cluster 1", cluster1,"\nCLuster 2: ", cluster2)
#step 2: Calculating new centriod for cluster1 m1_old=m1;
m1=[]
m1=np.mean(cluster1, axis=0) #axis=0 means columnwise

#calculating centroid for cluster2
m2_old=m2;
m2=[];
m2=np.mean(cluster2,axis=0)
print("m1 = ",m1," m2=",m2)
#adjusting diffrences of adjustment between m1 nd m1_old
xAvg=0.0;
yAvg=0.0;
xAvg=math.fabs(m1[0]-m1_old[0])+math.fabs(m2[0]-m2_old[0])
xAvg=xAvg/2;
yAvg=math.fabs(m1[1]-m1_old[1])+math.fabs(m2[1]-m2_old[1])
yAvg=yAvg/2;
if(xAvg>yAvg):
    difference=xAvg;
else:
    difference=yAvg;
print("Difference  : ", difference)
    iteration+=1;
    print("")
#final Output
print("Cluster 1 centroid : m1 = ",m1)
print("CLuster 1 points: ", cluster1)
print("Cluster 2 centroid : m2 = ",m2)
print("CLuster 2 points: ", cluster2)
clust1=np.array(cluster1)
clust2=np.array(cluster2)
#cluster 1 points
plt.plot(clust1[:,0],clust1[:,1],"o")
#cluster2 points
plt.plot(clust2[:,0], clust2[:,1],"*")
#centroids
plt.plot([m1[0],m2[0]],[m1[1],m2[1]],"^")
```

```
plt.show()
#same code
plt.scatter(clust1[:,0],clust1[:,1])
plt.scatter(clust2[:,0],clust2[:,1])
plt.scatter([m1[0],m2[0]],[m1[1],m2[1]],marker="*")
plt.show()
```

OUTPUT :



```
Iteration  0  : m1= [0.1, 0.6]  m2= [0.3, 0.2]

cluster 1 [[0.1, 0.6], [0.15, 0.71], [0.08, 0.9], [0.16,
0.85], [0.25, 0.5]]

CLuster 2:  [[0.2, 0.3], [0.24, 0.1], [0.3, 0.2]]

m1 =  [0.148 0.712]  m2= [0.24666667 0.2       ]

Difference  :  0.05600000000000001


Iteration  1  : m1= [0.148 0.712]  m2= [0.24666667 0.2 ]

cluster 1 [[0.1, 0.6], [0.15, 0.71], [0.08, 0.9], [0.16,
0.85]]

CLuster 2:  [[0.2, 0.3], [0.25, 0.5], [0.24, 0.1], [0.3, 0.2]]

m1 =  [0.1225 0.765 ]  m2= [0.2475 0.275 ]

Difference  :  0.06400000000000002


Iteration  2  : m1= [0.1225 0.765 ]  m2= [0.2475 0.275 ]

cluster 1 [[0.1, 0.6], [0.15, 0.71], [0.08, 0.9], [0.16,
0.85]]



CLuster 2:  [[0.2, 0.3], [0.25, 0.5], [0.24, 0.1], [0.3, 0.2]]

m1 =  [0.1225 0.765 ]  m2= [0.2475 0.275 ]

Difference  :  0.0

Cluster 1 centroid : m1 =  [0.1225 0.765 ]

CLuster 1 points:  [[0.1, 0.6], [0.15, 0.71], [0.08, 0.9],
[0.16, 0.85]]

Cluster 2 centroid : m2 =  [0.2475 0.275 ]

