Sinhgad Institutes

STES's

RMD SINHGAD TECHNICAL INSTITUTES CAMPUS

RMD Sinhgad School of Engineering, Warje-58

LABORATORY MANUAL

Laboratory Practice III

Department of Computer Engineering

BE (2019Course)

Study material provided by: Vishwajeet Londhe

Join Community by clicking below links

Telegram Channel

https://t.me/SPPU_TE_BE_COMP

(for all engineering Resources)

SPPU Engineering & Technical UPDATES
WhatsApp channel

WhatsApp Channel

(for all Engg & tech updates)

https://whatsapp.com/channel/0029ValjFriICVfpcV9HFc3b

@SPPU_ENGINEERING_UPDATE

Insta Page

(for all Engg & tech updates)

https://www.instagram.com/sppu_engineering_update

Sinhgad Institutes

SinhgadTechnicalEducationSociety
RMDSINHGADTECHNICALINSTITUTESCAMPUS
(ApprovedbyAICTE&AffiliatedtoSavitribaiPhulePuneUniversity,Pune)
Off.: S.No.111/1,Warje,Pune-MumbaiBypassHighway,Pune–411058.
Phone:020-29996622/33**E-mail:**principal.rmdssoe@sinhgad.edu**Website:**http://rmdstic.sinhgad.edu

DepartmentofComputerEngineering

Vision

उत्तमपुरुषान् उत्तमाभियंतृन् निर्मातुं कटीबध्दा: वयम्|

Wearecommittedtoproducegoodhumanbeingsalongwithgood Engineers.

Mission

Holistic development of students and teachers is what we believe in and work for. We strive to achieve this by imbibing a unique value system, transparent work culture, excellent academic and physical environment conducive to learning, creativity and technology transfer. Our mandate is to generate, preserve and share knowledge

SINHGAD TECHNICAL EDUCATION SOCIETY'S

RMDSINHGAD SCHOOL OF ENGINEERING

Warje,Pune411058

Department of Computer Engineering

Sinhgad Institutes

LABORATORY MANUAL

AY:2023-24

Laboratory Practice III

BE Computer
Engineering Semester–I
SubjectCode - 410247

TEACHINGSCHEME	CREDIT	EXAMINATION
Practical:2Hrs/ Week	01	TW:50 Marks

PreparedBy:

Mrs.Pradnya Kasture
Mrs.Vanita Babanne
Mrs.Jyoti Raghatwan
Mrs.Manisha Darak

(Assistant Professor, Department of Computer Engineering)

Sinhgad Technical Education Society's

RMD Sinhgad Technical Institutes Campus

RMD SINHGAD SCHOOL OF ENGINEERING, WARJE, PUNE-58

NAAC accredited with "A" grade

Sinhgad Institutes

CERTIFICATE

This is to certify that

Mr./Ms._____

of Class_____Roll No._____has completed all the practical work/term work in subject

_____

satisfactorily in the department of Computer Engineering as prescribed by Savitribai Phule Pune University during the academic year _____

Staff In-charge **Head of Department** **Principal**

Date

Savitribai Phule Pune University
Fourth Year of Computer Engineering (2019 Course)
410246: Laboratory Practice III

Teaching Scheme:	Credit	Examination Scheme:
Practical: 04 Hours/Week	02	Term work: 50 Marks Practical: 50 Marks

Companion Course: Design and Analysis of Algorithms (410241), Machine Learning(410242), Blockchain Technology(410243)

Course Objectives:

- Learn effect of data preprocessing on the performance of machine learning algorithms
- Develop in depth understanding for implementation of the regression models.
- Implement and evaluate supervised and unsupervised machine learning algorithms.
- Analyze performance of an algorithm.
- Learn how to implement algorithms that follow algorithm design strategies namely divide and conquer, greedy, dynamic programming, backtracking, branch and bound.
- Understand and explore the working of Blockchain technology and its applications.

Course Outcomes:

After completion of the course, students will be able to
CO1: Apply preprocessing techniques on datasets.
CO2: Implement and evaluate linear regression and random forest regression models.
CO3: Apply and evaluate classification and clustering techniques.
CO4: Analyze performance of an algorithm.
CO5: Implement an algorithm that follows one of the following algorithm design strategies: divide and conquer, greedy, dynamic programming, backtracking, branch and bound.
CO6: Interpret the basic concepts in Blockchain technology and its applications

Guidelines for Instructor's Manual

The instructor's manual is to be developed as a reference and hands-on resource. It should include prologue (about University/program/ institute/ department/foreword/ preface), curriculum of the course, conduction and assessment guidelines, topics under consideration, concept, objectives, outcomes, set of typical applications/assignments/ guidelines, and references.

Guidelines for Student's Laboratory Journal

The laboratory assignments are to be submitted by students in the form of a journal. Journal consists of Certificate, table of contents, and handwritten write-up of each assignment (Title, Date of Completion, Objectives, Problem Statement, Software and Hardware requirements, Assessment grade/marks and assessor's sign, Theory- Concept in brief, algorithm, flowchart, test cases, Test Data Set(if applicable), mathematical model (if applicable), conclusion/analysis. Program codes with sample output of all performed assignments are to be submitted as a softcopy. As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers as part of write-ups and program listing to a journal must be avoided. Use of DVD containing student programs maintained by Laboratory In-charge is highly encouraged. For reference one or two journals may be maintained with program prints in the Laboratory.

