Group A: Design and Analysis of Algorithms

1. Write a program non-recursive and recursive program to calculate Fibonacci numbers and analyze their time and space complexity.

Theory:

Fibonacci series is a series of natural numbers where next number is equivalent to the sum of previous two numbers i.e. fn = fn-1 + fn-2.In fibonacci sequence each item is the sum of the previous two. So, you wrote a recursive algorithm, for example, recursive function example for up to 5

fibonacci(5) = fibonacci(4) + fibonacci(3)

fibonacci(3) = fibonacci(2) + fibonacci(1)

fibonacci(4) = fibonacci(3) + fibonacci(2)

fibonacci(2) = fibonacci(1) + fibonacci(0)

The first two numbers in the Fibonacci sequence are either 1 and 1, or 0 and 1,

and each subsequent number is the sum of the previous two numbers.

Fibonacci Series Logic:

Initializing number.

the nth term is the sum of (n-1)th term and (n-2)th term

Call Same Function Until Origin Condition

Iterative Algorithm

```
1    Algorithm Fibonacci(n)
2    // Compute the nth Fibonacci number.
3    {
4        if (n ≤ 1) then
5            write (n);
6        else
7        {
8            fnm2 := 0; fnm1 := 1;
9            for i := 2 to n do
10           {
11               fn := fnm1 + fnm2;
12               fnm2 := fnm1; fnm1 := fn;
13           }
14           write (fn);
15       }
16   }
```

Analysis

- Two cases (1) n = 0 or 1 and (2) n > 1.

1) When n = 0 or 1, lines 4 and 5 get executed once each. Since each line has an s/e of 1, the total step count for this case is 2.

2) When n > 1, lines 4, 8, and 14 are each executed once. Line 9 gets executed n times, and lines 11 and 12 get executed n-1 times each. Line 8 has an s/e of 2, line 12 has an s/e of 2, and line 13 has an s/e of 0. The remaining lines that get executed have s/e's of 1. The total steps for the case n > 1 is therefore 4n + 1.

Iterative Program

Program to display the Fibonacci sequence up to n-th term

```
nterms = int(input("Enter number of terms "))
# first two terms
n1, n2 = 0, 1
count = 0
# check if the number of terms is valid
if nterms <= 0:
   print("Please enter a positive integer")
# if there is only one term, return n1
elif nterms == 1:
   print("Fibonacci sequence upto", nterms,":")
   print(n1)
# generate fibonacci sequence
else:
   print("Fibonacci sequence:")
   while count < nterms:
      print(n1)
      nth = n1 + n2
      # update values
      n1 = n2
      n2 = nth
      count += 1
```

Output

Enter number of terms 4 Fibonacci sequence:

0

1

1

2

Recursive Algorithm

```
Algorithm  rFibonacci(n)
{
        if (n <= 1)
                return n;
        else
                return rFibonacci(n - 1) + rFibonacci(n - 2); }
```

Analysis

$T(n) = T(n-1) + T(n-2) + c$

$= 2T(n-1) + c$ //from the approximation $T(n-1) \sim T(n-2)$

$= 2*(2T(n-2) + c) + c$

$= 4T(n-2) + 3c$

$= 8T(n-3) + 7c$

$= 2^k * T(n - k) + (2^k - 1)*c$

Let's find the value of k for which: $n - k = 0$

$k = n$

$T(n) = 2^n * T(0) + (2^n - 1)*c$

$= 2^n * (1 + c) - c$

$T(n) = 2^n$

Recursive Program

```
def fibonacci(n):
    if(n <= 1):
        return n
    else:
        return(fibonacci(n-1) + fibonacci(n-2))
n = int(input("Enter number of terms:"))
print("Fibonacci sequence:")
for i in range(n):
    print(fibonacci(i))
```

Output

Enter number of terms:4 Fibonacci sequence:

0

1

1

2

2. Write a program to implement Huffman Encoding using a greedy strategy.

Theory

Huffman coding is a lossless data compression algorithm. The idea is to assign variable-length

codes to input characters, lengths of the assigned codes are based on the frequencies

of corresponding characters. The most frequent character gets the smallest code and the

least frequent character gets the largest code.The variable-length codes assigned to input characters are

Prefix Codes, means the codes (bit sequences) are assigned in such a way that the code assigned

to one character is not the prefix of code assigned to any other character. This is how

Huffman Coding makes sure that there is no ambiguity when decoding the generated bitstream.
Let us understand prefix codes with a counter example. Let

there be four characters a, b, c and d, and their corresponding

variable length codes be 00, 01, 0 and 1. This coding leads to

ambiguity because code assigned to c is the prefix of codes

assigned to a and b. If the compressed bit stream is 0001, the de-compressed

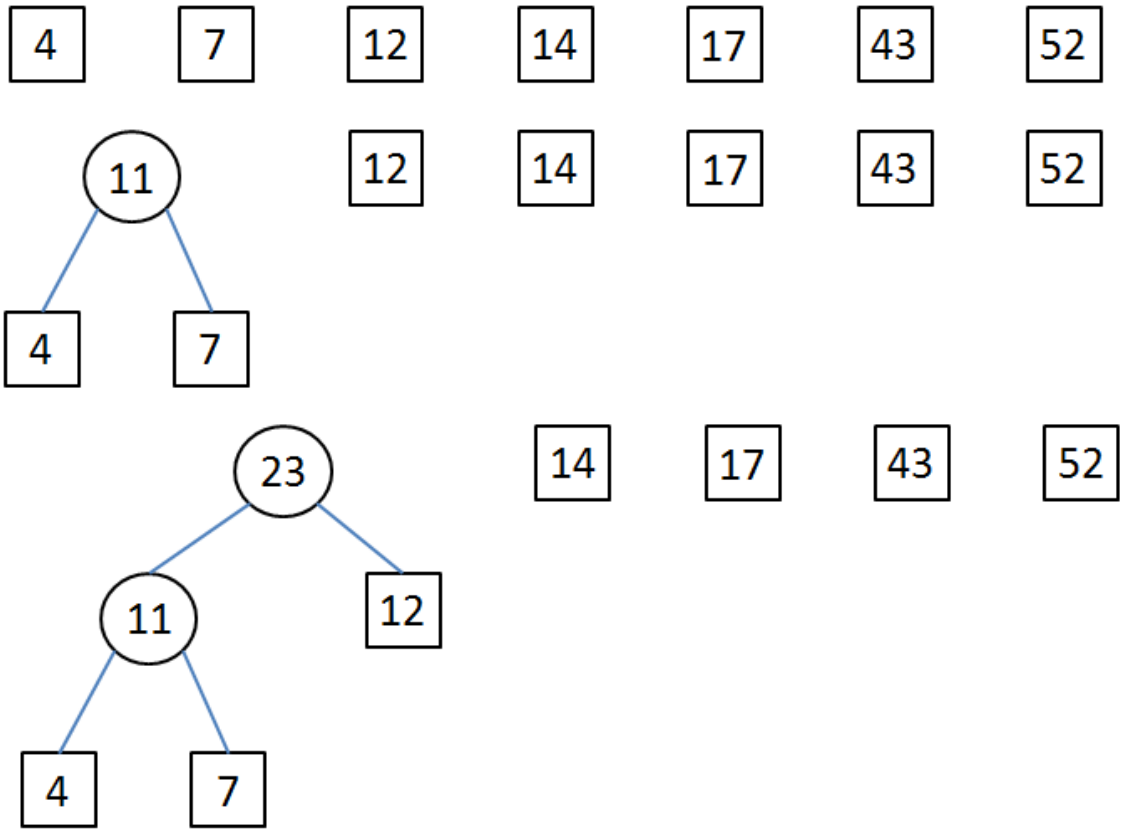
output may be “cccd” or “ccb” or “acd” or “ab”.
See this for applications of Huffman Coding.
There are mainly two major parts in Huffman Coding

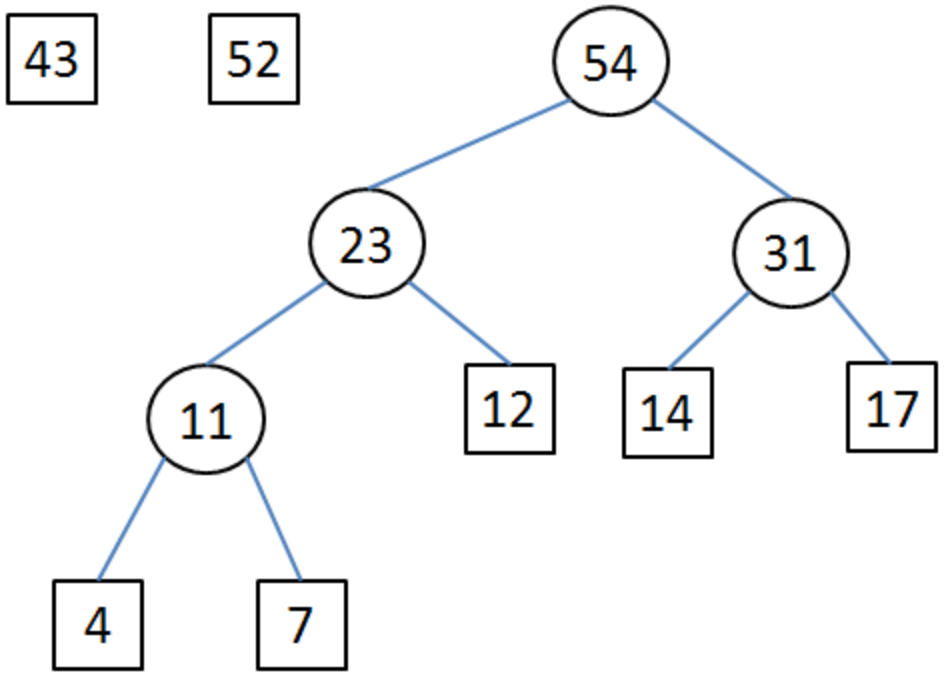
Build a Huffman Tree from input characters.

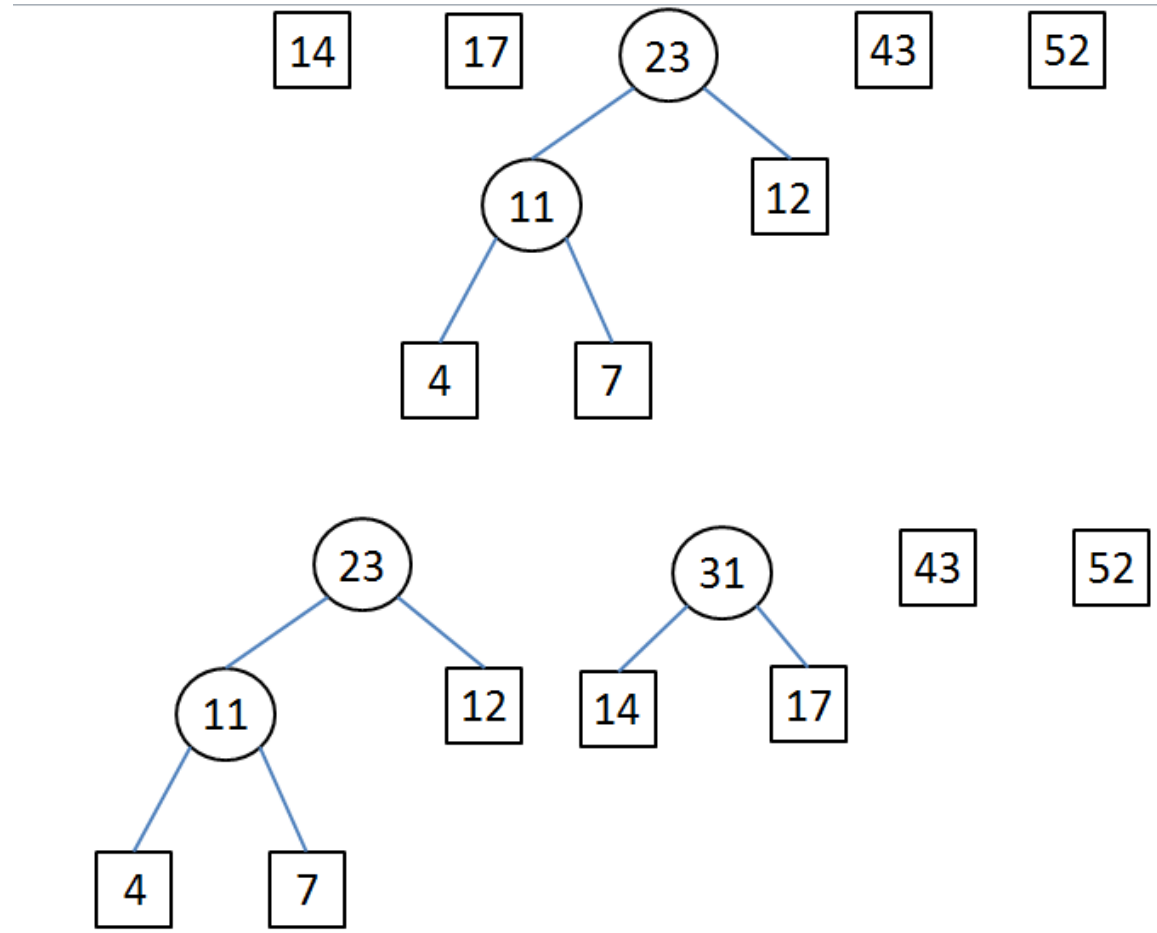
Traverse the Huffman Tree and assign codes to characters.

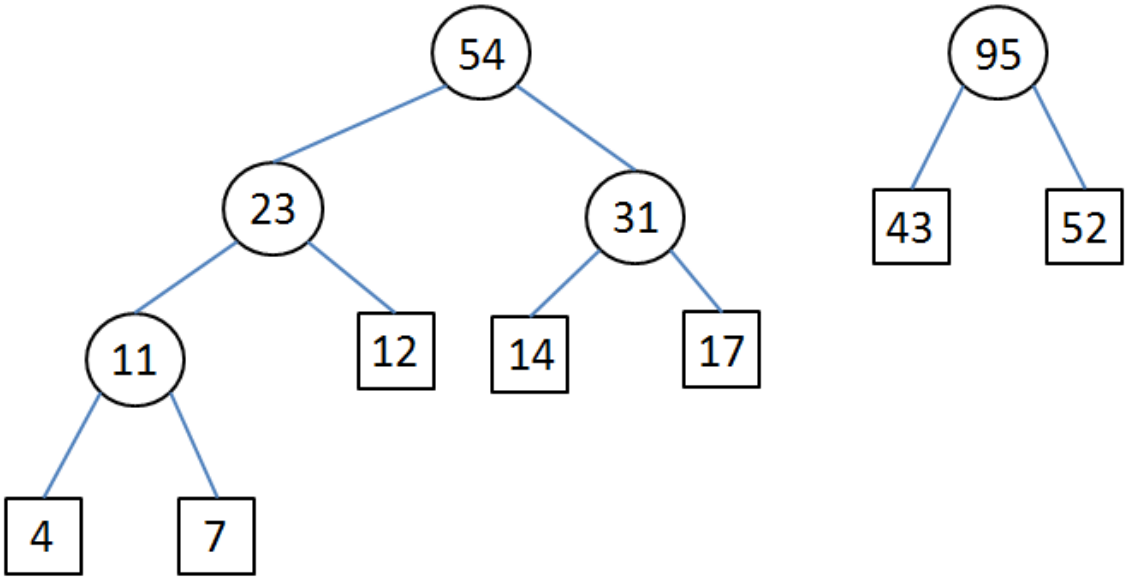
- Steps to build Huffman Tree
  Input is an array of unique characters along with their frequency of occurrences and output is Huffman Tree.
- Create a leaf node for each unique character and build a min heap of all leaf nodes (Min Heap is used as a priority queue.
- The value of frequency field is used to compare two nodes in min heap. Initially, the least frequent character is at root)
- Extract two nodes with the minimum frequency from the min heap.