CLuster 2 points:  [[0.2, 0.3], [0.25, 0.5], [0.24, 0.1],
[0.3, 0.2]]
```



Ruff & Tuff
Date
Page

ASSIGNMENT No : 5

TITLE : Implementation of S-DES

LEARNING OBJECTIVES : Learn Data Encryption Standard Algorithm (DES)

PROBLEM DEFINITION : Implementation of S-DES

OUTCOMES : After Completion of this assignment Students will be able to understand the Data Encryption Standard.

SOFTWARE REQUIREMENTS : Python 3

HARDWARE REQUIREMENTS : P4, 2GB RAM, 500GB HDD

Teacher Signature

PAGE :
DATE : / /

DES

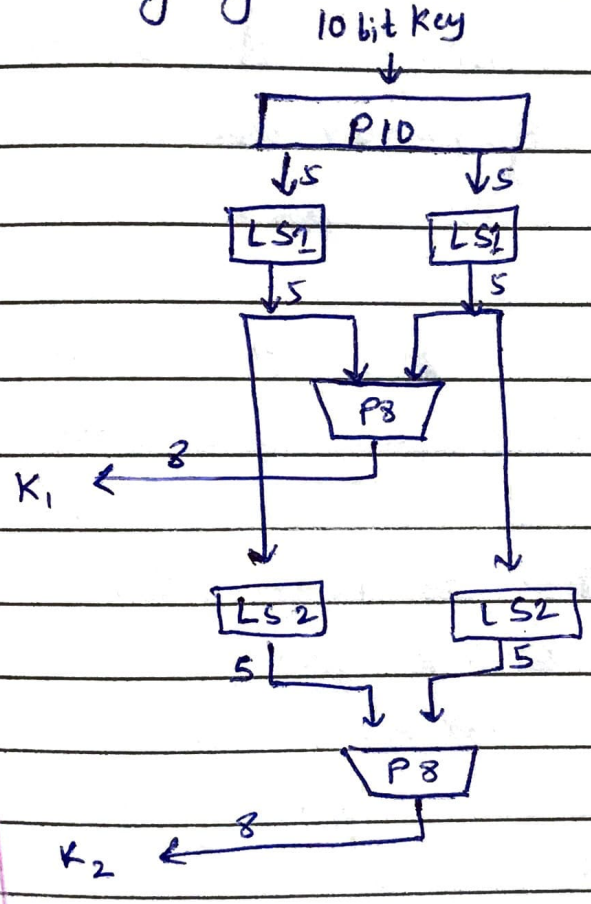
1.

	DES (Data Encryption Standard)	S-DES (Simplified Data Encryption Standard)
Type of Algo-	Symmetric	Symmetric
Cipher type-	Block Cipher	Block Cipher
Block Size-	64 bits	8 bits
Cipher block-	64 bits	8 bits
Key Size	56 bits	10 bits
No. of rounds	16 rounds	16 rounds
Technique	Substitution	Symmetric Key cipher

2. (a) The process begins with the 64 bit plain text block getting handed over to an initial permutation function.
(b) Initial permutation is performed on plain text.
(c) Next, IP creates two halves of permuted block known as left plain text and right plain text.
(d) Each LPT & RPT goes through 16 rounds of encryption process.
(e) Finally, LPT & RPT goes are rejoined to perform

final permutation on combined block.
The result of this process produces the desired 64 bit cipher text.

SDES Key generation



4. Types of attacks on DES-
(a) Differential cryptanalysis (DC)
(b) Linear Cryptanalysis (LC)
(c) Davie's attack.

PAGE :
DATE : / /

DC is a form of cryptanalysis applicable to block ciphers, also Stream ciphers and hash function.
LC is a form of cryptanalysis based on finding affine approximation to the action of a ciphers.
Davies attack is a known plain-text attack based on the non-uniform distribution of the outputs of pairs of adjacent S-boxes.

5. PBox is a key-less fixed transposition ciphes whereas SBox is a key less fixed Substitution ciphes.
An SBox is an M×N Substitution box where M and N may not be same. Each output is a boolean function of inputs. All S-Boxes are non-linear.
PBox (Permutation Box) is a method of bit-Shuffling used to permute or transpose bits across S-boxes inputs.
Diffusion hides the relationship between ciphes text and plain text whereas confusion hides the relationship b/w cipher text and Key.

PAGE :
DATE : /

Conclusion : Thus we learn that to how to Encrypt and Decrypt the message by using DES Algorithm.

CODE :

#Initial permut matrix for the datas

```
PI = [58, 50, 42, 34, 26, 18, 10, 2,
      60, 52, 44, 36, 28, 20, 12, 4,
      62, 54, 46, 38, 30, 22, 14, 6,
      64, 56, 48, 40, 32, 24, 16, 8,
      57, 49, 41, 33, 25, 17, 9, 1,
      59, 51, 43, 35, 27, 19, 11, 3,
      61, 53, 45, 37, 29, 21, 13, 5,
      63, 55, 47, 39, 31, 23, 15, 7]
```

#Initial permut made on the key CP_1 = [57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18, 10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36,

63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22, 14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4]

#Permut applied on shifted key to get Ki+1 CP_2 = [14, 17, 11, 24, 1, 5, 3, 28,

15, 6, 21, 10, 23, 19, 12, 4, 26, 8, 16, 7, 27, 20, 13, 2, 41, 52, 31, 37, 47, 55, 30, 40, 51, 45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32]

#Expand matrix to get a 48bits matrix of datas to apply the xor with Ki