Guidelines for Laboratory /Term Work Assessment

Continuous assessment of laboratory work should be based on overall performance of Laboratory assignments by a student. Assessment of each Laboratory assignment will assign grade/marks based on parameters, such as timely completion, performance, innovation, efficient codes, punctuality, documentation and neatness.

Guidelines for Practical Examination

Problem statements must be decided jointly by the internal examiner and external examiner. During practical assessment, maximum weightage should be given to satisfactory implementation of the problem statement. Relevant questions may be asked at the time of evaluation to test the student's understanding of the fundamentals, effective and efficient implementation. This will encourage, transparent evaluation and fair approach, and hence will not create any uncertainty or doubt in the minds of the students. So, adhering to these principles will consummate our team efforts to the promising start of student's academics.

Guidelines for Laboratory Conduction

The instructor is expected to frame the assignments by understanding the prerequisites, technological aspects, utility and recent trends related to the topic. The assignment framing policy needs to address the average students and inclusive of an element to attract and promote the intelligent students. Use of open source software is encouraged. Based on the concepts learned. Instructors may also set one assignment or mini-project that is suitable to each branch beyond the scope of the syllabus.
Operating System recommended :- 64-bit Open source Linux or its derivative
Programming tools recommended: - C++, Java, Python, Solidity, etc.

Virtual Laboratory:

- http://cse01-iiith.vlabs.ac.in/
- http://vlabs.iitb.ac.in/vlabs-dev/labs/blockchain/labs/index.php
- http://vlabs.iitb.ac.in/vlabs-dev/labs/machine_learning/labs/index.php

Suggested List of Laboratory Experiments/Assignments.

Assignments from all the Groups (A, B, C) are compulsory.

Course Contents

Group A: Design and Analysis of Algorithms

Any 4 assignments and 1 mini project are mandatory.

1.	Write a program to calculate Fibonacci numbers and find its step count.
2.	Implement job sequencing with deadlines using a greedy method.
3.	Write a program to solve a fractional Knapsack problem using a greedy method.
4.	Write a program to solve a 0-1 Knapsack problem using dynamic programming or branch and bound strategy.
5.	Write a program to generate binomial coefficients using dynamic programming.
6.	Design 8-Queens matrix having first Queen placed. Use backtracking to place remaining Queens to generate the final 8-queen's matrix.

7.	**Mini Project** 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. OR 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. OR 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.

Group B: Machine Learning

Any 4 assignments and 1 Mini project are mandatory.

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. 5. 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 two states: 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 whether they 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. 5. 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 confusion matrix, 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
7.	**Mini Project** Use the following dataset to analyze ups and downs in the market and predict future stock price returns based on Indian Market data from 2000 to 2020. Dataset Link: https://www.kaggle.com/datasets/sagara9595/stock-data OR Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (i.e. name, age, gender, socio-economic class, etc.). Dataset Link: https://www.kaggle.com/competitions/titanic/data

Group C: Blockchain Technology

Any 4 assignments and a Mini project are mandatory.

1.	Installation of Metamask and study spending Ether per transaction.
2.	Create your own wallet using Metamask for crypto transactions.
3.	Write a smart contract on a test network, for Bank account of a customer for following operations: • Deposit money • Withdraw Money • Show balance
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.
6.	**Mini Project:** Create a dApp (de-centralized app) for e-voting system.

@The CO-PO Mapping Matrix

CO/ PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	3	3	1	2	1	-	1	2	-	2	3
CO2	3	3	3	2	2	1	-	1	2	-	2	3
CO3	3	3	3	2	2	2	-	1	2	-	2	3
CO4	3	2	2	-	1	-	-	1	2	-	2	2
CO5	3	2	3	-	1	-	-	1	2	-	-	2
CO6	3	3	2	2	2	-	-	1	2	-	-	2

CONTENTS

Group: A

Assignment no. 1

Title of the Assignment: Title of the Assignment: Write a program non-recursive and recursive program to calculateFibonacci numbers and analyze their time and space complexity.

Objective of the Assignment: Students should be able to perform non-recursive and recursive programs to calculate Fibonacci numbers and analyze their time and space complexity.

Prerequisite:

1. Basic of Python or Java Programming
2. Concept of Recursive and Non-recursive functions
3. Execution flow of calculate Fibonacci numbers

Theory:

1. Introduction to Fibonacci numbers
2. Time and Space complexity

Introduction to Fibonacci numbers:

The Fibonacci series, named after Italian mathematician Leonardo Pisano Bogollo, later known as Fibonacci, is a series (sum) formed by Fibonacci numbers denoted as Fn. The numbers in Fibonacci sequence are given as: 0, 1, 1, 2, 3, 5, 8, 13, 21, 38, . . .