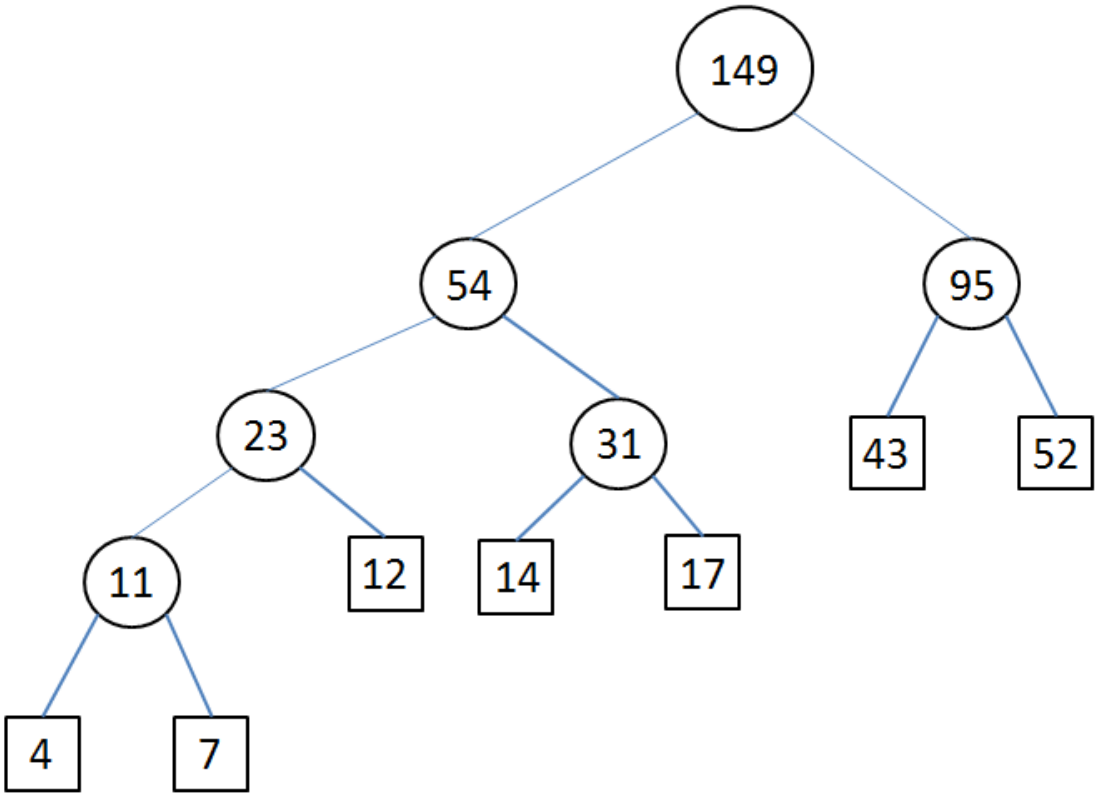
- Create a new internal node with a frequency equal to the sum of the two nodes frequencies. Make the first extracted node
- as its left child and the other extracted node as its right child. Add this node to the min heap.
- Repeat steps#2 and #3 until the heap contains only one node. The remaining node is the root node and the tree is complete.
  Let us understand the algorithm with an example:

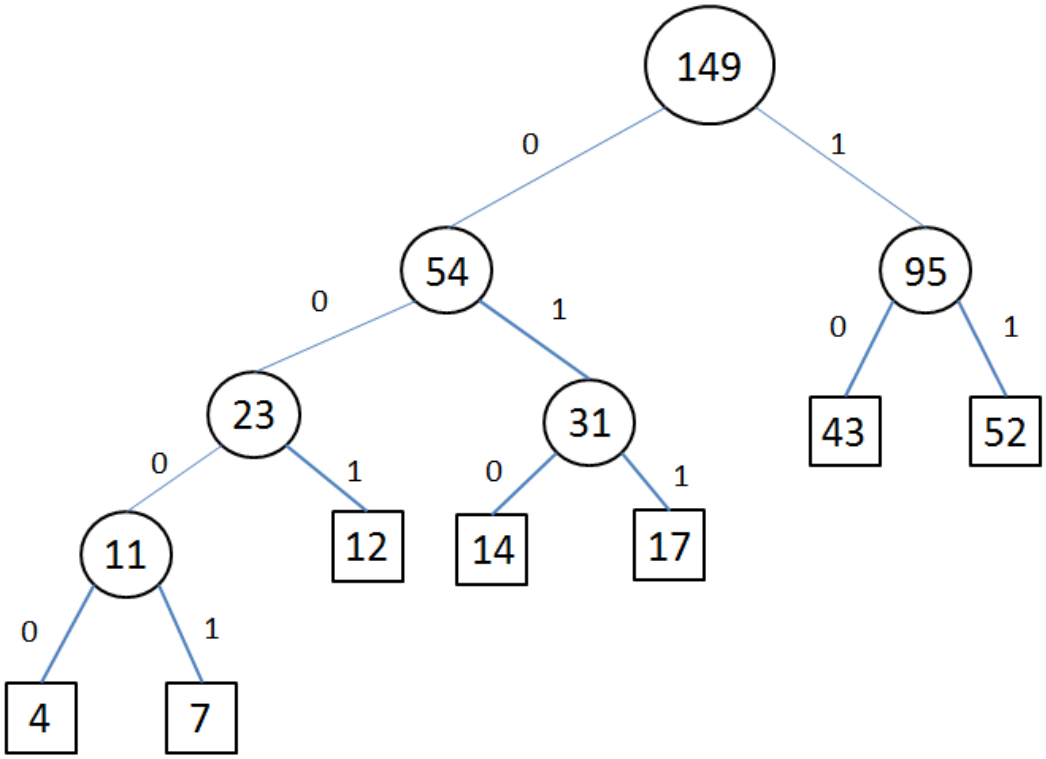


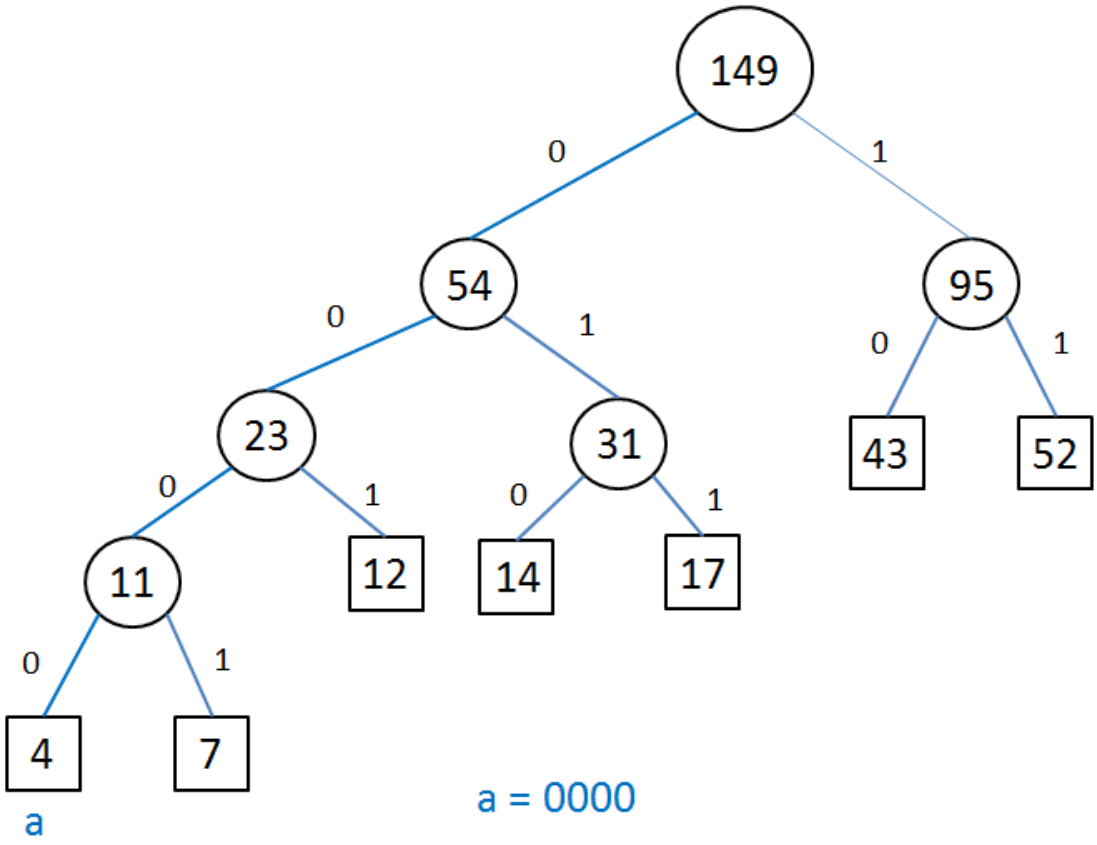

43
52
54
23
31
11
12
14
17
4
7

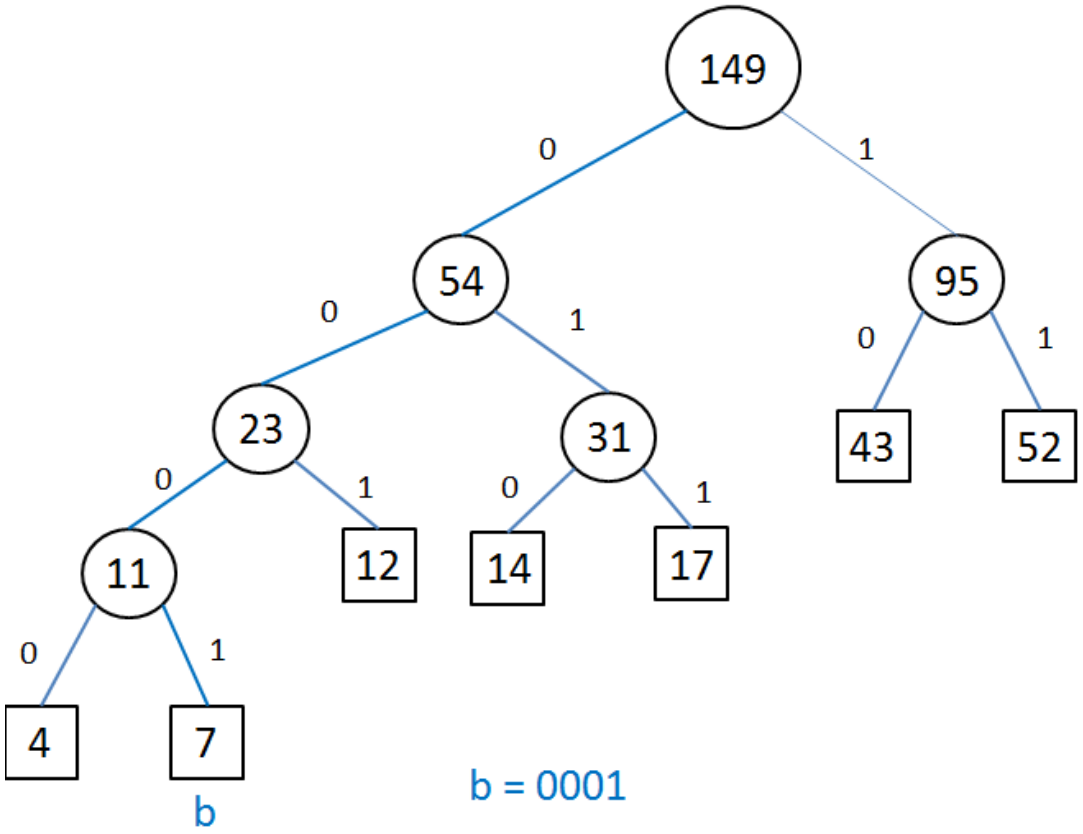

14
17
23
43
52
11
12
4
7
23
31
43
52
11
12
14
17
4
7

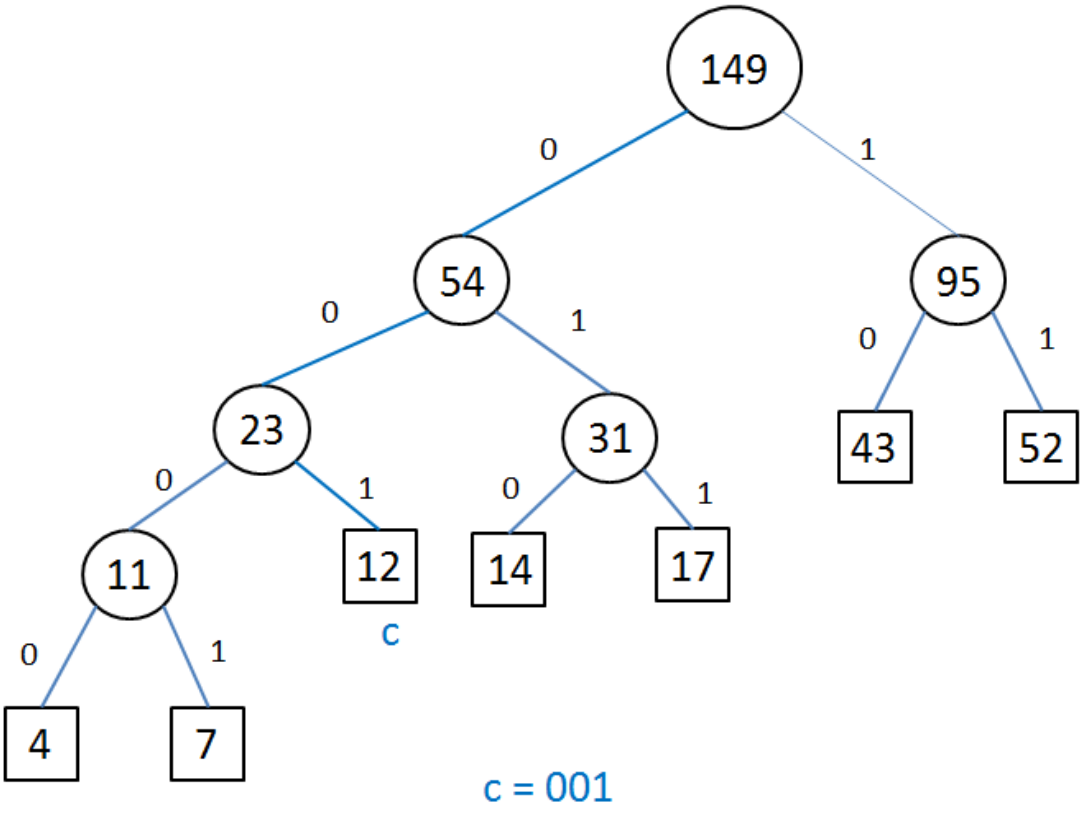

54
23
31
11
12
14
17
4
7
95
43
52


149
54
95
23
31
43
52
11
12
14
17
4
7

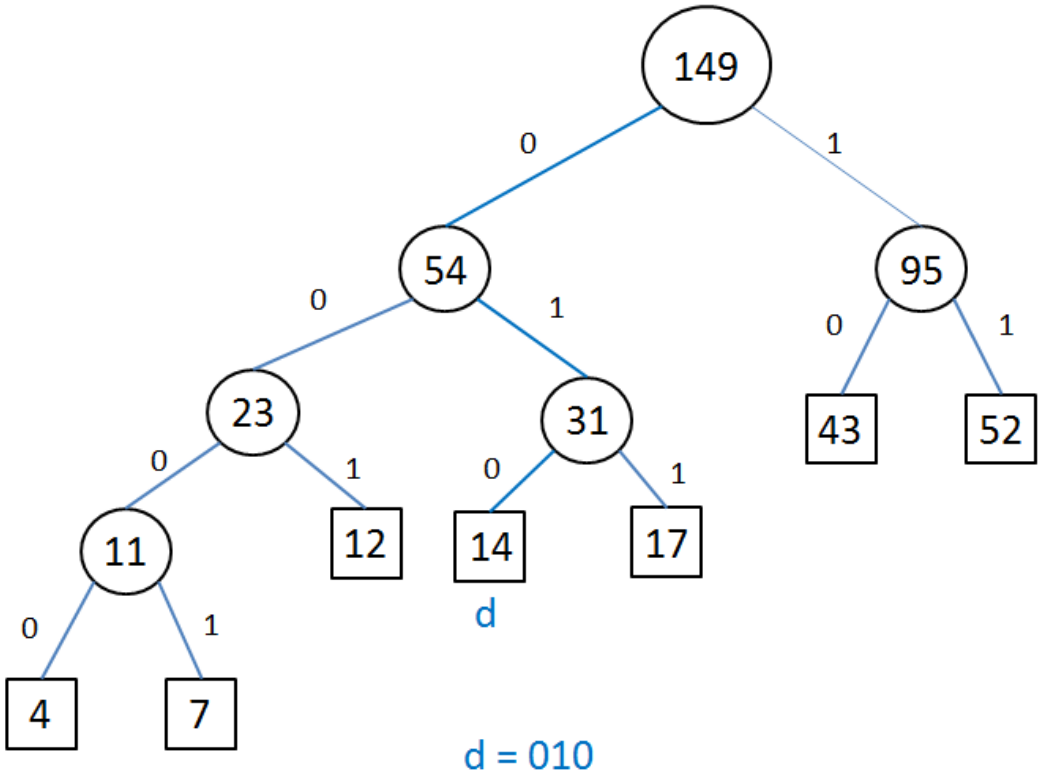

149
0
1
54
95
0
1
0
1
23
31
43
52
0
1
0
1
11
12
14
17
0
1
4
7


149
0
1
54
95
0
1
0
1
23
31
43
52
0
1
0
1
11
12
14
17
0
1
4
7
a
a = 0000

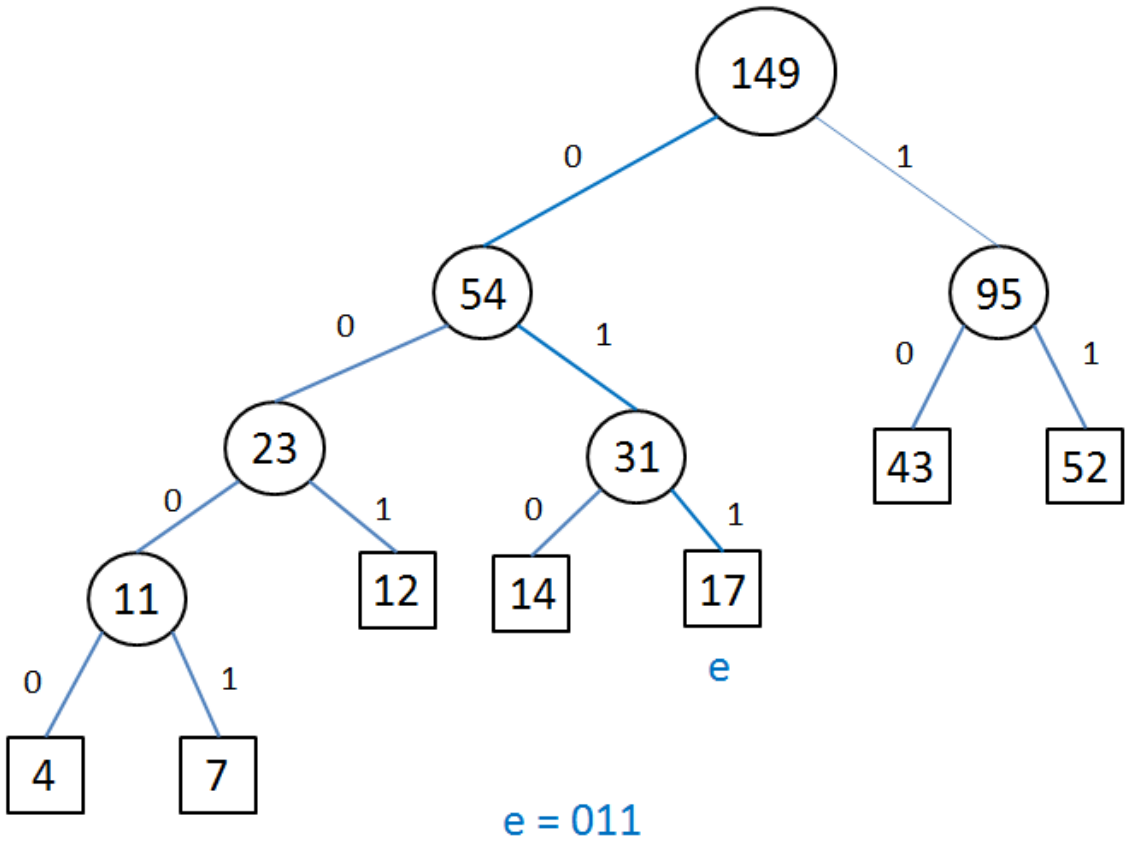

149
0
1
54
95
0
1
0
1
23
31
43
52
0
1
0
1
11
12
14
17
0
1
4
7
b
b = 0001

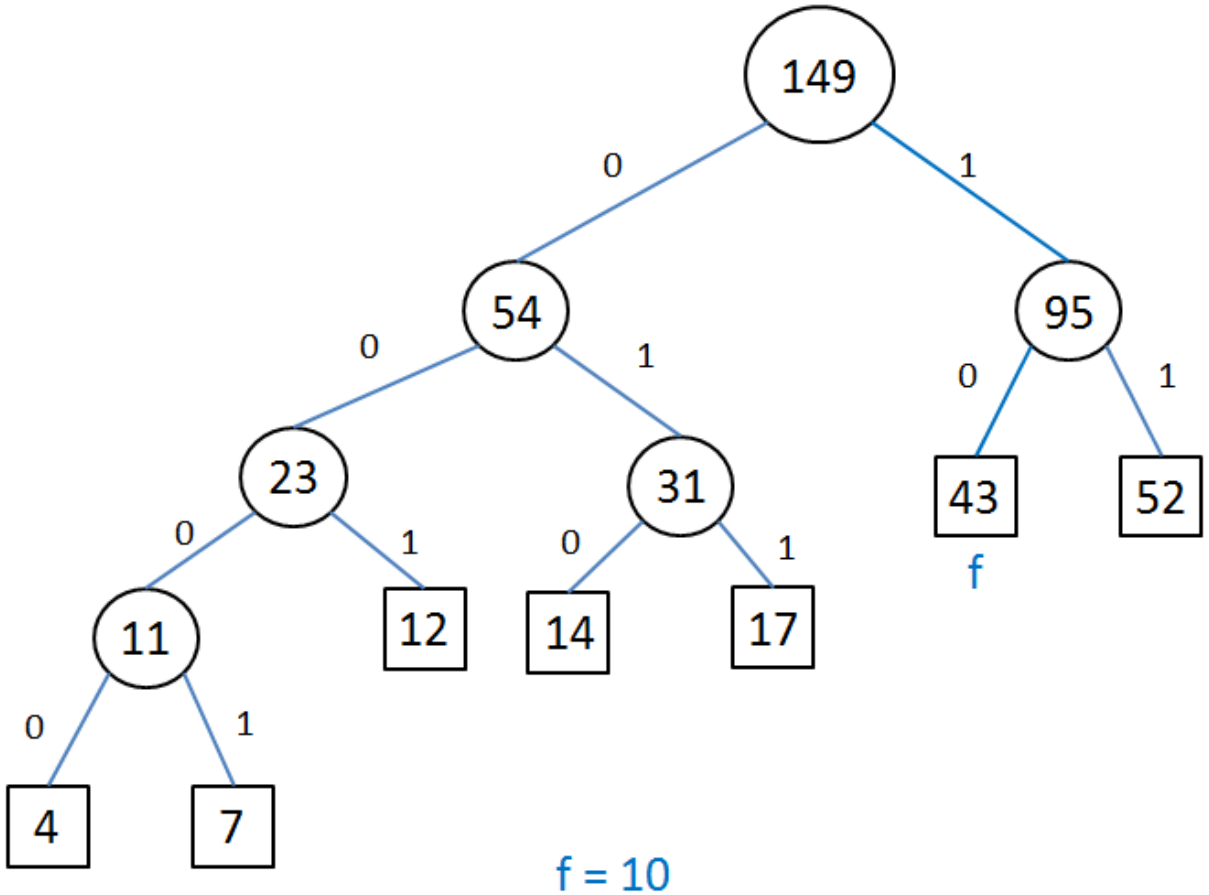


c = 001

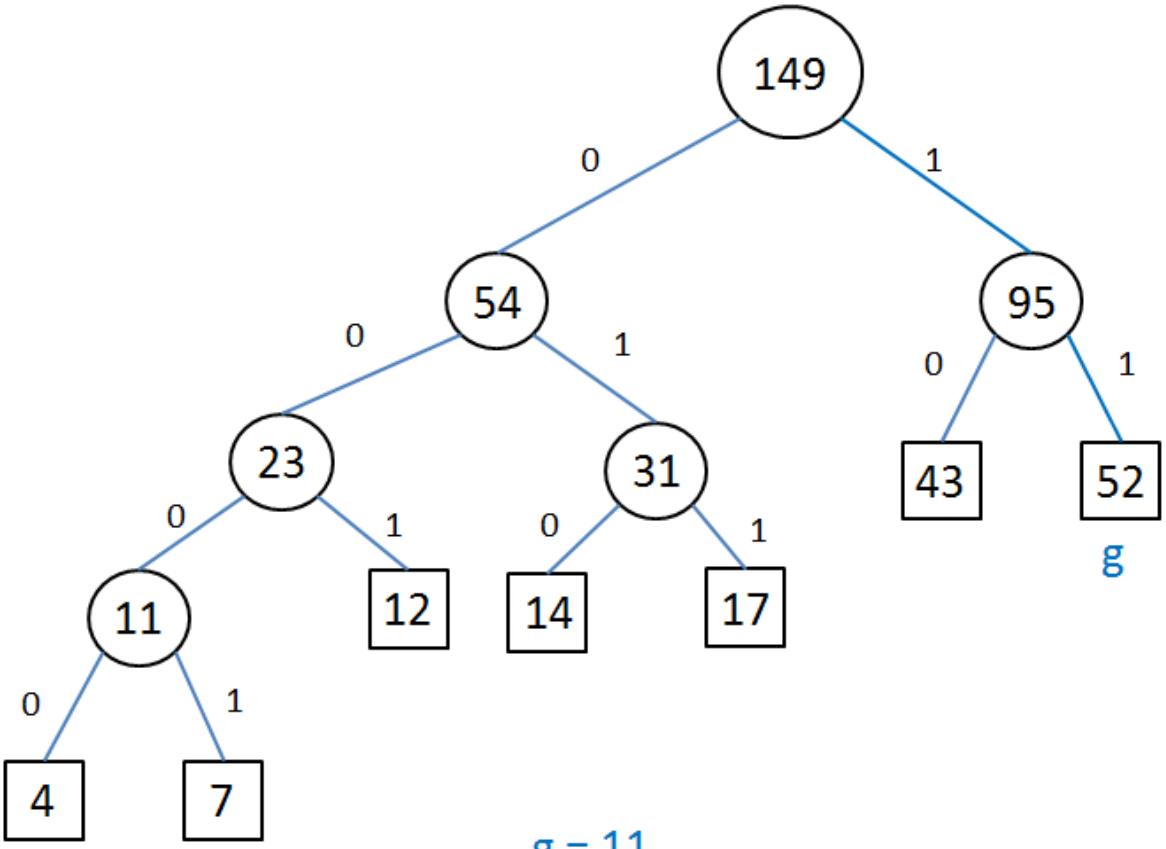


d = 010

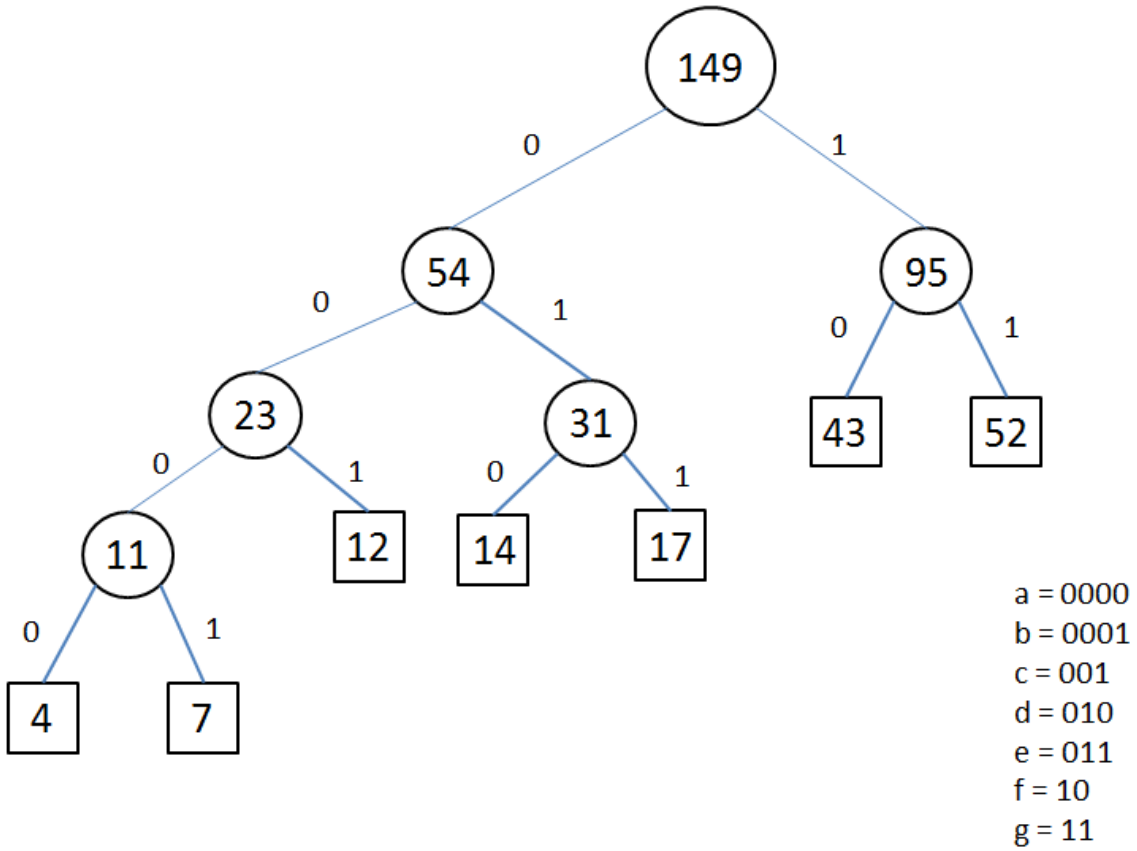

149
0
1
54
95
0
1
0
1
23
31
43
52
0
1
0
1
11
12
14
17
e
0
1
4
7
e = 011



f = 10



g = 11



Implementation

```
def printNodes(node, val=''):
        newVal = val + str(node.huff)
        if(node.left):
                printNodes(node.left, newVal)
        if(node.right):
                printNodes(node.right, newVal)
        if(not node.left and not node.right):
                print(f"{node.symbol} -> {newVal}")
```