```
E = [32, 1, 2, 3, 4, 5,
     4, 5, 6, 7, 8, 9,
     8, 9, 10, 11, 12, 13,
     12, 13, 14, 15, 16, 17,
     16, 17, 18, 19, 20, 21,
     20, 21, 22, 23, 24, 25,
     24, 25, 26, 27, 28, 29,
     28, 29, 30, 31, 32, 1]
```

S_BOX = [

4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7], 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8], [4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0],

[[14, [0,

12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13],

1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10], 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5], 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15],

8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9],

],

[13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7], [1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12],

```
],

[[7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15], [13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9],
[10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4], [3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14],

],

[[2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9], [14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6],
[4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14], [11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3],

],

[[12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11], [10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8],
[9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6], [4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13],

],

[[4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1], [13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6],
[1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2], [6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12],

],


[[13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7], [1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2],
[7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8], [2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11],

] ]

#Permut made after each SBox substitution for each round
P = [16, 7, 20, 21, 29, 12, 28, 17,
1, 15, 23, 26, 5, 18, 31, 10, 2, 8, 24, 14, 32, 27, 3, 9, 19, 13, 30, 6, 22, 11, 4, 25]

#Final permut for datas after the 16 rounds PI_1 = [40, 8, 48, 16, 56, 24, 64, 32,
39, 7, 47, 15, 55, 23, 63, 31,
38, 6, 46, 14, 54, 22, 62, 30,
37, 5, 45, 13, 53, 21, 61, 29,
36, 4, 44, 12, 52, 20, 60, 28,
35, 3, 43, 11, 51, 19, 59, 27,
34, 2, 42, 10, 50, 18, 58, 26,

33, 1, 41, 9, 49, 17, 57, 25]

#Matrix that determine the shift for each round of keys SHIFT = [1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1]

def string_to_bit_array(text):#Convert a string into a list of bits array = list()

    for char in text:
        binval = binvalue(char, 8)#Get the char value on one byte
array.extend([int(x) for x in list(binval)]) #Add the bits to the final list

return array

def bit_array_to_string(array): #Recreate the string from the bit array
```

```
res = ''.join([chr(int(y,2)) for y in [''.join([str(x) for x in _bytes]) for _bytes in nsplit(array,8)]])

return res

def binvalue(val, bitsize): #Return the binary value as a string of the given size

binval = bin(val)[2:] if isinstance(val, int) else bin(ord(val))[2:]

if len(binval) > bitsize:
raise "binary value larger than the expected size"

    while len(binval) < bitsize:
binval = "0"+binval #Add as many 0 as needed to get the wanted size

    return binval
def nsplit(s, n):#Split a list into sublists of size "n" return [s[k:k+n] for k in range(0, len(s), n)]

ENCRYPT=1
DECRYPT=0

class des():
    def __init__(self):
        self.password = None
        self.text = None
        self.keys = list()
def run(self, key, text, action=ENCRYPT, padding=False): if len(key) < 8:

raise "Key Should be 8 bytes long" elif len(key) > 8:

key = key[:8] #If key size is above 8bytes, cut to be


8bytes long

        self.password = key
        self.text = text
        if padding and action==ENCRYPT:
            self.addPadding()
elif len(self.text) % 8 != 0:#If not padding specified data size must be multiple of 8 bytes

            raise "Data size should be multiple of 8"
self.generatekeys() #Generate all the keys

text_blocks = nsplit(self.text, 8) #Split the text in blocks of 8 bytes so 64 bits

        result = list()
        for block in text_blocks:#Loop over all the blocks of data
bit array
permutation
(48bits)

Ki
```

```
block = string_to_bit_array(block)#Convert the block in block = self.permut(block,PI)#Apply the
initial

g, d = nsplit(block, 32) #g(LEFT), d(RIGHT) tmp = None
for i in range(16): #Do the 16 rounds

d_e = self.expand(d, E) #Expand d to match Ki size

    if action == ENCRYPT:
        tmp = self.xor(self.keys[i], d_e)#If encrypt use
else:

                tmp = self.xor(self.keys[15-i], d_e)#If decrypt
start by the last key
the SBOXes

tmp = self.substitute(tmp) #Method that will apply
tmp = self.permut(tmp, P)
tmp = self.xor(g, tmp)
g=d