In a Fibonacci series, every term is the sum of the preceding two terms, starting from 0 and 1 as first and second terms. In some old references, the term '0' might be omitted.

What is the Fibonacci Series?

The Fibonacci series is the sequence of numbers (also called Fibonacci numbers), where every number is the sum of the preceding two numbers, such that the first two terms are '0' and '1'.

In some older versions of the series, the term '0' might be omitted. A Fibonacci series can thus be given as, 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, . . . It can be thus be observed that every term can be calculated by adding the two terms before it.

Given the first term, F0 and second term, F1 as '0' and '1', the third term here can be given as, F2

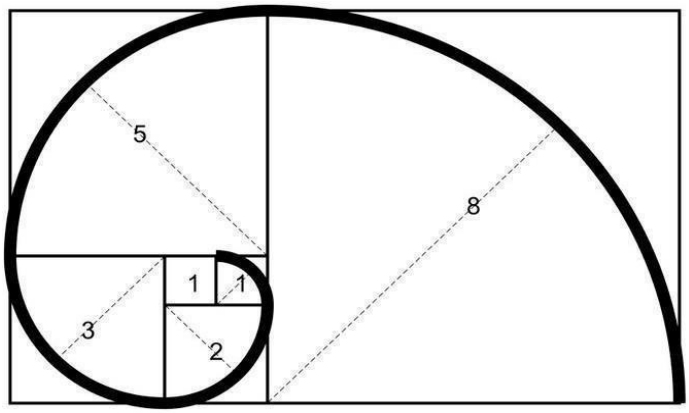
= 0 + 1 = 1

Similarly,

F3 = 1 + 1 = 2

F4 = 2 + 1 = 3

Given a number n, print n-th Fibonacci Number.



Fibonacci Sequence Formula

The Fibonacci sequence of numbers "Fn" is defined using the recursive relation with the seed values F0=0 and F1=1:

Fn = Fn-1+Fn-2

Here, the sequence is defined using two different parts, such as kick-off and recursive relation.

The kick-off part is F0=0 and F1=1.

The recursive relation part is Fn = Fn-1+Fn-2.

It is noted that the sequence starts with 0 rather than 1. So, F5 should be the 6th term of the sequence.

Examples:

Input: n=2

Output: 1

Input: n=9

Output : 34

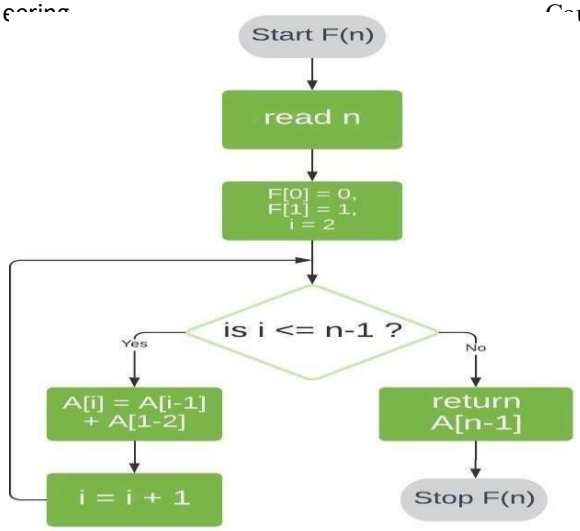
The list of Fibonacci numbers are calculated as follows:

Fn	**Fibonacci Number**
0	0
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34
… and so on.	… and so on.

Method 1 (Use Non-recursion)

A simple method that is a direct recursive implementation of mathematical recurrence relation is given above.
First, we'll store 0 and 1 in F[0] and F[1], respectively.
Next, we'll iterate through array positions 2 to n-1. At each position i, we store the sum of the two preceding array values in F[i].

Finally, we return the value of F[n-1], giving us the number at position n in the sequence. Here's a visual representation of this process:



Time and Space Complexity of Space Optimized Method

- The time complexity of the Fibonacci series is **T(N) i.e, linear**. We have to find the sum of two termsand it is repeated n times depending on the value of n.
- The space complexity of the Fibonacci series using dynamic programming is O(1).

Time Complexity and Space Complexity of Dynamic Programming

- The time complexity of the above code is T(N) i.e, linear. We have to find the sum of two terms and it is repeated n times depending on the value of n.
- The space complexity of the above code is O(N).

Method 2 (Use Recursion)

Let's start by defining F(n) as the function that returns the value of Fn.

To evaluate *F*(*n*) for *n* > 1, we can reduce our problem into two smaller problems of the same kind: *F*(*n*-1) and *F*(*n*-2).

We can further reduce F(n-1) and F(n-2) to F((n-1)-1) and F((n-1)-2); and F((n-2)-1) and F((n-2)-2), respectively.

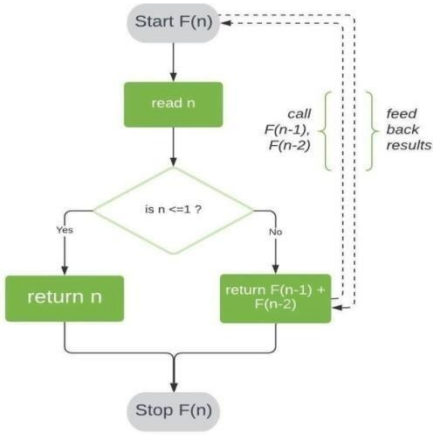
If we repeat this reduction, we'll eventually reach our known base cases and, thereby, obtain a

solution to F(n).