- # characters for huffman tree

- chars = ['a', 'b', 'c', 'd', 'e', 'f', 'g']
- # frequency of characters
- freq = [4, 7, 12, 14, 17, 43, 54]
- # list containing unused nodes
- nodes = []
- # converting characters and frequencies into huffman tree nodes
- for x in range(len(chars)):
- nodes.append(node(freq[x], chars[x]))
- while len(nodes) > 1:
- # sort all the nodes in ascending order based on their frequency
- nodes = sorted(nodes, key=lambda x: x.freq)
- # pick 2 smallest nodes
- left = nodes[0]
- right = nodes[1]
- # assign directional value to these nodes
- left.huff = 0
- right.huff = 1
- # combine the 2 smallest nodes to create new node as their parent
- newNode = node(left.freq+right.freq, left.symbol+right.symbol, left, right)
- # remove the 2 nodes and add their parent as new node among others
- nodes.remove(left)
- nodes.remove(right)

- nodes.append(newNode)
- # Huffman Tree is ready!
- printNodes(nodes[0])

Output

a -> 0000 b -> 0001 c -> 001

d -> 010

e -> 011

f -> 10

g -> 11

3. Write a program to solve a fractional Knapsack problem using a greedy method.

Theory

What is the Knapsack Problem?

A thief went to a store to steal some items. There are multiple items available of different weights & profits.

Let suppose there are 'n' No. of items & weight of these are W1, W2,……Wn respectively, and the profit of these items are P1, P2,…….Pn respectively.

The thief wants to do steal in such a way so that his overall profit be 'Maximum ' and 'Capacity constraint' of knapsack don't break.

- Knapsack Problem may be of 2 types:
- [A] 0/1 Knapsack problem
- [B] Fractional Knapsack problem
- 0/1 Knapsack problem
-
- In this problem, either a whole item is selected(1) or the whole item not to be selected(0).
-
- Here, the thief can't carry a fraction of the item.
-
- In the LPP(Linear programming problem) form, it can be described as:

```
1   Algorithm GreedyKnapsack(m, n)
2   // p[1 : n] and w[1 : n] contain the profits and weights respectively
3   // of the n objects ordered such that p[i]/w[i] ≥ p[i + 1]/w[i + 1].
4   // m is the knapsack size and x[1 : n] is the solution vector.
5   {
6       for i := 1 to n do x[i] := 0.0; // Initialize x.
7       U := m;
8       for i := 1 to n do
9       {
10          if (w[i] > U) then break;
11          x[i] := 1.0; U := U − w[i];
12      }
13      if (i ≤ n) then x[i] := U/w[i];
14  }
```

Implementation

- def fractional_knapsack(value, weight, capacity):
- # index = [0, 1, 2, ..., n - 1] for n items

```
    index = list(range(len(value)))
    # contains ratios of values to weight
    ratio = [v/w for v, w in zip(value, weight)]
    # index is sorted according to value-to-weight ratio in decreasing order
    index.sort(key=lambda i: ratio[i], reverse=True)
     max_value = 0
    fractions = [0]*len(value)
for i in index:
        if weight[i] <= capacity:
            fractions[i] = 1
            max_value += value[i]
            capacity -= weight[i]
        else:
            fractions[i] = capacity/weight[i]
            max_value += value[i]*capacity/weight[i]
            break
     return max_value, fractions
n = int(input('Enter number of items: '))
value = input('Enter the values of the {} item(s) in order: '.format(n)).split()
value = [int(v) for v in value]
weight = input('Enter the positive weights of the {} item(s) in order:
'.format(n)).split()
weight = [int(w) for w in weight]
```

- capacity = int(input('Enter maximum weight: '))
-
- max_value, fractions = fractional_knapsack(value, weight, capacity)
- print('The maximum value of items that can be carried:', max_value)
- print('The fractions in which the items should be taken:', fractions)

Output

Enter number of items: 3

Enter the values of the 3 item(s) in order: 24 15 25

Enter the positive weights of the 3 item(s) in order: 15 10 18 Enter maximum weight: 20

The maximum value of items that can be carried: 31.5

The fractions in which the items should be taken: [1, 0.5, 0]

Write a Program for Queens matrix having first Queen placed. Use backtracking to place remaining Queens to generate the final 8-queen's matrix.

- Backtracking Algorithm
- A backtracking algorithm is a problem-solving algorithm that uses a brute force approach for finding the desired output. The Brute force approach tries out all the possible solutions and chooses the desired/best solutions. The term backtracking suggests that if the current solution is not suitable, then backtrack and try other solutions.
  The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes, then we backtrack and return false.

Implementation

1) Start in the leftmost column
2) 2) If all queens are placed return true
3) 3) Try all rows in the current column. Do following for every tried row.
4) a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing queen here leads to a solution.
5) b) If placing the queen in [row, column] leads to a solution then return true.
6) c) If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and go to step (a) to try other rows.
7) 4) If all rows have been tried and nothing worked, return false to trigger backtracking.

```
/* C++ program to solve N Queen Problem using

backtracking */


#include <bits/stdc++.h>

#define N 4

using namespace std;


/* A utility function to print solution */

void printSolution(int board[N][N])

{

    for (int i = 0; i < N; i++) {

            for (int j = 0; j < N; j++)
```

```
                cout << " " << board[i][j] << " ";
            printf("\n");
    }
}

/* A utility function to check if a queen can
be placed on board[row][col]. Note that this
function is called when "col" queens are
already placed in columns from 0 to col -1.
So we need to check only left side for
attacking queens */
bool isSafe(int board[N][N], int row, int col)
{
    int i, j;

    /* Check this row on left side */
    for (i = 0; i < col; i++)
            if (board[row][i])
                    return false;

    /* Check upper diagonal on left side */
    for (i = row, j = col; i >= 0 && j >= 0; i--, j--)
            if (board[i][j])
```

```
                return false;
    /* Check lower diagonal on left side */
    for (i = row, j = col; j >= 0 && i < N; i++, j--)
        if (board[i][j])
                return false;

    return true;
}

/* A recursive utility function to solve N
Queen problem */
bool solveNQUtil(int board[N][N], int col)
{
    /* base case: If all queens are placed
    then return true */
    if (col >= N)
        return true;

    /* Consider this column and try placing
    this queen in all rows one by one */
    for (int i = 0; i < N; i++) {
        /* Check if the queen can be placed on
        board[i][col] */
```

```
            if (isSafe(board, i, col)) {
                /* Place this queen in board[i][col] */
                board[i][col] = 1;

                /* recur to place rest of the queens */
                if (solveNQUtil(board, col + 1))
                    return true;

                /* If placing queen in board[i][col]
                doesn't lead to a solution, then
                remove queen from board[i][col] */
                board[i][col] = 0; // BACKTRACK
            }
        }

        /* If the queen cannot be placed in any row in
            this column col then return false */
        return false;
    }

    /* This function solves the N Queen problem using
    Backtracking. It mainly uses solveNQUtil() to
    solve the problem. It returns false if queens
```

```
cannot be placed, otherwise, return true and
prints placement of queens in the form of 1s.
Please note that there may be more than one
solutions, this function prints one of the
feasible solutions.*/
bool solveNQ()
{
    int board[N][N] = { { 0, 0, 0, 0 },
                                        { 0, 0, 0, 0 },
                                        { 0, 0, 0, 0 },
                                        { 0, 0, 0, 0 } };

    if (solveNQUtil(board, 0) == false) {
            cout << "Solution does not exist";
            return false;
    }

    printSolution(board);
    return true;
}

// driver program to test above function
int main()
```

```
{
    solveNQ();
    return 0;
}
```

Output

- 0 0 1 0
- 1 0 0 0
- 0 0 0 1
- 0 1 0 0

Mini projects

- Write a program to implement matrix multiplication. Also implement multithreaded matrix multiplication with either one thread per row or one thread per cell. Analyze and compare their performance.
- Implement merge sort and multithreaded merge sort. Compare time required by both the algorithms. Also analyze the performance of each algorithm for the best case and the worst case.
- Implement the Naive string matching algorithm and Rabin-Karp algorithm for string matching. Observe difference in working of both the algorithms for the same input.

Sinhgad Technical Education Society's
SMT. KASHIBAI NAVALE COLLEGE OF ENGINEERING VADGAON (BK), PUNE-411041

Class: BE
Assignment: LP3

Sr NO.	Name
Group A: Design and Analysis of Algorithms	
1.	Write a program non-recursive and recursive program to calculate Fibonacci numbers andanalyze their time and space complexity.
2.	Write a program to implement Huffman Encoding using a greedy strategy.
3.	Write a program to solve a fractional Knapsack problem using a greedy method.
4.	Write a program for analysis of quick sort by using deterministic and randomized variant.
Group B: Machine Learning	
1.	Predict the price of the Uber ride from a given pickup point to the agreed drop-off location.Perform following tasks: 1. Pre-process the dataset. 2. Identify outliers. 3. Check the correlation. 4. Implement linear regression and random forest regression models. Evaluate the models and compare their respective scores like R2, RMSE, etc. Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset
2.	Classify the email using the binary classification method. Email Spam detection has twostates: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance. Dataset link: The emails.csv dataset on the Kaggle https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv
3.	Given a bank customer, build a neural network-based classifier that can determine whetherthey will leave or not in the next 6 months. Dataset Description: The case study is from an open-source dataset from Kaggle.The dataset contains 10,000 sample points with 14 distinct features such as CustomerId, CreditScore, Geography, Gender, Age, Tenure, Balance, etc. Link to the Kaggle project: https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling Perform following steps: 1. Read the dataset. 2. Distinguish the feature and target set and divide the data set into training and test sets. 3. Normalize the train and test data. 4. Initialize and build the model. Identify the points of improvement and implement the same. Print the accuracy score and confusion matrix (5 points).

4.	Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.
5.	Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusionmatrix, accuracy, error rate, precision and recall on the given dataset. Dataset link : https://www.kaggle.com/datasets/abdallamahgoub/diabetes
6.	Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset.Determine the number of clusters using the elbow method. Dataset link : https://www.kaggle.com/datasets/kyanyoga/sample-sales-data
Group C: Blockchain Technology	
1.	Installation of MetaMask and study spending Ether per transaction.
2.	Create your own wallet using Metamask for crypto transactions.
4.	Write a program in solidity to create Student data. Use the following constructs: • Structures • Arrays • Fallback Deploy this as smart contract on Ethereum and Observe the transaction fee and Gas values.
5.	Write a survey report on types of Blockchains and its real time use cases.

Subject Teacher: Prof. Swati B. Dhadake

HOD
Computer Department

Group B: Machine Learning:

Assignment No. 1

Aim: Predict the price of the Uber ride from a given pickup point to the agreed drop-off location.

Perform following tasks:

5. Pre-process the dataset.
6. Identify outliers.
7. Check the correlation.
8. Implement linear regression and random forest regression models.

Evaluate the models and compare their respective scores like R2, RMSE, etc.Dataset link:

https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

Objective:

1. Implement linear regression and random forest regression models.
2. Evaluate the models and compare their respective scores like R2, RMSE

Theory:

Introduction:

Regression analysis is a very widely used statistical tool to establish a relationship model between two variables. One of these variable is called predictor variable whose value is gathered through experiments. The other variable is called response variable whose value is derived from the predictor variable.

Linear models are the simplest parametric methods and always deserve the right attention, because many problems, even intrinsically non-linear ones, can be easily solved with these models. A regression is a prediction where the target is continuous and its applications are several, so it's important to understand how a linear model can fit the data, what its strengths and weaknesses are, and when it's preferable to pick an alternative.

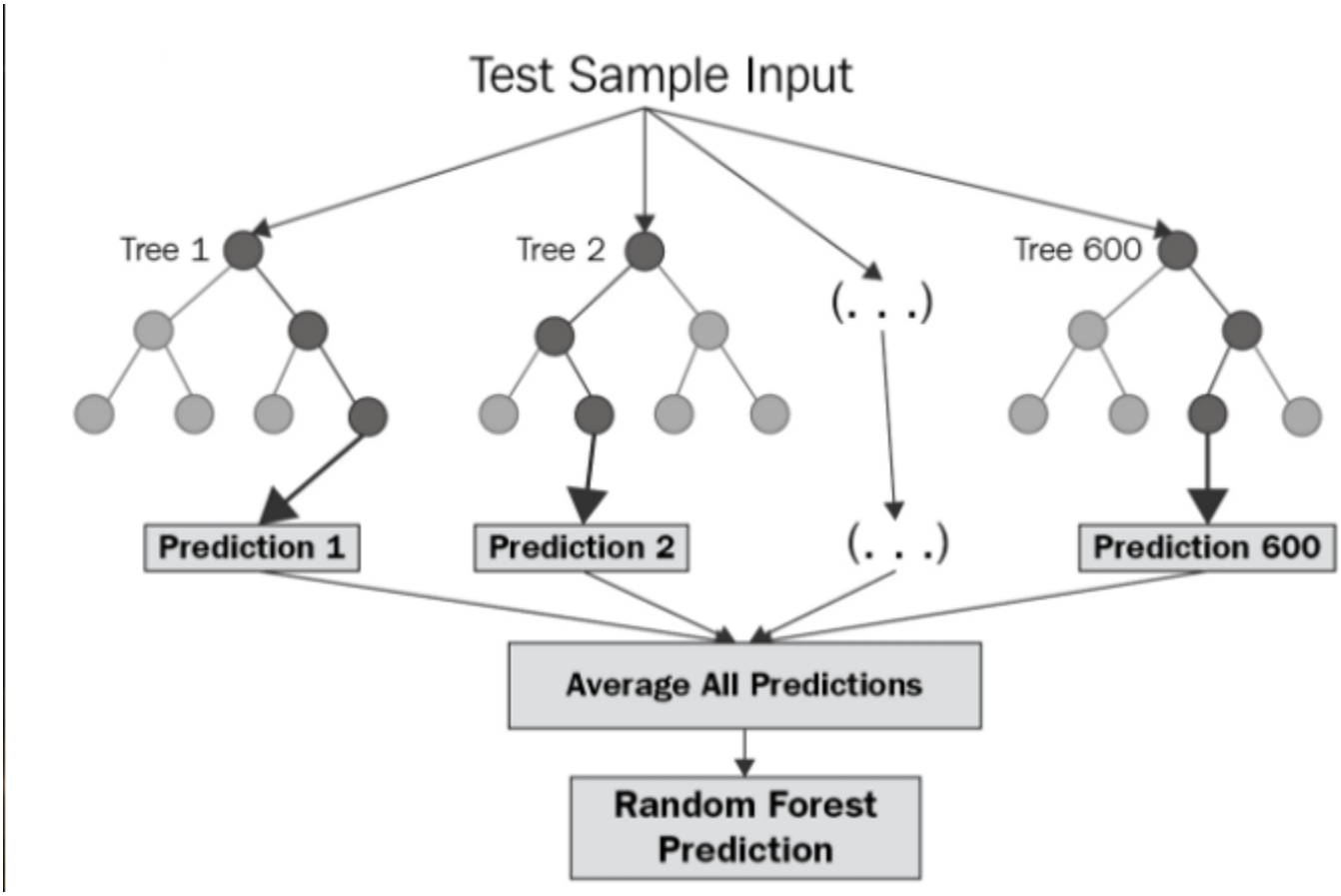
Types of Regression:

- Linear

- Multiple Linear
- Logistic
- Polynomial

Random Forest Regression:

Random Forest Regression is a supervised learning algorithm that uses **ensemble learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.