d = tmp

            result += self.permut(d+g, PI_1) #Do the last permut and
append the result to result
        final_res = bit_array_to_string(result)
        if padding and action==DECRYPT:
            return self.removePadding(final_res) #Remove the padding
if decrypt and padding is true
else:

            return final_res #Return the final string of data
ciphered/deciphered
def substitute(self, d_e):#Substitute bytes using SBOX
subblocks = nsplit(d_e, 6)#Split bit array into sublist of 6 bits

result = list()
for i in range(len(subblocks)): #For all the sublists
    block = subblocks[i]

            row = int(str(block[0])+str(block[5]),2)#Get the row
with the first and last bit
            column = int(''.join([str(x) for x in block[1:][:-
1]]),2) #Column is the 2,3,4,5th bits
val = S_BOX[i][row][column] #Take the value in the SBOX appropriated for the round (i)

bin = binvalue(val, 4)#Convert the value to binary

            result += [int(x) for x in bin]#And append it to the
resulting list
        return result
def permut(self, block, table):#Permut the given block using the given table (so generic method)
```

```
        return [block[x-1] for x in table]
def expand(self, block, table):#Do the exact same thing than permut but for more clarity has been
renamed

        return [block[x-1] for x in table]


def xor(self, t1, t2):#Apply a xor and return the resulting list return [x^y for x,y in zip(t1,t2)]

def generatekeys(self):#Algorithm that generates all the keys self.keys = []

        key = string_to_bit_array(self.password)
key = self.permut(key, CP_1) #Apply the initial permut on the key

g, d = nsplit(key, 28) #Split it in to (g->LEFT),(d->RIGHT) for i in range(16):#Apply the 16 rounds

g, d = self.shift(g, d, SHIFT[i]) #Apply the shift associated with the round (not always 1)

            tmp = g + d #Merge them
            self.keys.append(self.permut(tmp, CP_2)) #Apply the
permut to get the Ki
def shift(self, g, d, n): #Shift a list of the given value return g[n:] + g[:n], d[n:] + d[:n]

def addPadding(self):#Add padding to the datas using PKCS5 spec. pad_len = 8 - (len(self.text) %
8)
self.text += pad_len * chr(pad_len)

def removePadding(self, data):#Remove the padding of the plain text (it assume there is padding)

        pad_len = ord(data[-1])
        return data[:-pad_len]
def encrypt(self, key, text, padding=False): return self.run(key, text, ENCRYPT, padding)

def decrypt(self, key, text, padding=False): return self.run(key, text, DECRYPT, padding)



if __name__ == '__main__':
    print("Name: Atharva Abhay Karkhanis ")
    key = "secret_key"
    text= "Hello world"
    d = des()
    r = d.encrypt(key,text,padding=True)
    r2 = d.decrypt(key,r,padding=True)
    print("Ciphered: %r" % r)
    print("Deciphered: ", r2
```

OUTPUT :