Employing this logic, our algorithm for $F(n)$ will have two steps:

1. Check if $n \leq 1$. If so, return n.
2. Check if $n > 1$. If so, call our function F with inputs n-1 and n-2, and return the sum of thetwo results.

Here's a visual representation of this algorithm:



Time and Space Complexity

- The time complexity of the above code is **T(2^N) i.e, exponential**.
- The Space complexity of the above code is **O(N) for a recursive series**

Method	**Time complexity**	**Space complexity**
Using recursion	T(n) = T(n-1) + T(n-2)	O(n)
Using DP	O(n)	O(1)
Space optimization of DP	O(n)	O(1)
Using the power of matrixmethod	O(n)	O(1)
Optimized matrix method	O(log n)	O(log n)
Recursive method in O(log n)time	O(log n)	O(n)
Using direct formula	O(log n)	O(1)
DP using memoization	O(n)	O(1)

Applications of Fibonacci Series

The Fibonacci series finds application in different fields in our day-to-day lives. The different patterns found in a varied number of fields from nature, to music, and to the human body follow the Fibonacci series. Some of the applications of the series are given as:

- It is used in the grouping of numbers and used to study different other special mathematical sequences.
- It finds application in Coding (computer algorithms, distributed systems, etc). For example, Fibonacci series are important in the computational run-time analysis of Euclid's algorithm, used for determining the GCF of two integers.
- It is applied in numerous fields of science like quantum mechanics, cryptography, etc.
- In finance market trading, Fibonacci retracement levels are widely used in technical analysis.

Conclusion- Hence after successful implementation of this assignment we have achieves CO4 that is analysis the performance of algorithm and recursive and non recursive method.

Group: A

Assignment no. 2

Title of the Assignment: Write a program to implement Huffman Encoding using a greedy strategy.

Objective of the Assignment: Students should be able to understand and solve Huffman Encoding using greedy method.

Prerequisite:

1. Basic of Python or Java Programming
2. Concept of Greedy method
3. Huffman Encoding concept

Theory:

1. Greedy Method
2. Huffman Encoding
3. Example solved using Huffman encoding

What is a Greedy Method?

- A greedy algorithm is an approach for solving a problem by selecting the best option available at the moment. It doesn't worry whether the current best result will bring the overall optimal result.
- The algorithm never reverses the earlier decision even if the choice is wrong. It works in a top- down approach.
- This algorithm may not produce the best result for all the problems. It's because it always goes for the local best choice to produce the global best result.

Advantages of Greedy Approach

- The algorithm is easier to describe.
- This algorithm can perform better than other algorithms (but, not in all cases).

Drawback of Greedy Approach

- As mentioned earlier, the greedy algorithm doesn't always produce the optimal solution. This is the major disadvantage of the algorithm
- For example, suppose we want to find the longest path in the graph below from root to leaf.

Greedy Algorithm

- To begin with, the solution set (containing answers) is empty.
- At each step, an item is added to the solution set until a solution is reached.
- If the solution set is feasible, the current item is kept.
- Else, the item is rejected and never considered again.

Huffman Encoding

- Huffman Coding is a technique of compressing data to reduce its size without losing any of the details. It was first developed by David Huffman.
- Huffman Coding is generally useful to compress the data in which there are frequently occurring characters.
- Huffman Coding is a famous Greedy Algorithm.
- It is used for the lossless compression of data.
- It uses variable length encoding.
- It assigns variable length code to all the characters.
- The code length of a character depends on how frequently it occurs in the given text.
- The character which occurs most frequently gets the smallest code.
- The character which occurs least frequently gets the largest code.
- It is also known as **Huffman Encoding**.

Prefix Rule

- Huffman Coding implements a rule known as a prefix rule.
- This is to prevent the ambiguities while decoding.
- It ensures that the code assigned to any character is not a prefix of the code assigned to any other character

Major Steps in Huffman Coding-

There are two major steps in Huffman Coding-

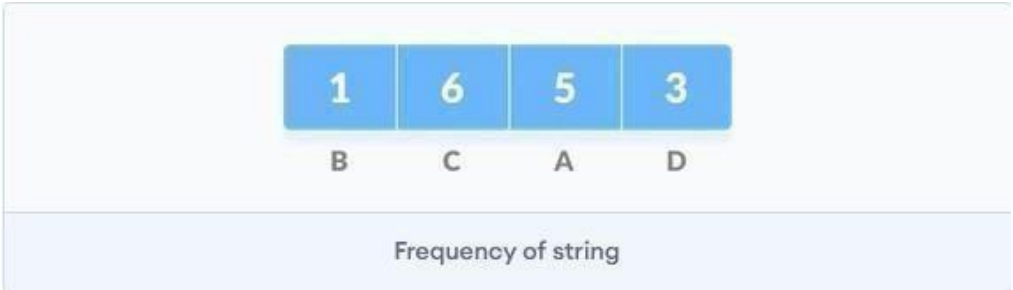
1. Building a Huffman Tree from the input characters.
2. Assigning code to the characters by traversing the Huffman Tree.