The diagram above shows the structure of a Random Forest. You can notice that the trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees. To get a better understanding of the Random Forest algorithm, let's walk through the steps:

1. Pick at random *k* data points from the training set.
2. Build a decision tree associated to these *k* data points.
3. Choose the number *N* of trees you want to build and repeat steps 1 and 2.
4. For a new data point, make each one of your *N*-tree trees predict the value of *y* for the data point in question and assign the new data point to the average across all of the predicted *y* values.

A Random Forest Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationships. Disadvantages, however, include the following: there is no interpretability, overfitting may easily occur, we must choose the number of trees to include in the model.

Preprocess Data in Python Step-by-Step:

1. Load data in Pandas.
2. Drop columns that aren't useful.
3. Drop rows with missing values.
4. Create dummy variables.
5. Take care of missing data.
6. Convert the data frame to NumPy.
7. Divide the data set into training data and test data.

Identify outliers:

When exploring data, the outliers are the extreme values within the dataset. That means the outlier data points vary greatly from the expected values—either being much larger or significantly smaller. For data that follows a normal distribution, the values that fall more than three standard deviations from the mean are typically considered outliers.

Outliers can find their way into a dataset naturally through variability, or they can be the result of issues like human error, faulty equipment, or poor sampling. Regardless of how they get into the data, outliers can have a big impact on statistical analysis and machine learning because they impact calculations like mean and standard deviation, and they can skew hypothesis tests. A data analyst should use various techniques to visualize and identify outliers before deciding whether they should be dropped, kept, or modified.

Review this article to learn more about the different types of outliers:

Loading and describing example data

The examples throughout this article use the Uber Fares Dataset available on Kaggle.com. Download the CSV to follow along. It has nine columns and 200k rows. These are the fields we will use:

- **key**—a unique identifier for each trip
- **fare_amount**—the cost of each trip in usd
- **pickup_datetime**—date and time when the meter was engaged
- **passenger_count**—the number of passengers in the vehicle (driver entered value)

Check the correlation:

A correlation could be positive, meaning both variables move in the same direction, or negative, meaning that when one variable's value increases, the other variables' values decrease. Correlation can also be neutral or zero, meaning that the variables are unrelated.

- **Positive Correlation**: both variables change in the same direction.
- **Neutral Correlation**: No relationship in the change of the variables.
- **Negative Correlation**: variables change in opposite directions.

The performance of some algorithms can deteriorate if two or more variables are tightly related, called multicollinearity. An example is linear regression, where one of the offending correlated variables should be removed in order to improve the skill of the model.

We may also be interested in the correlation between input variables with the output variable in order provide insight into which variables may or may not be relevant as input for developing a model.

The structure of the relationship may be known, e.g. it may be linear, or we may have no idea whether a relationship exists between two variables or what structure it may take. Depending what is known about the relationship and the distribution of the variables, different correlation scores can be calculated.

Conclusion: We have studied the Linear Regression and Random Forest Regression also implemented successfully.

Assignment no. 2

Aim: Implement Gradient Descent Algorithm to find the local minima of a function.
For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.

Objective:

- The Basic Concepts of Gradient Descent algorithm.
- to find the local minima of a function at point x=2

Theory:

Introduction

Gradient Descent is an iterative algorithm that is used to minimize a function by finding the optimal parameters. Gradient Descent can be applied to any dimension function i.e. 1-D, 2-D, 3-D. In this article, we will be working on finding global minima for parabolic function (2-D) and will be implementing gradient descent in python to find the optimal parameters for the linear regression equation (1-D). Before diving into the implementation part, let us make sure the set of parameters required to implement the gradient descent algorithm. To implement a gradient descent algorithm, we require a cost function that needs to be minimized, the number of iterations, a learning rate to determine the step size at each iteration while moving towards the minimum, partial derivates for weight & bias to update the parameters at each iteration, and a prediction function

Algorithm for Gradient Descent

Steps should be made in proportion to the negative of the function gradient (move away from the gradient) at the current point to find local minima. Gradient Ascent is the procedure for approaching a local maximum of a function by taking steps proportional to the positive of the gradient (moving towards the gradient).

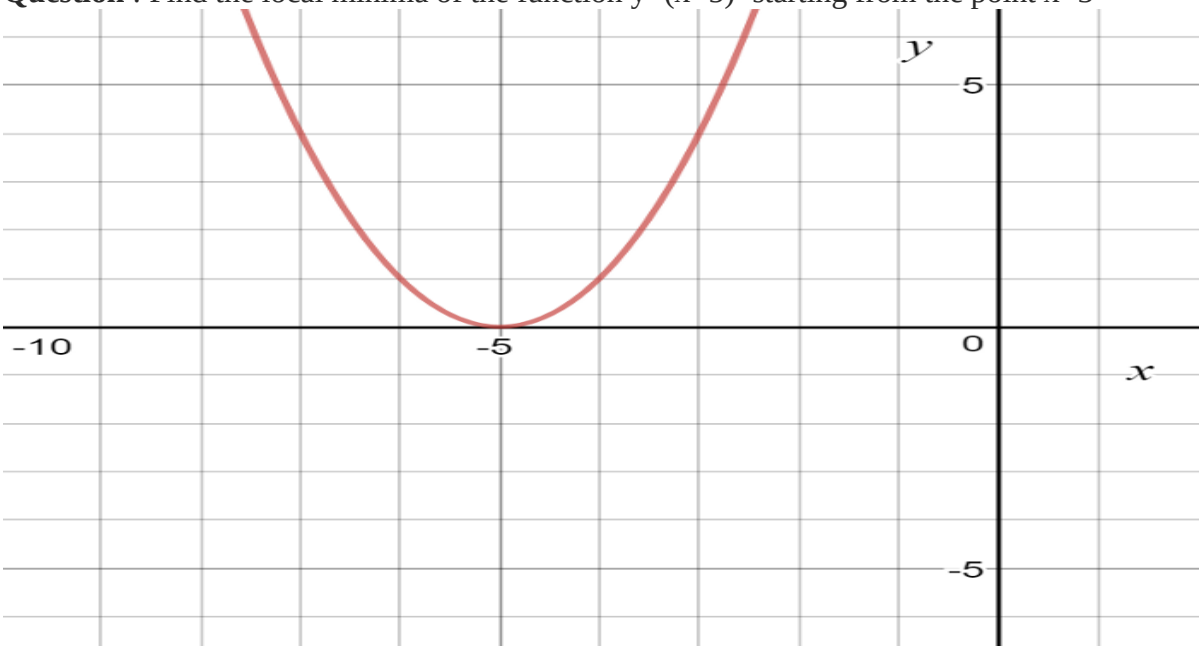
repeat until convergence

{

```
    w = w - (learning_rate * (dJ/dw))
    b = b - (learning_rate * (dJ/db))
}
```

Example by hand :

Question : Find the local minima of the function $y=(x+5)^2$ starting from the point x=3



Solution : We know the answer just by looking at the graph. $y = (x+5)^2$ reaches it's minimum value when x = -5 (i.e when x=-5, y=0). Hence x=-5 is the local and global minima of the function.

Now, let's see how to obtain the same numerically using gradient descent.

Step 1 : Initialize x =3. Then, find the gradient of the function, dy/dx = 2*(x+5).

Step 2 : Move in the direction of the negative of the gradient (Why?). But wait, how much to move? For that, we require a learning rate. Let us assume the **learning rate → 0.01**

Step 3 : Let's perform 2 iterations of gradient descent

Initialize Parameters :

$$X_0 = 3$$

$$Learning\ rate = 0.01$$

$$\frac{dy}{dx} = \frac{d}{dx}(x+5)^2 = 2 * (x+5)$$

Iteration 1 :

$$X_1 = X_0 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_1 = 3 - (0.01) * \big(2 * (3+5)\big) = 2.84$$

Iteration 2 :

$$X_2 = X_1 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_2 = 2.84 - (0.01) * \big(2 * (2.84+5)\big) = 2.6832$$

Step 4 : We can observe that the X value is slowly decreasing and should converge to -5 (the local minima). However, how many iterations should we perform?

Let us set a precision variable in our algorithm which calculates the difference between two consecutive "x" values . If the difference between x values from 2 consecutive iterations is lesser than the precision we set, stop the algorithm !

Gradient descent in Python :

Step 1 : Initialize parameters

```
cur_x = 3 # The algorithm starts at x=3
rate = 0.01 # Learning rate
precision = 0.000001 #This tells us when to stop the algorithm
previous_step_size = 1 #
max_iters = 10000 # maximum number of iterations
iters = 0 #iteration counter
df = lambda x: 2*(x+5) #Gradient of our function
```

Step 2 : Run a loop to perform gradient descent :

i. Stop loop when difference between x values from 2 consecutive iterations is less than 0.000001 or when number of iterations exceeds 10,000

```
while previous_step_size > precision and iters < max_iters:
    prev_x = cur_x #Store current x value in prev_x
    cur_x = cur_x - rate * df(prev_x) #Grad descent
    previous_step_size = abs(cur_x - prev_x) #Change in x
    iters = iters+1 #iteration count
    print("Iteration",iters,"\nX value is",cur_x) #Print iterations

print("The local minimum occurs at", cur_x)
```

Output : From the output below, we can observe the x values for the first 10 iterations- which can be cross checked with our calculation above. The algorithm runs for 595 iterations before it terminates. The code and solution is embedded below for reference.

Conclusion: We have studied Gradient Descent Algorithm to find the local minima of a function also implemented successfully.

Assignment no. 3

Aim: Implement K-Nearest Neighbors algorithm on diabetes.csv dataset

Objective:

- The Basic Concepts of k-NN algorithm.
- Compute confusionmatrix, accuracy, error rate, precision and recall on the given dataset.

Theory:

Introduction

KNN is a non-parametric, lazy learning algorithm. Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.

When we say a technique is non-parametric, it means that it does not make any assumptions on the underlying

data distribution. In other words, the model structure is determined from the data. Therefore, KNN could and probably should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution data.

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.

There are other ways of calculating distance, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

KNN Algorithm

We can implement a KNN model by following the below steps:

- □ Load the data
- □ Initialize the value of k
- □ For getting the predicted class, iterate from 1 to total number of training data points
    - Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it‘s the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
    - Sort the calculated distances in ascending order based on distance values
    - Get top k rows from the sorted array
    - Get the most frequent class of these rows
    - Return the predicted class

Choosing the right value for K

To select the K that‘s right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm‘s ability to accurately make predictions when it‘s given data it hasn‘t seen before.

Here are some things to keep in mind:

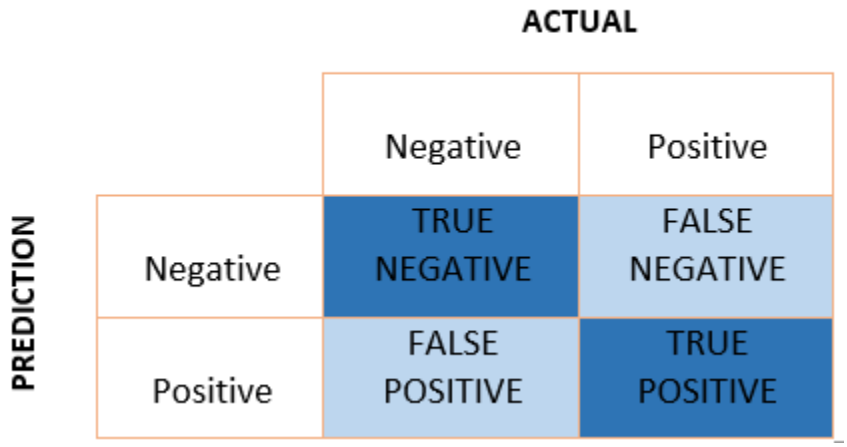
- As we decrease the value of K to 1, our predictions become less stable. Just think for a minute, image K=1 and we have a query point surrounded by several reds and one green (I‘m thinking aboutthe top left corner of the colored plot above), but the green is the single nearest neighbor. Reasonably, we would think the query point is most likely red, but because K=1, KNN incorrectly predicts that the query point is green.
- □ Inversely, as we increase the value of K, our predictions become more stable due to majority voting / averaging, and thus, more likely to make more accurate predictions (up to a certain point).

Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far.

- In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.

Confusion matrix:

It is a matrix of size 2×2 for binary classification with actual values on one axis and predicted on another.

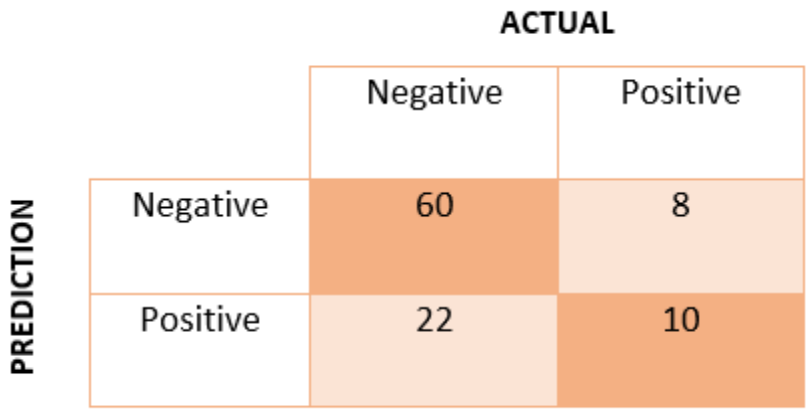


Confusion Matrix

Let's understand the confusing terms in the confusion matrix: **true positive, true negative, false negative, and false positive** with an example.

EXAMPLE

A machine learning model is trained to predict tumor in patients. The test dataset consists of 100 people.



Confusion Matrix for tumor detection

True Positive (TP) — model correctly predicts the positive class (prediction and actual both are positive). In the above example, **10 people** who have tumors are predicted positively by the model.