```
Ciphered: 'ßåýåÚ\x9f\\\x9d\x89r\x9c\x16Û\x0fá\x8b'
Deciphered:  Hello world.
```

ASSIGNMENT No : 6

TITLE: Implementation of S-AES

LEARNING OBJECTIVES: Learn How to apply Advanced Encryption Standard Algorithm to encryption of given data.

PROBLEM DEFINITION: Implementation of S-AES

SOFTWARE REQ: Python 3

HARDWARE REQ: P4, 2GB RAM, 500GB HDD

OUTCOMES: After Completion of this assignment Students will be able to implement code for Advanced Encryption Standard Algorithm for given data and find the encrypted data of the given data.

Teacher Signature

PAGE :
DATE : / /

AES

1.

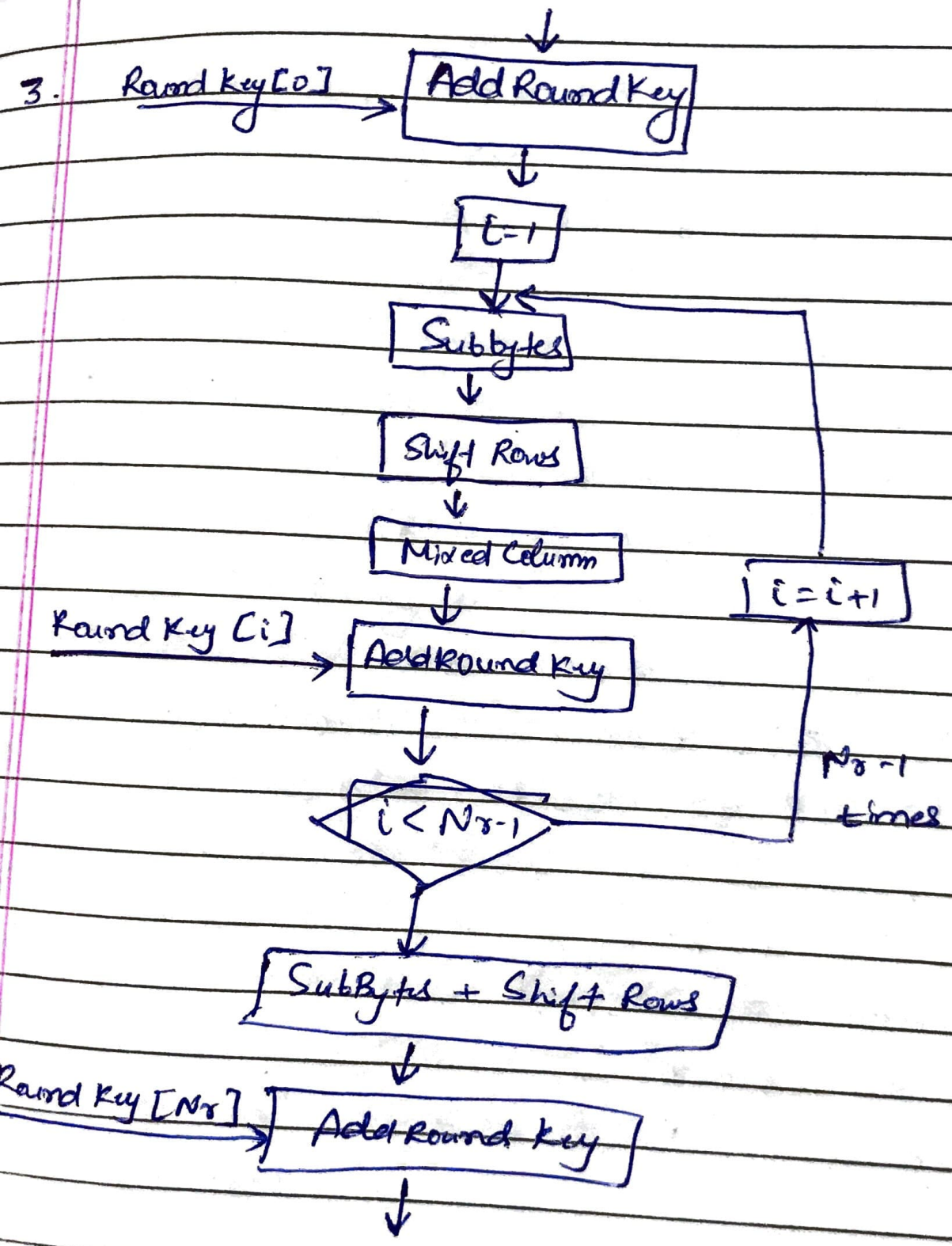
	AES	S-AES
Type of Algo	Symmetric	Symmetric
block size	128 bits	16 bits
Key size	128/196/256 bits	16 bits
Cipher type	block cipher	block ciphe
no. of rounds	10/12/14 rounds	2 rounds
technique	Transposition	Transposition

2. AES algorithm uses Substitution permutation or SP networks to produce cipher text. The no. of rounds depends on key size being used. Each of these rounds require a round key but since only one key is inputted in the algorithm, this key needs to be expanded to get keys for each round including round 0.
Steps in each round -

(a) Substitution of bytes
(b) Shifting of rows
(c) Mixing of Columns
(d) Adding the round key

PAGE :
DATE : / /

3.


Round key[0]
AddRoundKey
i=1
SubBytes
Shift Rows
Mixed Column
Round Key [i]
AddRound Key
i < Nr-1
i = i+1
Nr-1 times
SubBytes + Shift Rows
Round Key [Nr]
Add Round Key

PAGE:
DATE: / /

4 There are practical side-channel attacks against implementation of AES but there are not any practical cryptanalysis attacks against the abstract algorithm.
The best Cryptanalysis attack against Complete cipher in a reasonable attack model is the biclique attack which only knocks a few bits off the key. The related key attacks against AES have attack models which are unreasonable and so are less applicable even though the bottom line of the computational cost of the attack is more pronounced.

5 Transposition is a simple movement of data to an alternate location. There are two relevant steps in AES when it comes to transposition.

(a) Add Round Key :- It incorporates a bitwise rotate left.

(b) Shift Rows :- It is transposition of bytes in the state. It is a byte wise transposition.

PAGE :
DATE : /

Conclusion : Thus we learn that to how to Encrypt and Decrypt the message by using AES Algorithm.

CODE :