How does Huffman Coding work?

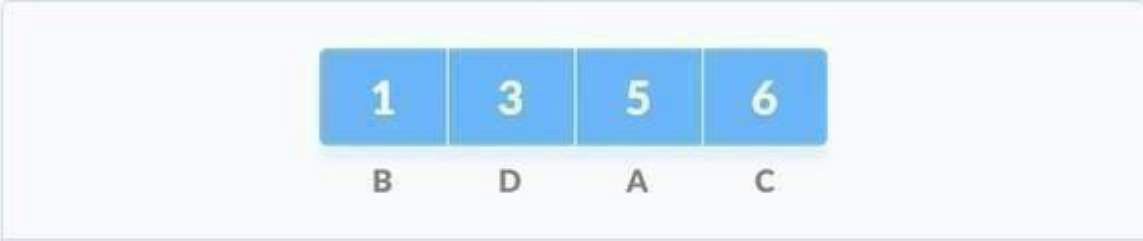
Suppose the string below is to be sent over a network.

- Each character occupies 8 bits. There are a total of 15 characters in the above string. Thus, a total of 8 * 15 = 120 bits are required to send this string.
- Using the Huffman Coding technique, we can compress the string to a smaller size.
- Huffman coding first creates a tree using the frequencies of the character and then generates code for each character.
- Once the data is encoded, it has to be decoded. Decoding is done using the same tree.
- Huffman Coding prevents any ambiguity in the decoding process using the concept of prefix code ie. a code associated with a character should not be present in the prefix of any other code. The tree created above helps in maintaining the property.
- Huffman coding is done with the help of the following steps.

1. **Calculate the frequency of each character in the string.**



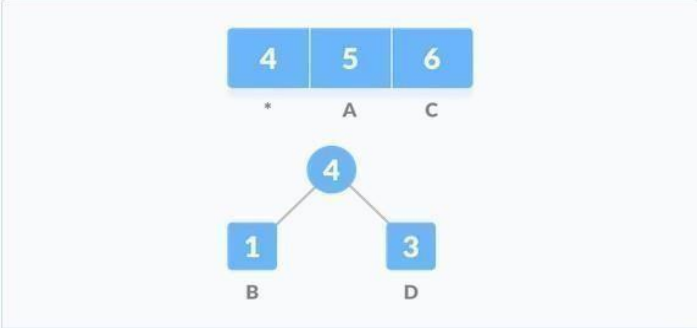
2. **Sort the characters in increasing order of the frequency. These are stored in a priority queue Q.**



Characters sorted according to the frequency

3. **Make each unique character as a leaf node.**

4. **Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above twominimum frequencies.**

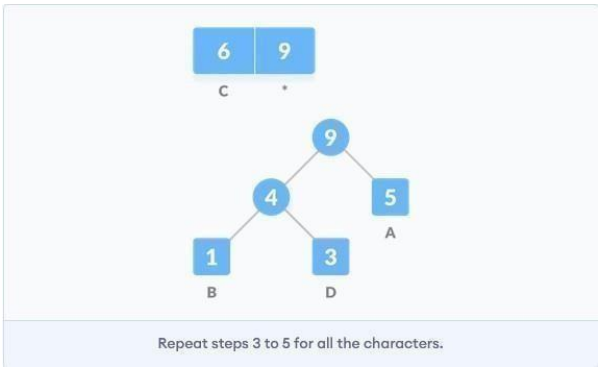


Getting the sum of the least numbers

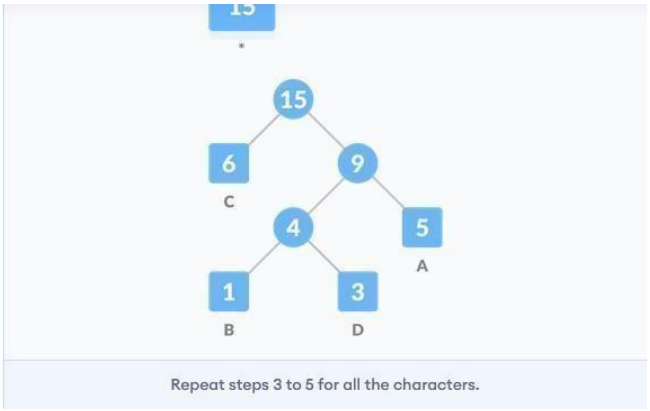
5. **Remove these two minimum frequencies from Q and add the sum into the list of frequencies (*denote the internal nodes in the figure above**

6. **Insert node z into the tree.**

7. **Repeat steps 3 to 5 for all the characters.**



Repeat steps 3 to 5 for all the characters.



Repeat steps 3 to 5 for all the characters.