True Negative (TN) — model correctly predicts the negative class (prediction and actual both are negative). In the above example, **60 people** who don't have tumors are predicted negatively by the model.

False Positive (FP) — model gives the wrong prediction of the negative class (predicted-positive, actual-negative). In the above example, **22 people** are predicted as positive of having a tumor, although they don't have a tumor. FP is also called a **TYPE I** error.

False Negative (FN) — model wrongly predicts the positive class (predicted-negative, actual-positive). In the above example, **8 people** who have tumors are predicted as negative. FN is also called a **TYPE II** error.

With the help of these four values, we can calculate True Positive Rate (TPR), False Negative Rate (FPR), True Negative Rate (TNR), and False Negative Rate (FNR).

$$TPR = \frac{TP}{Actual\ Positive} = \frac{TP}{TP + FN}$$

$$FNR = \frac{FN}{Actual\ Positive} = \frac{FN}{TP + FN}$$

$$TNR = \frac{TN}{Actual\ Negative} = \frac{TN}{TN + FP}$$

$$FPR = \frac{FP}{Actual\ Negative} = \frac{FP}{TN + FP}$$

Even if data is imbalanced, we can figure out that our model is working well or not. For that, **the values of TPR and TNR should be high, and FPR and FNR should be as low as possible.**

With the help of TP, TN, FN, and FP, other performance metrics can be calculated.

Precision, Recall :

While searching something on the web, the model does not care about something **irrelevant** and **not retrieved** (this is the true negative case). Therefore only TP, FP, FN are used in Precision and Recall.

Precision

Out of all the positive predicted, what percentage is truly positive.

$$Precision = \frac{TP}{TP + FP}$$

The precision value lies between 0 and 1.

Recall

Out of the total positive, what percentage are predicted positive. It is the same as TPR (true positive rate).

$$Recall = \frac{TP}{TP + FN}$$

Confusion Matrix for Credit Card Fraud Detection

Advantages

- No assumptions about data—useful, for example, for nonlinear data

- Simple and easy to implement algorithm—to explain and understand/interpret

- High accuracy (relatively)—it is pretty high but not competitive in comparison to better supervised learning models

- Versatile—useful for classification or regression

Disadvantages

- Computationally expensive—because the algorithm stores all of the training data

□ High memory requirement

□ Stores all (or almost all) of the training data

□ Prediction stage might be slow (with big N)

□ Sensitive to irrelevant features and the scale of the data

□ The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

Conclusion: We have studied the k-NN Classification algorithm and also implemented successfully.

Assignment No. 4

Aim: Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset.Determine the number of clusters using the elbow method.

Dataset link : https://www.kaggle.com/datasets/kyanyoga/sample-sales-data

Objective:

- The Basic Concepts of K-means algorithm.
- Implementation logic of K-means algorithm.

Theory:

Introduction:

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.
Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labeled, outcomes.

A cluster refers to a collection of data points aggregated together because of certain similarities.
You'll define a target number k, which refers to the number of centroids you need in the dataset. A centroidis the imaginary or real location representing the center of the cluster.

Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.
In other words, the K-means algorithm identifies k number of centroids, and then allocates every data pointto the nearest cluster, while keeping the centroids as small as possible.

The _means' in the K-means refers to averaging of the data; that is, finding the centroid.
To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids

It halts creating and optimizing clusters when either:
The centroids have stabilized—there is no change in their values because the clustering has been successful.

The defined number of iterations has been achieved.

The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori.

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into *k* clusters so that the resulting intra-cluster similarity is high but the inter-cluster similarity is low.

Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster‘s centroid or center of gravity.

The k-means algorithm proceed as follows

First, it randomly selects k of the objects, each of which initially represents a cluster mean or center.

For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean.

It then computes the new mean for each cluster. This process iterates until the criterion function converges.

Typically, the square-error criterion is used, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2,$$

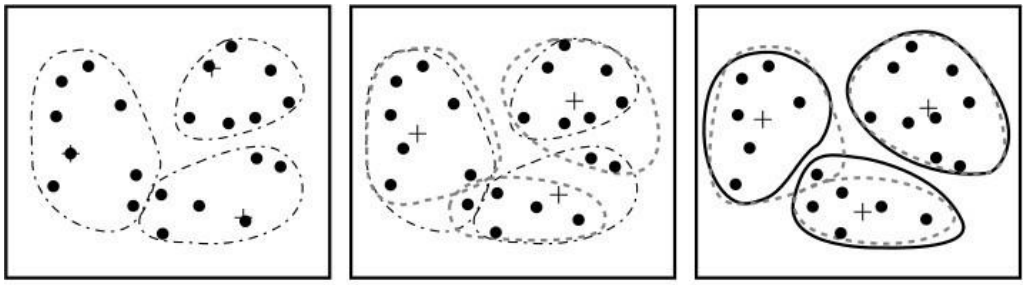
where *E* is the sum of the square error for all objects in the data set;

p is the point in space representing a given object;

mi is the mean of cluster *Ci* (both p and mi are multidimensional).

In other words, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This criterion tries to make the resulting k clusters as compact and as separate as possible.

The k-means procedure is summarized in following figure.



(a) (b) (c)

Above figure represent clustering of set of objects based on clustering methods. In figure, mean of each cluster marked by a

K-means Algorithm

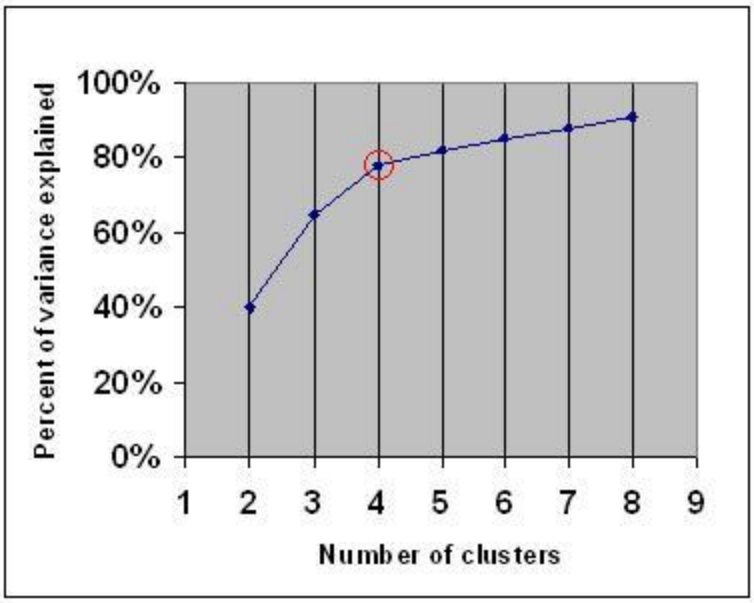
The k-means algorithm for partitioning, where is cluster center is represented by the mean value of the object in the cluster.

Algorithm:

1. Arbitrary choose k objects from D as the initial cluster centers
2. Repeat
3. (re)Assign each object to the cluster to which the object is most similar,
   Based on the mean value of the objects in the cluster;
4. Update the cluster means, i.e., calculate the mean value of the objects for each cluster,
5. Until no change

Elbow method:

In cluster analysis, the **elbow method** is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters and picking the elbow of the curve as the number of clusters to use. The same method can be used to choose the number of parameters in other data-driven models, such as the number of principal components to describe a data set.



Advantages

- □ Easy to implement.
- □ An instance can change cluster (move to another cluster) when the centroids are re-computed.
- □ If variables are huge, then K-Means most of the times computationally faster than hierarchical clustering, if we keep k smalls.
- □ K-Means produce tighter clusters than hierarchical clustering, especially if the clusters are globular.

Disadvantages

- Difficult to predict the number of clusters. (K-Value)
- □ Initial seeds have a strong impact on the final results.
- □ The order of the data has an impact on the final results.
- □ Sensitive to scale: rescaling your datasets (normalization or standardization) will completely change results. While this itself is not bad, not realizing that you have to spend extra attention to scaling your data might be bad.

Difficult to predict K-Value.

- □ With global cluster, it didn't work well.

- □ Different initial partitions can result in different final clusters.
- □ It does not work well with clusters (in the original data) of Different size and Different density

Conclusion: We have studied the k-means clustering algorithm and also implemented successfully.

l

GROUP C

Assignment no: 01

AIM : Installation of Metamask and study spending Ether per transaction.

Theory:

What is Meta mask?

The Metamask wallet is basically a crypto wallet that provides support for ETH-based tokens such as ERC-721 and ERC-20 tokens. It is available as a browser plugin that you can install easily, just like any other browser extension. Interestingly, you can enjoy a seamless connection to any Ethereum-based decentralized app after installing the Metamask Chrome extension or Firefox extension. You could easily access any decentralized application like yield farming protocols and NFT marketplaces with the wallet.

With the facility of web browser integration in the form of plugins, you can have convenient experiences in the use of Meta Mask. This is probably one of the foremost reasons for its rapidly increasing rates of adoption. As the demand for a decentralized web starts to gain momentum, Meta Mask can serve as a gateway for you into a new world of exciting opportunities with dApps, web browsing, and DeFi, and block chain technology.

Steps for Setting up Metamask

- **Downloading Metamask**

The first step would obviously start with Metamask download, and you can use the official website for the same. The official website can offer you support for three major operating systems such as Android, iOS, and Windows. You need to download the Metamask Firefox or Chrome extension according to the browser of your choice.

You could also avail the crypto wallet on the desktop or mobile platforms according to your convenience. Interestingly, the Metamask download procedure is nearly similar for all types of supported browsers, thereby offering better ease of setup.

- **Creating Your Wallet**

After you finish installing the Metamask Chrome or Firefox extension, you need to identify the steps for creating your wallet. The splash screen which comes after you have installed Meta Mask will welcome you to the platform. Find out the "Get Started" button and click on it to start the process of creating your Metamask wallet. All you have to do is tap on the "Create a Wallet" button which pops up as soon as you click on "Get Started."

Furthermore, you would find prompt asking users whether they want to improve the platform. If you don't have any interest in the same, then you can click on "No Thanks" or click on "I agree." Now, you could clearly notice the ease and flexibility in answering how to set up Metamask within minutes.

- **Creating Your Password**

The next step in using the Metamask Firefox or Chrome extension you have installed involves creating your password. You need to pick up a strong password which must have a minimum of 8 characters. Make sure that you have used a distinctively unique password that is difficult to guess.

The best practice for creating your password for Meta Mask would refer to the use of symbols, upper and lower case letters, and symbols. Confirm your password once again in the specified field and read the "Terms of Use" by clicking on the link. Once you are done with the "Terms of Use," you can just click on "Create" to set your password.

- **Confirming the Backup Phrase**

Once you have created the password for your Metamask wallet, you will receive a 12-word backup phrase. The backup phrase is quite crucial for accessing your funds in the wallet, and you should note it down exactly as it is displayed on the screen. The backup phrase proves helpful for recovering your crypto wallet if you cannot access your computer.

Therefore, you need to store your backup phrase in a safe location. You can find your answers for 'is Metamask safe' with the help of the backup phrase. Do not expose your backup phrase to anyone and keep the 12-word phrase a secret for safeguarding your assets.

After writing down the secret backup phrase, you need to click on the "Next" button. You have to confirm the backup phrase in the next screen. All you have to do is enter the backup phrase in the exact same order as seen on the previous screen. Once you have entered the phrase, you can click on "Confirm" to complete the process

- **Final Steps**

Confirming the secret backup phrase is the penultimate step in how to set up Metamask. You have to click on "All Done" on the final page. Subsequently, users would be logged in automatically to Metamask. You can log in again by clicking on the icon added for the browser extension, generally found near the URL bar.

Using Metamask

Now that you have set up MetaMask, you need to find out how to use it. As a matter of fact, you wouldn't find any complicated answers for how to use Metamask. You can use the wallet with basic functionalities such as the "Assets" tab, which helps you find a list of your assets. In addition, the "Activity" tab can help you take a look at your transaction history.

- **Sending Transactions**

Another interesting highlight about the Metamask wallet is that it simplifies transactions. You can enter the recipient address and the amount you want to send alongside a transaction fee and click on "Send" to send transactions. Users can also leverage information from ETH Gas Station or related platforms for manual adjustment of the transaction fee. Once you click on "Next," you could have the flexibility of confirming or rejecting the transaction on the following page.

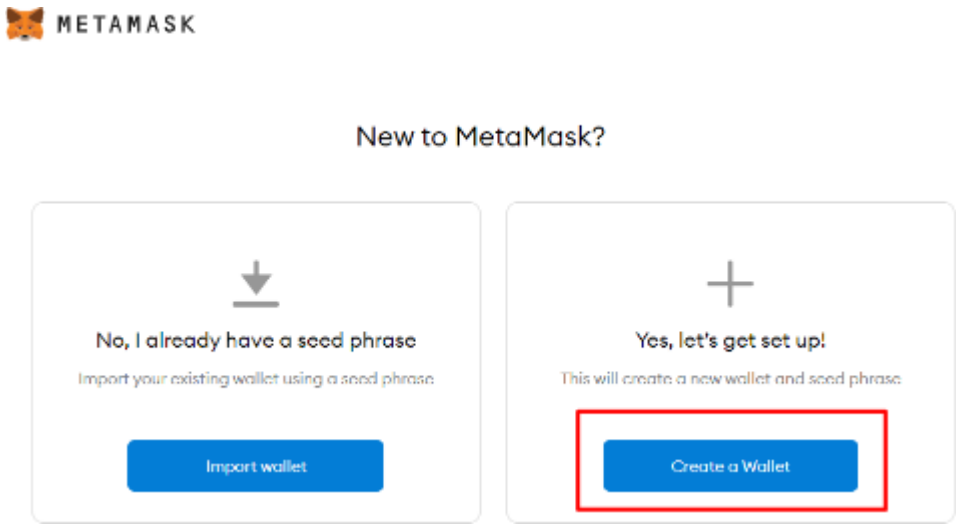
- **Connecting with dApps**

The next important and most notable application of MetaMask focuses on connecting with dApps or smart contracts. You can click on the "Connect to Wallet" button or a similar option on the decentralized app platform you want to use. Once you have clicked on the button, you will discover a prompt asking you for permission to connect the dApp to your wallet. When you connect with a decentralized application, it can view your public addresses. However, they could not access your funds. Interestingly, dApps connect automatically to the Metamask wallet, thereby ensuring a simplified connection process.

Step 1: Go to Chrome Web Store Extensions Section.
Step 2: Search *MetaMask.*
Step 3: Check the number of downloads to make sure that the legitimate MetaMask is being installed, as hackers might try to make clones of it.

Step 4: Click the *Add to Chrome* button.
Step 5: Once installation is complete this page will be displayed. Click on the *Get Started* button.