```
import sys
# S-Box
sBox  = [0x9, 0x4, 0xa, 0xb, 0xd, 0x1, 0x8, 0x5,
        0x6, 0x2, 0x0, 0x3, 0xc, 0xe, 0xf, 0x7]
# Inverse S-Box
sBoxI = [0xa, 0x5, 0x9, 0xb, 0x1, 0x7, 0x8, 0xf,
        0x6, 0x0, 0x2, 0x3, 0xc, 0x4, 0xd, 0xe]
# Round keys: K0 = w0 + w1; K1 = w2 + w3; K2 = w4 + w5
w = [None] * 6
def mult(p1, p2):
"""Multiply two polynomials in GF(2^4)/x^4 + x + 1""" p=0
while p2:

        if p2 & 0b1:
            p ^= p
        p1 <<= 1
        if p1 & 0b10000:
            p1 ^= 0b11
        p2 >>= 1
    return p & 0b1111
def intToVec(n):
"""Convert a 2-byte integer into a 4-element vector""" return[n>>12,(n>>4)&0xf,(n>>8)&0xf,
n&0xf]

def vecToInt(m):
    """Convert a 4-element vector into 2-byte integer"""
    return (m[0] << 12) + (m[2] << 8) + (m[1] << 4) + m[3]
def addKey(s1, s2):
    """Add two keys in GF(2^4)"""
    return [i ^ j for i, j in zip(s1, s2)]
def sub4NibList(sbox, s):
    """Nibble substitution function"""
    return [sbox[e] for e in s]
def shiftRow(s):
    """ShiftRow function"""
    return [s[0], s[1], s[3], s[2]]
def keyExp(key):
    """Generate the three round keys"""
    def sub2Nib(b):
        """Swap each nibble and substitute it using sBox"""
        return sBox[b >> 4] + (sBox[b & 0x0f] <<
    Rcon1, Rcon2 = 0b10000000, 0b00110000
    w[0] = (key & 0xff00) >> 8
    w[1] = key & 0x00ff
    w[2] = w[0] ^ Rcon1 ^ sub2Nib(w[1])
    w[3] = w[2] ^ w[1]
    w[4] = w[2] ^ Rcon2 ^ sub2Nib(w[3])
    w[5] = w[4] ^ w[3]
def encrypt(ptext):
    """Encrypt plaintext block"""
```

```
    def mixCol(s):
        return [s[0] ^ mult(4, s[2]), s[1] ^ mult(4, s[3]),
                s[2] ^ mult(4, s[0]), s[3] ^ mult(4, s[1])]
    state = intToVec(((w[0] << 8) + w[1]) ^ ptext)
    state = mixCol(shiftRow(sub4NibList(sBox, state)))
    state = addKey(intToVec((w[2] << 8) + w[3]), state)
    state = shiftRow(sub4NibList(sBox, state))
    return vecToInt(addKey(intToVec((w[4] << 8) + w[5]),
state))
def decrypt(ctext):
    """Decrypt ciphertext block"""
    def iMixCol(s):
        return [mult(9, s[0]) ^ mult(2, s[2]), mult(9, s[1]) ^
mult(2, s[3]),
                mult(9, s[2]) ^ mult(2, s[0]), mult(9, s[3]) ^
mult(2, s[1])]
    state = intToVec(((w[4] << 8) + w[5]) ^ ctext)
    state = sub4NibList(sBoxI, shiftRow(state))
state = iMixCol(addKey(intToVec((w[2] << 8) + w[3]), state))

    state = sub4NibList(sBoxI, shiftRow(state))
    return vecToInt(addKey(intToVec((w[0] << 8) + w[1]),
state))
if __name__ == '__main__':
# Test vectors from "Simplified AES" (Steven Gordon) # (http://hw.siit.net/files/001283.pdf)
print("Name: Atharva Abhay Karkhanis ")
plaintext = 0b1101011100101000
key = 0b0100101011110101
ciphertext = 0b0010010011101100
keyExp(key)
try:

        assert encrypt(plaintext) == ciphertext
    except AssertionError:
        print("Encryption error")
        print(encrypt(plaintext), ciphertext)
        sys.exit(1)
    try:
        assert decrypt(ciphertext) == plaintext
    except AssertionError:
        print("Decryption error")
        print(decrypt(ciphertext), plaintext)
        sys.exit(1)
    print("Test ok!")
    sys.exit()
```

OUTPUT :

```
Test ok!