8. **For each non-leaf node, assign 0 to the left edge and 1 to the right edge**

Assign 0 to the left edge and 1 to the right edge

For sending the above string over a network, we have to send the tree as well as the above compressed-code. The total size is given by the table below.

Character	Frequency	code	Size
A	5	11	5*2=10
B	1	100	1*3=3
C	6	0	6*1=6
D	3	101	3*3=9
4*8=32bits	15 bits		28 bits

Without encoding, the total size of the string was 120 bits. After encoding the size is reduced to 32 + 15 + 28 = 75.

Example

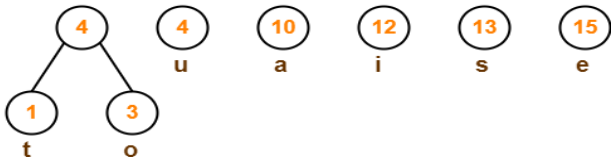
A file contains the following characters with the frequencies as shown. If Huffman Coding is used for data compression, determine-

1. Huffman Code for each character
2. Average code length
3. Length of Huffman encoded message (in bits)

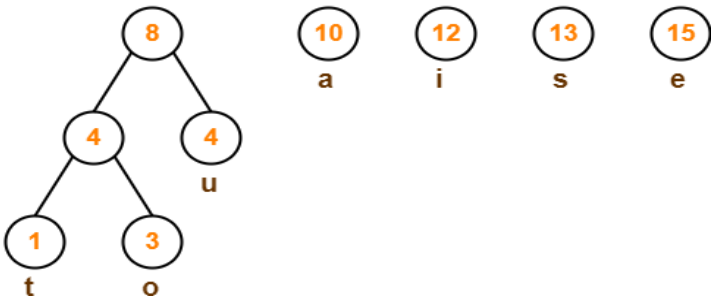
Character	Frequencies
a	10
e	15
i	12
o	3
u	4
s	13
t	1

Step-01:

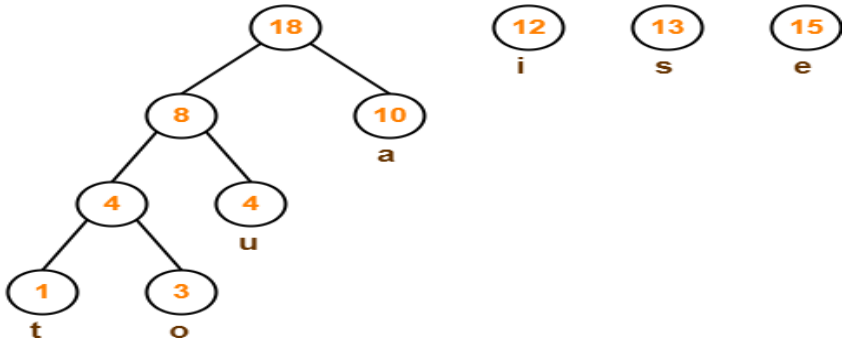
Step-02:



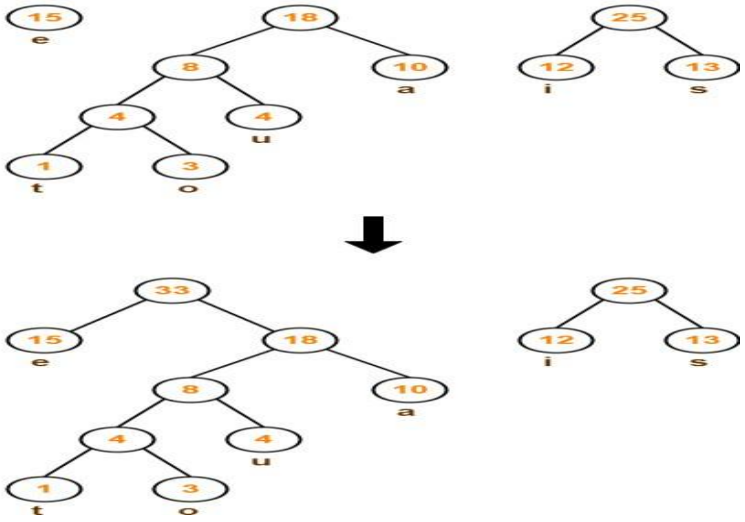
Step-03:



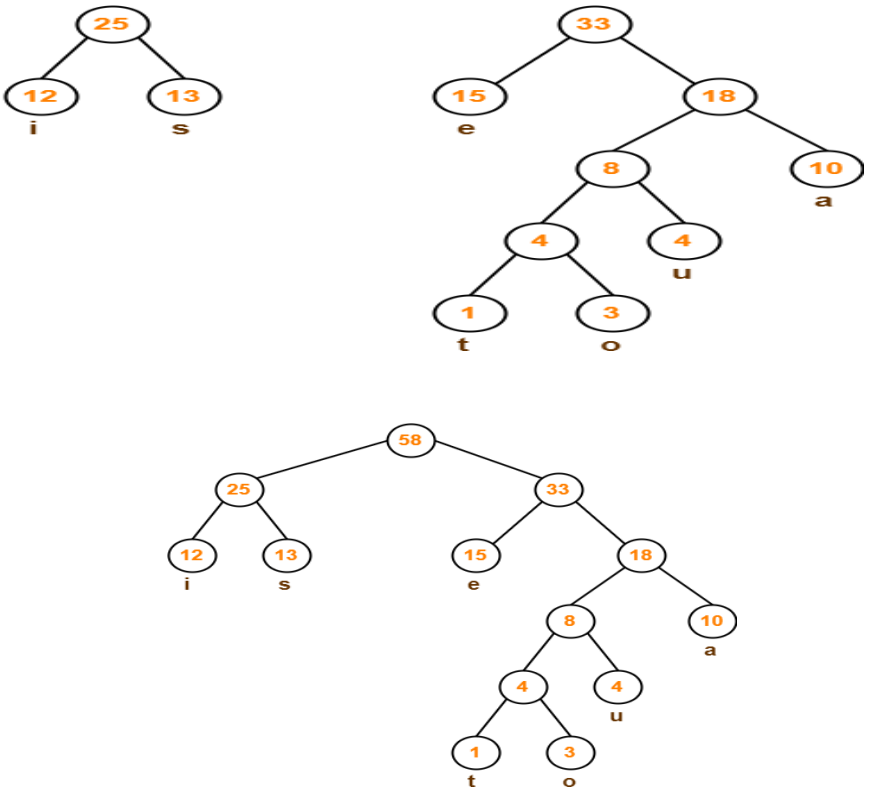
Step-04:



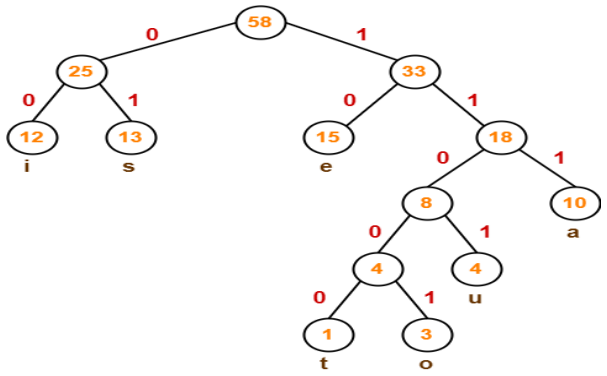
Step-06:



Step-07:



After assigning weight to all the edges, the modified Huffman Tree is-



Huffman Tree

To write Huffman Code for any character, traverse the Huffman Tree from root node to the leaf node of that character.Following this rule, the Huffman Code for each character is-

a = 111

e = 10

i = 00

o = 11001

u = 1101

s = 01

t = 11000

Time Complexity-

The time complexity analysis of Huffman Coding is as follows-

- extractMin( ) is called 2 x (n-1) times if there are n nodes.
- As extractMin( ) calls minHeapify( ), it takes O(logn) time.

Thus, Overall time complexity of Huffman Coding becomes O(nlogn).

Conclusion-

Hence after successful implementation of this assignment we have achieved CO5 that analyze the performance of greedy method for job sequencing with deadlines algorithm design strategies

Group: A

Assignment no. 3

Title of the Assignment: Write a program to solve a fractional Knapsack problem using a greedy method.

Objective of the Assignment: Students should be able to understand and solve fractional Knapsack problems using a greedy method.

Prerequisite:

1. Basic of Python or Java Programming
2. Concept of Greedy method
3. fractional Knapsack problem

Theory:

1. Greedy Method
2. Fractional Knapsack problem
3. Example solved using fractional Knapsack problem

Greedy Method:-

- A greedy algorithm is an approach for solving a problem by selecting the best option available at the moment. It doesn't worry whether the current best result will bring the overall optimal result.
- The algorithm never reverses the earlier decision even if the choice is wrong. It works in a top-down approach.
- This algorithm may not produce the best result for all the problems. It's because it always goesfor the local best choice to produce the global best result.

Advantages of Greedy Approach

- The algorithm is easier to describe.
- This algorithm can perform better than other algorithms (but, not in all cases).

Drawback of Greedy Approach

- As mentioned earlier, the greedy algorithm doesn't always produce the optimal solution. This the major disadvantage of the algorithm
- For example, suppose we want to find the longest path in the graph below from root to leaf.

Greedy Algorithm

1. To begin with, the solution set (containing answers) is empty.
2. At each step, an item is added to the solution set until a solution is reached.
3. If the solution set is feasible, the current item is kept.
4. Else, the item is rejected and never considered again.

Knapsack Problem

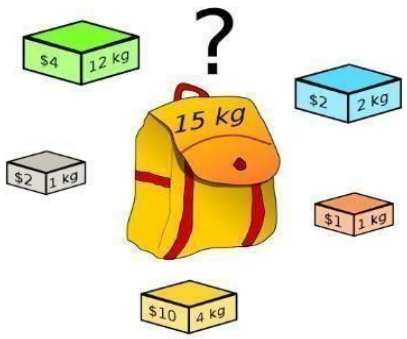
You are given the following-

- A knapsack (kind of shoulder bag) with limited weight capacity.
- Few items each having some weight and value.