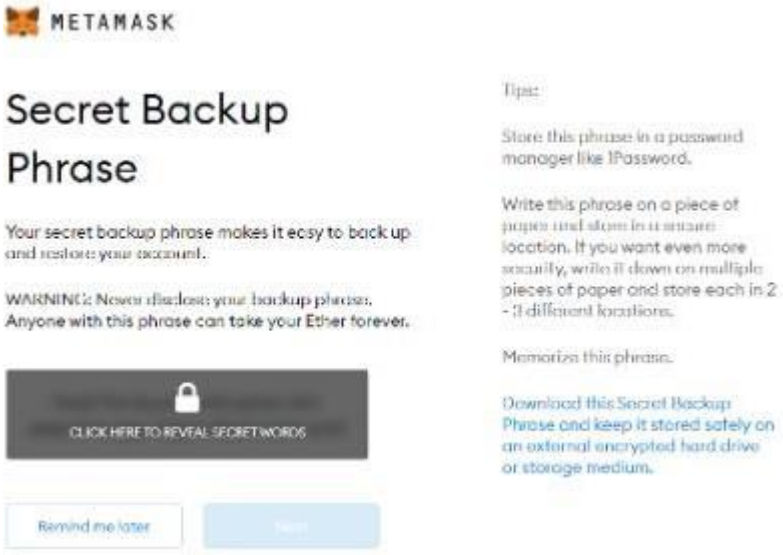
Step 6: This is the first time creating a wallet, so click the *Create a Wallet* button. If there is already a wallet then import the already created using the *Import Wallet* button.



Step 7: Click *I Agree* button to allow data to be collected to help improve MetaMask or else click the *No Thanks* button. The wallet can still be created even if the user will click on the *No Thanks* button.

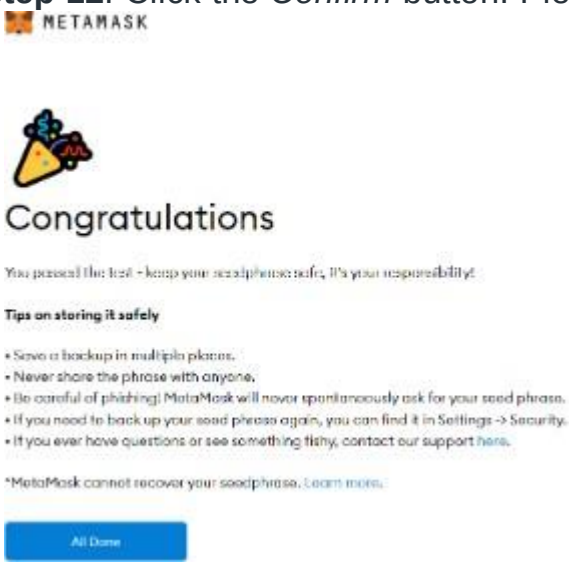
Step 8: Create a password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask. A new password needs to be created if chrome is uninstalled or if there is a switching of browsers. In that case, go through the *Import Wallet* button. This is because MetaMask stores the keys in the browser. Agree to *Terms of Use*.

Step 9: Click on the dark area which says *Click here to reveal secret words* to get your secret phrase.
Step 10: This is the most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer. Please read everything on this screen until you understand it completely before proceeding. The secret phrase is the only way to access your wallet if you forget your password. Once done click the *Next* button.

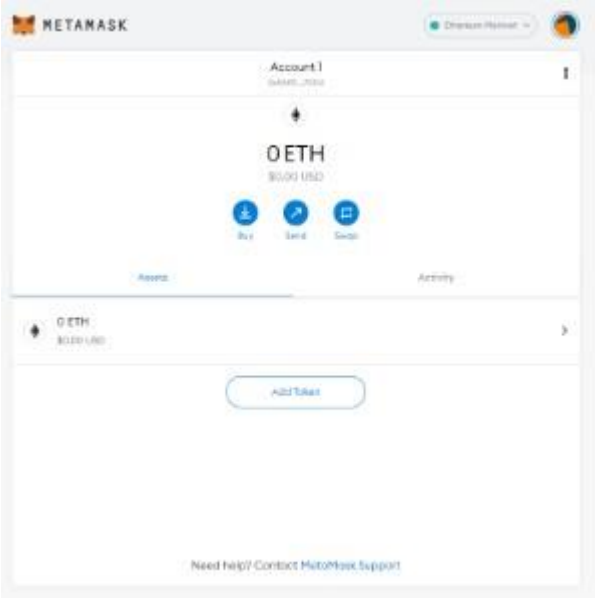


Step 11: Click the buttons respective to the order of the words in your seed phrase. In other words, type the seed phrase using the button on the screen. If done correctly the *Confirm* button should turn blue.

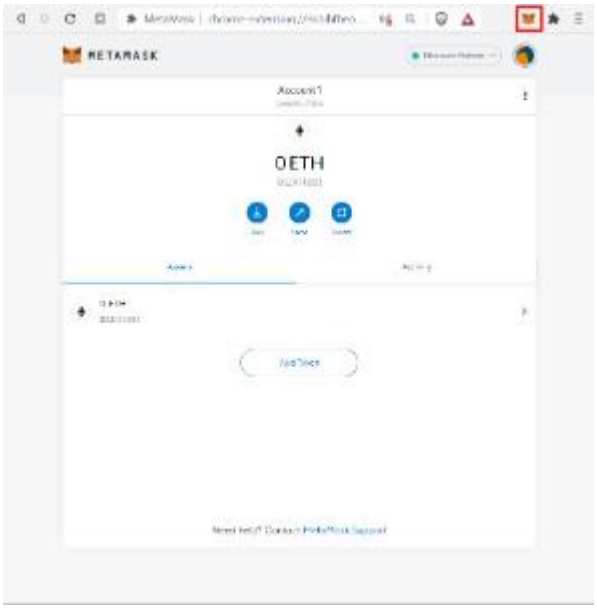
Step 12: Click the *Confirm* button. Please follow the tips mentioned.



Step 13: One can see the balance and copy the address of the account by clicking on the *Account 1* area.



Step 14: One can access MetaMask in the browser by clicking the Foxface icon on the top right. If the Foxface icon is not visible, then click on the puzzle piece icon right next to it.



Conclusion:

Assignment no: 2

Title: Create your own wallet using Meta mask for crypto transactions.

Theory:

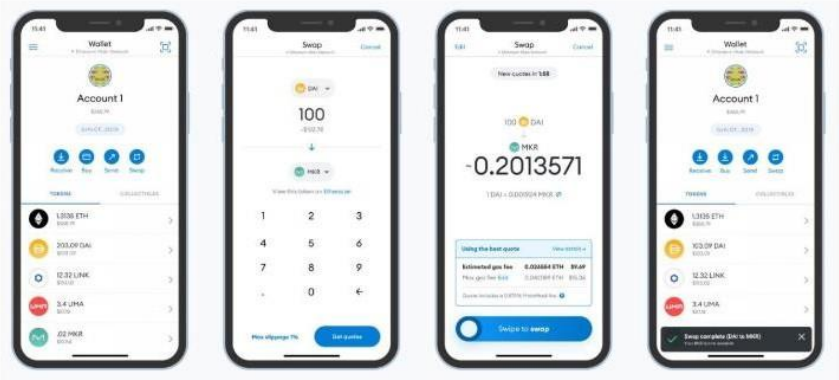
How does the Meta Mask wallet function?

MetaMask crypto currency wallet employs the web3.js library to function. This library is a part of the official Ethereum product. The library was developed focusing on the requirements of web applications that could interact with the Ethereum block chain and take advantage of all block chain's benefits and functions.
MetaMask is a crypto currency wallet for Ethereum and an instrument that helps to interact with DApps. MetaMask connects the extension to the DApp so that to fulfil both tasks. When the application identifies the MetaMask, it creates a connection, and the user can start using all the features of a specific application.

MetaMask wallet key features:

Easy to use. The first and most apparent benefit of the MetaMask wallet is the ease of use. The wallet offers an intuitive and straightforward user interface that makes the management of crypto currencies and interaction with DApps easier than ever before.

Furthermore, it allows the creation of several wallets. When users create a new wallet, they generate new public and private keys that MetaMask users have fast access.



Integration with various DApps. Meta Mask's users can connect with numerous decentralized applications with a single tap on the screen. It allows them to exchange their tokens swiftly on Uniswap or Pancake Swap exchangers, launch block chain games like Gods Unchained,.

Integration with other block chains. The additional benefit of MetaMask is that you can join it to other block chains

Hardware wallets support. MetaMask service is compatible with hardware crypto currency wallets like Ledger, Nano, or Trezor.

Swaps support. In 2021 MetaMask offered its users a relatively cheap and fast opportunity to exchange their tokens within the wallet with Swaps. This function supports several automatic marker makers (MM) that help find the best token exchange rate.

NFT support. MetaMask wallet also allows storing NFTs. When a user buys an NFT on a platform, the token is automatically displayed in the tab of collectible items in the wallet.

Main functions of a crypto currency wallet

- Registration.
- Applicable exchange rate
- Operations with the crypto currency assets
- Favourite addresses
- Employing other block chains

- NFT support

Main functions of a cryptocurrency wallet

Extended functions set for MetaMask clone

The process of wallet development

Step 1: Opening

Step 2: Developing the architecture

Step 3: UX/UI design development

Step 5: product testing

Step 6: deployment, release

Step 7: product support,

The cost of developing a MetaMask clone

Conclusion:

Assignment: 3

Title:

Write a program in solidity to create Student data. Use the following constructs:

- Structures
- Arrays
- Fall back

Deploy this as smart contract on Ethereum and observe the transaction fee and Gas values

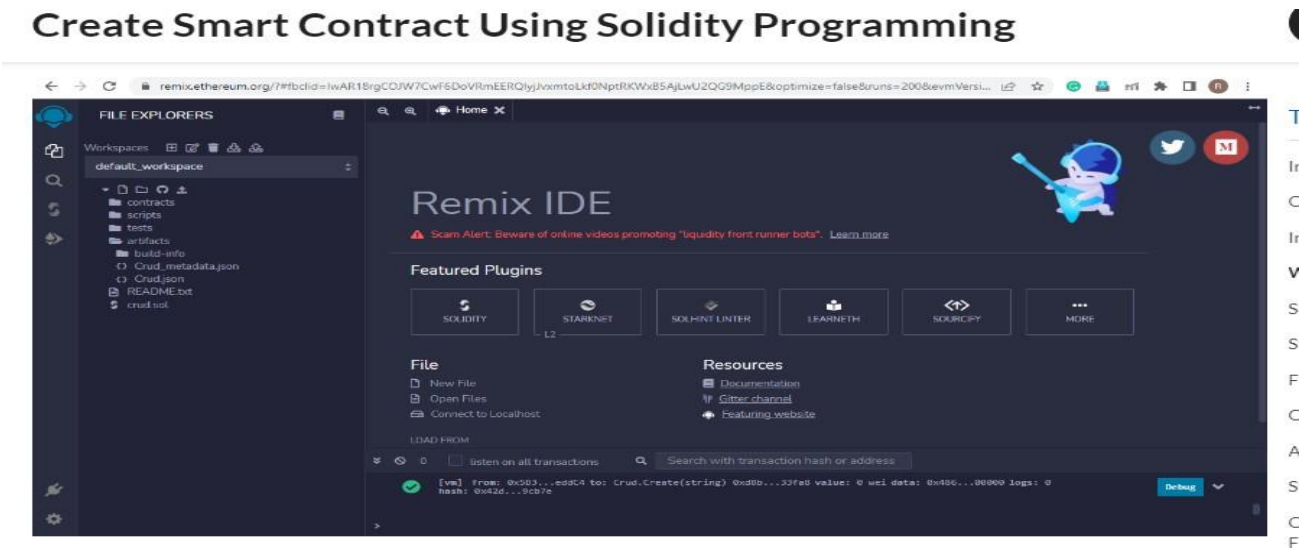
Theory:

Introduction to Solidity Programming

Solidity is object-oriented, high-level statically-typed programing language used to create smart contracts. Solidity programming looks similar to Java script, but there are a lot of differences between both languages. In solidity, you need to compile the program first, while in Java script, you can run the program directly in your browser or by using Node JS. With solidity, you can create contracts for uses such as voting, crowd funding, blind auctions, and multi-signature wallets. It is also a case-sensitive programming language. Visit the official solidity documentation to read more and be updated about any new functionality release.

What is Remix IDE?

It is an online IDE for creating solid, smart contracts, so you do not need to install or download anything to do any setup. You can develop, deploy, and administer your solidity smart contract using Remix IDE. Visit this link to access the Remix IDE where you will find multiple options and a window shown below. The window is a little bit similar to VS code where on the left-hand side, you will find some icons to terminate to other options like a compiler, file explorer, search files, deploy, etc.

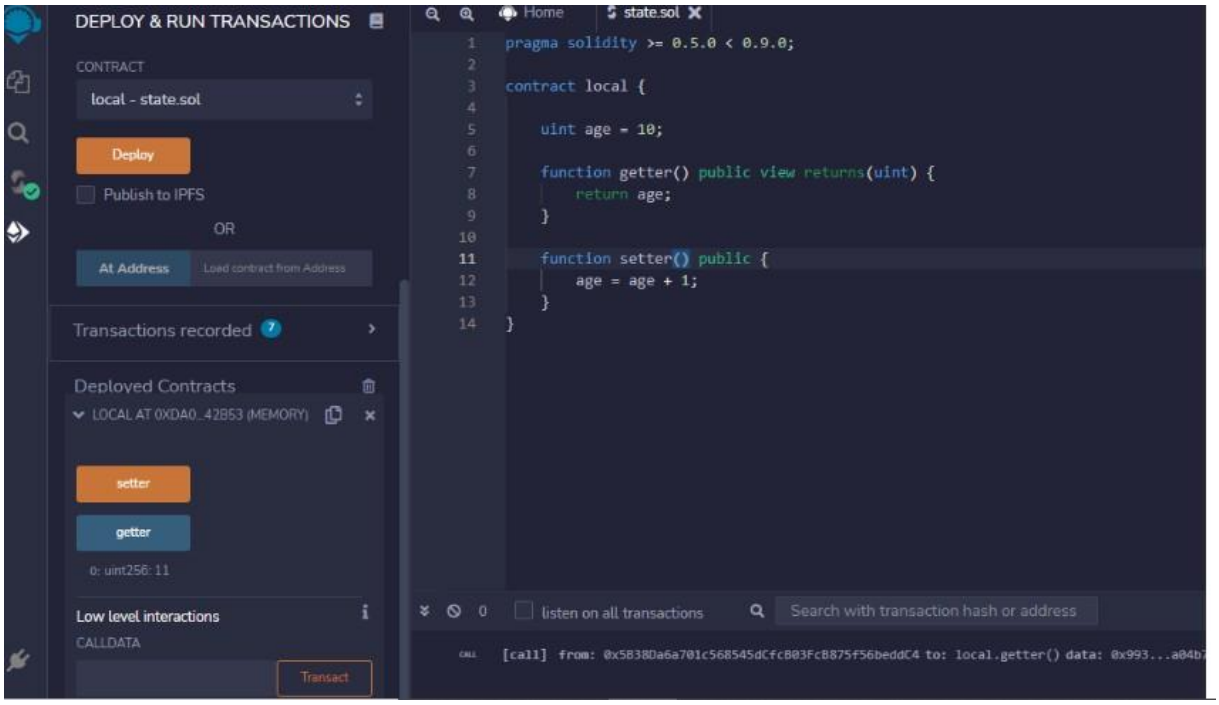


Functions in Solidity

1. **Simple public function.**

```
pragma solidity >= 0.5.0 < 0.9.0;
contract local {
uint age = 10;
function getter() public view returns(uint) {
return age;
}
function setter() public {
age = age + 1;
}
}
```