Encryption error
34029 9452
```

Ruff & Tuff
Date
Page

Assignment No: 07

TITLE: Implementation of Diffie-Hellman Key Exchange (DH)

LEARNING OBJECTIVES: Learn Diffie-Hellman Key Exchange (DH)

PROBLEM DEFINITION: Implementation of Driffie Hellman Key Exchange

SOFTWARE REQ: Python 3

HARDWARE REQ: P4, 2GB RAM, 500 MB HDD

OUTCOMES: After Completion of this assignment Students will be able to understand the Diffie Hellman Key exchange.

PAGE:
DATE: / /

Diffie Hellmen Key exchange

1. Diffie Hellmen Key exchange uses large numbers and a lot of computation for cryptography. Both parties of get the same result without sending the shared secret across the communication channel. If some adversary peeps into the exchange, they could only access some standard data.
It allows two parties to communicate over an unsecured connection and still comes up with a secret that can be used for making encryption key for future communication.
It uses following protocols and techniques -
(a) Eliptic - curve Diffe Hellman.
(b) TLS
(c) Elgamon
(d) Station to Station (STS)

2. Weakness →
(a) The algorithm cannot be used for any asymmetric key exchange.
(b) It cannot be used for signing digital signatures.

PAGE :
DATE : / /

(c) Since it does not authenticate any party in transmission, it is prone to man in the middle attack.

3.

There are two D-H key exchange. User 1 and Attacker share same key (K_1) and attacker and user 2 share the other key (K_2).
Because user 1 and user 2 had no authentication or prior knowledge of each other. But the attacker must keep listening and forwarding.

4. Diffie Hellman uses a private-public key pair to establish a shared secret usually a symmetric key. It is not a symmetric algorithm used to establish a shared secret for symmetric key algorithms. Authenticated key argument exchange a union key in key exchange protocol which also authenticate the identities of parties involved in key exchange.

PAGE :
DATE : / /

5. Advantages →

a) Sender and receiver does not need any prior knowledge of each other.

b) Sharing of secret key is safe.

c) Once the keys are exchanged, the communication of data can be done through an insecure channel.

Conclusion : Thus we have studied and implement Diffie-Hellman key exchange

CODE :

```
import socket

def cal(g,a,p):
    res = pow(g,a,p)
    return(res)
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('127.0.0.1', 8080))
print(client.recv(100))
p = input("Enter first prime Number(P):")
g = input("Enter second prime number (G):")
client.send(str(p))
client.send(str(g))
b = input("Enter a number specific for User 2 :")
print("Calculating the value of B ")
B = cal(g,b,p)
print("value of B :" + str(B))
client.send(str(B))
A = client.recv(100)
print("Recieved value of A :" + A)
print("Calculating the value of Key K1 ")
K1 = cal(int(A),b,p)
print("value of Key K1 is :" + str(K1))
client.close()
```

OUTPUT :

Server Output:

```
I am server
Enter first prime Number(P):13
Enter second prime number (G):7
Enter a number specific for User 2 :3
Calculating the value of B
value of B :5
Recieved value of A :11
Calculating the value of Key K1
value of Key K1 is :5
```

Client Output:

```
First Prime Number : 13,Second Prime Number : 7
Enter a number specific for User 1 :5
Calculating the value of A
value of A :11
Received value of B :5
Calculating the value of Key K2
value of Key K2 is :5
client is disconnected
```

Ruff & Tuff
Date
Page

ASSIGNMENT No : 8

TITLE : Implementation of RSA algorithm

LEARNING OBJECTIVES : Learn RSA Algorithm

PROBLEM DEFINITION : Implementation of RSA Algorithm

SOFTWARE REQ : Python 3.7

HARDWARE REQ : 2GB RAM, 500GB HDD

OUTCOMES : After Completion of this assignment Students will be able to understand the how to encrypt and decrypt messages.

Teacher Signature

PAGE :
DATE : / /

RSA Algorithm

1. RSA is asymmetric cryptography algorithm. It works on two different key - Public and Private. Public key is given to everyone and private key is private.
Since it's asymmetric nobody else except browser can decrypt the data even if third party has public key of the browser.

2. Examples →
(a) A client sends its public key to the server and request for some data.
(b) The server encrypts data using client's public key and sends encrypted data
(c) Client receives it and decrypts it.

3. Symmetric encryption algorithm
(a) Advanced Encryption Standard (AES)
(b) Data Encryption Standard (DES)
(c) International data encryption algorithm (IDEA)
(d) Blowfish
(e) Rivest cipher 4, 5 and 6.

PAGE:
DATE: / /

4. Asymmetric encryption algorithm
(a) Diffie Hellman Key exchange
(b) Digital Signature Standard (DSS)
(c) El Gamal
(d) Eliptic curve cryptography
(e) Pallier cryptography.

5. ECC is probably better for most purpose but each of them have their own advantages and disadvantages.

Advantages of ECC -
(a) Very fast Key generation
(b) Smaller key, cipher texts and signatures
(c) Moderately fast encryption & decryption
(d) Binary curves are fast in hardware
(e) Right protocols for authenticated key exchange.

Disadvantages of ECC -
(a) Complicated & tricky to implement securely.
(b) Newer algorithm could have unknown weakness
(c) Public Key operations are slow.

Ruff & Tuff
Date
Page

Advantages RSA -

a. Easier to implement and understand
b. Signing and decryption are similar encryption and verification are similar
c. Widely deployed, better industry support

Disadvantages RSA.

a. Very slow key generation
b. Slow Signing and decryption
c. Two part Key is vulnerable to CCD attack if poorly implemented.

Conclusion : Thus we learn that to how to encrypt and Decrypt the message by using RSA algorithm.

Teacher Signature

CODE :

```
try:
   input = raw_input
except NameError:
pass try:

   chr = unichr
except NameError:
pass

p=int(input('Enter prime p: '))
q=int(input('Enter prime q: '))
print("Choosen primes:\np=" + str(p) + ", q=" + str(q) + "\n")
n=p*q

print("n = p * q = " + str(n) + "\n")
phi=(p-1)*(q-1)
print("Euler's function (totient) [phi(n)]: " + str(phi) +
"\n")
def gcd(a, b):
while b != 0: c=a%b

a=b

b=c return a

def modinv(a, m):
    for x in range(1, m):
        if (a * x) % m == 1:
            return x
    return None
def coprimes(a):
    l = []
    for x in range(2, a):
        if gcd(a, x) == 1 and modinv(x,phi) != None and x <
(p-1) and x < (q-1):
            l.append(x)
    for x in l:
        if x == modinv(x,phi):
            l.remove(x)
    return l
def encrypt(p,k, plaintext):
    #Unpack the key into it's components
key, n = p,k

#Convert each letter in the plaintext to numbers based on the character using a^b mod m

    cipher = [(ord(char) ** key) % n for char in plaintext]
    #Return the array of bytes
    return cipher
def decrypt(p,k, ciphertext):
    #Unpack the key into its components
```

```
    key, n = p,k

#Generate the plaintext based on the ciphertext and key
using a^b mod m
    plain = [chr((char ** key) % n) for char in ciphertext]
    #Return the array of bytes as a string
    return ''.join(plain)
print("Choose an e from a below coprimes array:\n")
print(str(coprimes(phi)) + "\n")
e=int(input())
d=modinv(e,phi)
print("\nYour public key is a pair of numbers (e=" + str(e) +
", n=" + str(n) + ").\n")
print("Your private key is a pair of numbers (d=" + str(d) +
", n=" + str(n) + ").\n")
plaintext=input("Enter plaintext : ")
print ("\n\nEncrypting message with public key (", d,",",n ,")
. . .")
print ("\nYour ciphertext is:")
emsg = encrypt(d,n,plaintext)
print ("\t\t",''.join(map(lambda x: str(x), emsg)))
print ("\n\nDecrypting message with public key (", e,",",n ,")
. . .")
print ("\nYour message is:")
print ("\t\t",decrypt(e,n, emsg))
```

OUTPUT :

```
Enter prime p: 17
Enter prime q: 11
Chosen primes:
p=17, q=11

n = p * q = 187
Euler's function (totient) [phi(n)]: 160
Choose an e from a below coprimes array:
[3, 7, 9]
9
Your public key is a pair of numbers (e=9, n=187). Your private key is a pair of numbers (d=89,
n=187). Enter plaintext : hello, how are you?

Encrypting message with public key ( 89 , 187 ) . . .
Your ciphertext is:
53501811811111433253111170325124503212111115173
Decrypting message with public key (9, 187) . . .
Your message is:
                hello, how are you?
```