2. After writing the above code on Remix IDE in a new file with sol extension, you can compile the code, visit the deploy section, and deploy the code to observe the deploy section output as shown below. The value will increase as you click the setter and getter function buttons



3. **Constructor in solidity**

```
pragma solidity >= 0.5.0 < 0.9.0;
contract local {
     uint public count;
     constructor(uint new_count) {
          count = new_count;
     }
}
```

4. Loops

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Loops {
    uint [3] public arr;
    uint public count;
    function Whileloop() public {
        while(count < arr.length) {
            arr[count] = count;
            count++;
        }
    }
    function Forloop() public {
        for(uint i=count; i<arr.length; i++) {
            arr[count] = count;
            count++;
        }
    }
```

If-else Statements in Solidity

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
    function check(int a) public pure returns(string memory) {
        string memory value;
        if(a > 0) {
            value = "Greater Than zero";
        }
        else if(a == 0) {
            value = "Equal to zero";
        }
        else {
            value = "Less than zero";
        }
        return value;
    }
}
```

Arrays:

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
    uint [4] public arr = [10, 20, 30, 40];
    function setter(uint index, uint value) public {
        arr[index] = value;
    }
    function length() public view returns(uint) {
        return arr.length;
    }
}
```

For dynamic array

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
    uint [] public arr;
    function PushElement(uint item) public {
        arr.push(item);
    }
    function Length() public view returns(uint) {
        return arr.length;
    }
    function PopElement() public {
        arr.pop();
    }
}
```

Structure in Solidity

```
pragma solidity >= 0.5.0 < 0.9.0;
struct Student {
    uint rollNo;
    string name;
}
contract Demo {
    Student public s1;
    constructor(uint _rollNo, string memory _name) {
        s1.rollNo = _rollNo;
        s1.name = _name;
    }
    // to change the value we have to implement a setter function
    function changeValue(uint _rollNo, string memory _name) public {
        Student memory new_student = Student( {
            rollNo : _rollNo,
            name : _name
        });
        s1 = new_student;
    }
}
```

Create a Smart Contract with CRUD Functionality

```
pragma solidity ^0.5.0;

contract Crud {

    struct User {

        uint id;

        string name;

    }

    User[] public users;

    uint public nextId = 0;

    function Create(string memory name) public {

        users.push(User(nextId, name));

        nextId++;

    }

    function Read(uint id) view public returns(uint, string memory) {

        for(uint i=0; i<users.length; i++) {

            if(users[i].id == id) {

                return(users[i].id, users[i].name);

            }

        }
```

```
    }
    function Update(uint id, string memory name) public {
        for(uint i=0; i<users.length; i++) {
            if(users[i].id == id) {
                users[i].name =name;
            }
        }
    }
    function Delete(uint id) public {
        delete users[id];
    }
    function find(uint id) view internal returns(uint) {
        for(uint i=0; i< users.length; i++) {
            if(users[i].id == id) {
                return i;
            }
        }
        // if user does not exist then revert back
        revert("User does not exist");
    }
}
```

Conclusion:

Solidity is an object-oriented high-level programming language for creating a smart contract that runs on the Ethereum block chain. We have learned about the smart contract and its creation using solidity programming. Let us conclude the article with the main critical points through learning from an article.

Assignment no:4

Title: Write a survey report on types of Block chains and its real time use cases

Theory:

INTRODUCTION

Blockchain is digital, decentralized technology which maintains a record of all the transactions which happen over a peer-to-peer network. These records are stored in decentralized systems which are interconnected. Blockchains are "tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company or government)".

Blockchain innovation empowers the "production of a decentralized domain, where the cryptographically approved exchanges and information are not under the control of any outsider association". Any exchange at any point finished is recorded in an unchanging record in a certain, safe, straightforward and perpetual way, with a timestamp and different subtleties.

Blockchain allows industries to store records of any type on the blockchain. Some of them are:

- Identity management
- Business transactions
- Medical transaction records
- Transaction processing
- General documentation
- Management activities
- **Public and Private Blockchains**
    - Securing applications
    - Auditing
    - Database security
    - Digital workforce training
    - Continuity planning
    - Proper testing methods
- **Key Principles of Blockchain Which Ensure Safety**

In a public blockchain, anyone can participate in the network. It is a permissionless blockchain where even anonymous people can enter. These use the Proof-of-Work and Proof-of-Stake consensus algorithms. Public blockchains have a low transaction speed.

Examples: Bitcoin and Ethereum
Private blockchains ensure privacy and security of data. Here, participation requires an invitation which is validated by a set of rules. There is an added layer of privacy on public blockchains as participant activity for certain transactions is restricted. It uses the voting or multi-party consensus mechanism. It is a lighter blockchain and hence the transaction speed is high.

Examples: Hyperledger

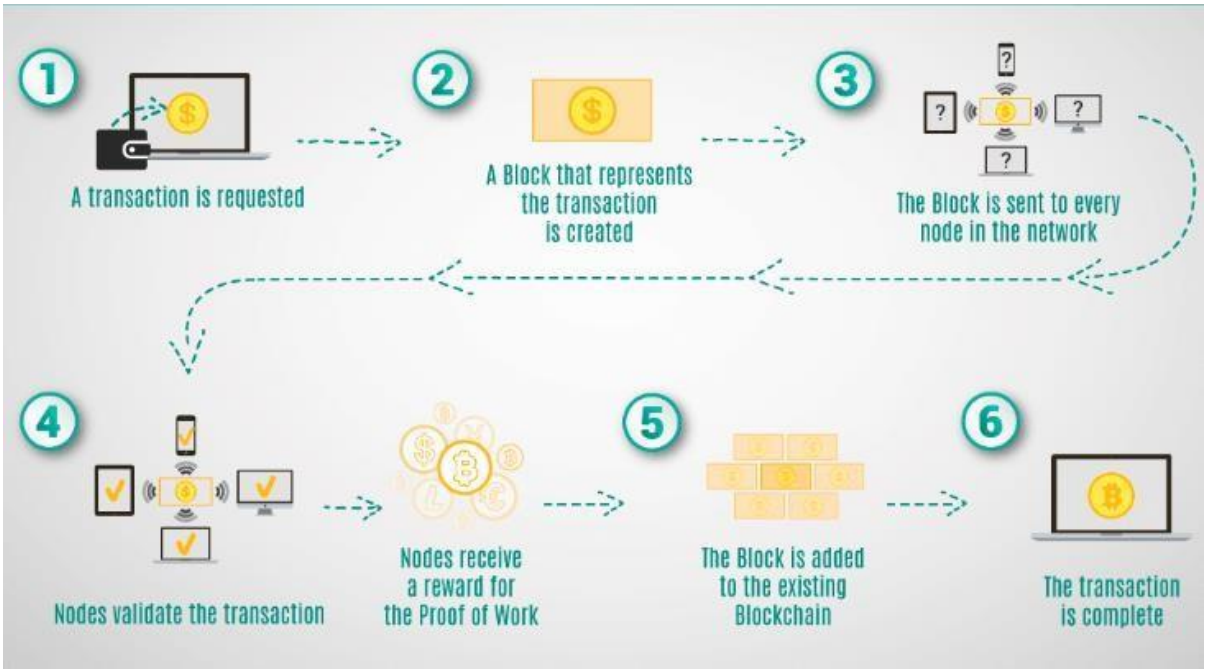


Figure: Block chain architecture diagram

Each blockchain block consists of:

- certain data
- the hash of the block
- the hash from the previous block

BLOCKCHAIN USE CASES

1. Supply Chain Management

The principle issues looked here are:

- Lack of straightforwardness as the item moves along its production network.
- Issues in item legitimacy as shoppers can now and again get fake merchandise.

The production network is a system which is built up between a business and its providers. Blockchain guarantees to discover a solution for inventory network issues through the digitization of benefits. It takes into consideration items to be labeled and relegated with one of kind characters which are then transplanted onto an unchanging, straightforward, and secure blockchain. Blockchain helps in following significant item data, for example, the condition of the item, time span of usability, time, and area. With blockchain-empowered resource digitization set up, an item's production network can be adequately transplanted onto a store network.

Consumers can verify the authenticity of purchased goods through a blockchain-enabled supply chain. Products can be accurately tracked across different locations and stages in a supply chain. This way, stakeholders will have the capability to isolate and tackle any potential issues.

2. Digital Identity

Today, digital identity is becoming problematic as centralized entities are becoming increasingly susceptible to identity thefts and data breaches. Ownership is concentrated in the hands of

applications and services to which we provide consent to use our data. Digital identity can be described as an online record of information pertaining to individuals and organizations.

In a blockchain, users can have control over their information. Instead of providing consent to many service providers, users can store their digital identity data in an encrypted digital hub. Individuals can control access to the hub and can also revoke access, if necessary. Using blockchain technology, the user can be in control of their digital data and the way in which it is utilized.

3. Healthcare

Current issues:

- Medical practitioners lack a clear and complete understanding of a patient's medical history. This hinders in providing effective healthcare solutions.
- Counterfeit or fake medicines are also a major issue within the medical supply chain.

Blockchain will serve as a tamper-proof and secure database to alleviate the problems faced in the healthcare industry. Patient medical records can be stored on a blockchain. This will make it significantly easier for medical practitioners to get a better idea of a patient's medical history. Blockchain would also help tag and track drugs at every stage of the supply chain. It will act as a medium to assure the authenticity of the drugs. Patients will also have control over the data stored in the blockchain. Others can view their data only if the patients grant them permission for the same.

4. Notary

A noteworthy part of possession resources is in paper structure. This leaves space for the records to be altered or exposed to deceitful action. Authentication is a cheat hindrance process which guarantees the gatherings of an exchange that a record can be trusted and is bona fide. Blockchain innovation will increase the value of the procedure of legally approbation. The alter safe and straightforward component of blockchain makes it an appropriate possibility for legally approbation. Blockchain can be utilized in legally approbation to guarantee evidence of-presence. Blockchain helps in demonstrating the presence of the record since the time it was made and alterations can likewise be recognized. Checking if a record has been adjusted or not should be possible by hashing the archive. On the off chance that there has been any change, the report will bring about an alternate hash and the proprietor will end up mindful of the adjustment.

5. Intellectual Property (IP)

Poorly maintained IT protocols cause unnecessary legal disputes. Adding a blockchain system can serve as a platform which provides accurate and clear ownership of IP assets. Tamper-resistant blockchains can provide a timestamp to indicate the exact recording time of an idea. This will solve any disputes regarding the origin of an idea. Blockchain also gives intellectual property owners the added advantage of protecting their IP assets infringers, for example, patent trolls.

TOP 5 BLOCKCHAIN PLATFORMS FOR DIGITAL MARKETING

In this section, we will understand the top five blockchain platforms which would suit digital marketing.

1. Ethereum

Ethereum is the world's leading programmable blockchain. Ethereum is a distributed public blockchain network. Ethereum's primary focus is on running the programming code of any

decentralized application. The Ether cryptocurrency of the Ethereum blockchain fuels the network and it is used by application developers to pay for services and transactions on the Ethereum network.

In digital marketing, hyperledger can be used as the internet to build in money and payments. It assures the security and transparency of data in digital marketing. The tokenization feature of the Ethereum blockchain helps protect credit card numbers and bank account numbers in a secure, virtual vault which can be transmitted across wireless networks. Ethereum will help set up a payment gateway which is needed for tokenization to work in order to store sensitive data which will allow the generation of the random token. The ERC-20 token can also be used for digital marketing transactions as it allows for uniform and fast transactions and confirms the transactions in an efficient manner.

2. Hyperledger

This is an open-source collaborative project which has been created to advance cross-industry blockchain technologies. It is a global collaboration which is hosted by the Linux Foundation. Hyperledger will be highly useful in digital marketing as it helps build a new generation of transactional applications which helps establish trust, transparency, and accountability for digital marketers and helps them streamline business processes and legal constraints.

Its peer-to-peer distributed ledger technology uses a system of smart contracts and other assistive technologies to help digital marketers secure the data related to their company, customers, and business processes. It reduces the cost and complexity of reaching out to customers as it stores information over a shared network which can be viewed by any marketer and customer who has access to the blockchain network. It acts as a decentralized digital operating system for marketplaces and data-sharing networks.

3. Quorum

Quorum is a version of the public Ethereum blockchain which has been developed by JP Morgan Chase. Quorum equips the Ethereum blockchain with a top layer and this enables it to perform private transactions and use different consensus algorithms which make it more robust. Quorum is an open source blockchain platform which is designed to support enterprise needs. It is specifically designed to ensure the privacy of private transactions between nodes. The quorum network does not charge for transactions. Security is the cornerstone to any business and digital marketing is no exception. As digital marketing reaches a wide audience, sharing a link to a website that hosts malware and running outdated versions of plugins, themes, and core components makes a network more prone to hacks and security breaches.

The quorum blockchain is ideal for digital marketing as it meets the organizational requirements of high speed and high throughput of private transactions. Quorum protects information greatly as information which is agreed to be private is never broadcast across the network. Privacy is of utmost importance in a quorum blockchain. Transactions are usually verified through the consensus algorithms. The constellation-transaction manager component of Quorum is responsible for the privacy of transactions. Though the transaction manager stores and allows access to encrypted transaction data and exchanges encrypted payloads, it does not hold access to sensitive private keys. Constellation enclave is another component which leverages cryptographic techniques for historical data preservation, transaction authenticity, and participant authentication. The enclave and transaction manager components of the quorum blockchain work hand in hand to secure and strengthen transaction privacy.

4. Recordskeeper

Recordskeeper is an Open Public Mineable Blockchain which is used for record-keeping and data security. It comprises of a full suite of structured and easily accessible record keeping for both individuals and organizations alike. RecordsKeeper capitalizes over the advantages of

the blockchain network such as secure transfer, authorization, integrity, and authenticity of data.

The servers used in centralized systems are prone to tampering, whereas RecordsKeeper is designed in such a manner that the data used for digital marketing is stored and protected over a secure network. RecordsKeeper cuts down costs for saving data as it stores a hash of data which acts as the digital fingerprint.

5. Corda

Corda is a flexible and agile open source blockchain platform which can scale to meet the needs of any business. A robust community of developers supports corda by working on enhancements, functionalities, and added features. Corda provides privacy to businesses by enabling them to transact directly in a secure manner using smart contracts. Corda helps streamline business operations and reduces transaction and record-keeping costs.

Apart from Quorum, digital marketers can also think of leveraging the capabilities of the Quorum blockchain for ensuring transaction privacy and securing business information. Transaction history and transaction data on the corda ledger are encrypted and are shared only with necessary parties. Its interoperable nature allows businesses to transact with each other on the corda network in a free and commercial manner. Digital marketers can use corda as the foundation to help solve the real-world problems of the digital marketing industry and build robust solutions.

Conclusion:

Blockchain is making the impossible possible by allowing people to secure digital relationships. Thanks to the advent of the blockchain, data is now being recorded, disclosed, and secured differently. Due to its tremendous potential, blockchain is being leveraged by many smart and innovative companies for enhancing their business processes and eventually becoming the business leaders of their respective industries.

Sinhgad Technical Education Society's
SMT. KASHIBAI NAVALE COLLEGE OF ENGINEERING VADGAON (BK), PUNE-411041

Assignment no 6

Title: Mini Project: Create a dApp (de-centralized app) for e-voting system.

Contains:

1. Results printouts
2. Output printouts