The problem states-

Which items should be placed into the knapsack such that-

- The value or profit obtained by putting the items into the knapsack is maximum.
- And the weight limit of the knapsack does not exceed.



Knapsack Problem

Knapsack Problem Variants

Knapsack problem has the following two variants-

1. Fractional Knapsack Problem
2. 0/1 Knapsack Problem

Fractional Knapsack Problem-

In Fractional Knapsack Problem,

- As the name suggests, items are divisible here.
- We can even put the fraction of any item into the knapsack if taking the complete item is not possible.
- It is solved using the Greedy Method.

Fractional Knapsack Problem Using Greedy Method-

Fractional knapsack problem is solved using greedy method in the following steps-

Step-01:

For each item, compute its value / weight ratio.

Step-02:

Arrange all the items in decreasing order of their value / weight ratio.

Step-03:

Start putting the items into the knapsack beginning from the item with the highest ratio. Put as many items as you can into the knapsack.

Problem-

For the given set of items and knapsack capacity = 60 kg, find the optimal solution for the fractionalknapsack problem making use of greedy approach.

Item	Weight	Value
1	5	30
2	10	40
3	15	45
4	22	77
5	25	90

n = 5

w = 60 kg

(w1, w2, w3, w4, w5) = (5, 10, 15, 22, 25)

(b1, b2, b3, b4, b5) = (30, 40, 45, 77, 90)

Solution-

Step-01:

Compute the value / weight ratio for each item-

Items	Weight	Value	Ratio
1	5	30	6
2	10	40	4
3	15	45	3
4	22	77	3.5
5	25	90	3.6

Step-02:

Sort all the items in decreasing order of their value / weight ratio-

I1 I2 I5 I4 I3

(6) (4) (3.6) (3.5) (3)

Step-03:

Start filling the knapsack by putting the items into it one by one.

Knapsack Weight	Items in Knapsack	Cost
60	Ø	0
55	I1	30
45	I1, I2	70
20	I1, I2, I5	160

Now,

- Knapsack weight left to be filled is 20 kg but item-4 has a weight of 22 kg.
- Since in fractional knapsack problem, even the fraction of any item can be taken.
- So, knapsack will contain the following items-

< I1 , I2 , I5 , (20/22) I4 >

Total cost of the knapsack

= 160 + (20/22) x 77

= 160 + 70

= 230 units

Time Complexity-

- The main time taking step is the sorting of all items in decreasing order of their value / weight ratio.
- If the items are already arranged in the required order, then while loop takes O(n) time.
- The average time complexity of Quick Sort is O(nlogn).
- Therefore, total time taken including the sort is O(nlogn).

Conclusion- Hence after successful Implementation of this assignment we have achieved CO5 and analyze the performance the algorithm design strategy of Fractional Knapsack using greedy method.

Group: A

Assignment no. 4

Title of the Assignment:- Write a program to solve a 0-1 Knapsack problem using dynamic programming or branch and bound strategy.

Objective of the Assignment:- Students should be able to understand and solve 0-1 Knapsack problem using dynamic programming.

Prerequisite:

1. Basic of Python or Java Programming
2. Concept of Dynamic Programming
3. 0/1 Knapsack problem

Theory:

1. Dynamic Programming

2. 0/1 Knapsack problem

3. Example solved using 0/1 Knapsack problem

Dynamic Programming:-

- Dynamic Programming is also used in optimization problems. Like divide-and-conquer method, Dynamic Programming solves problems by combining the solutions of subproblems.
- Dynamic Programming algorithm solves each sub-problem just once and then saves its answer in a table, thereby avoiding the work of re-computing the answer every time.
- Two main properties of a problem suggest that the given problem can be solved using Dynamic Programming. These properties are overlapping sub-problems and optimal substructure.
- Dynamic Programming also combines solutions to sub-problems. It is mainly used where the solution of one sub-problem is needed repeatedly. The computed solutions are stored in a table, so that these don't have to be re-computed. Hence, this technique is needed

where overlapping sub- problem exists.

- For example, Binary Search does not have overlapping sub-problem. Whereas recursive program of Fibonacci numbers have many overlapping sub-problems.

Steps of Dynamic Programming Approach

Dynamic Programming algorithm is designed using the following four steps −

- Characterize the structure of an optimal solution.
- Recursively define the value of an optimal solution.
- Compute the value of an optimal solution, typically in a bottom-up fashion.
- Construct an optimal solution from the computed information.

Applications of Dynamic Programming Approach

- Matrix Chain Multiplication
- Longest Common Subsequence
- Travelling Salesman Problem

Knapsack Problem

You are given the following-

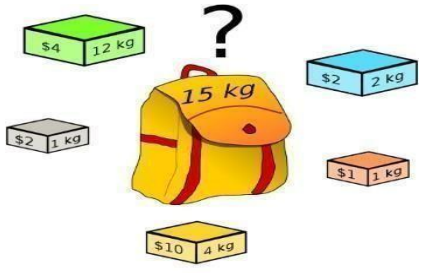
- A knapsack (kind of shoulder bag) with limited weight capacity.
- Few items each having some weight and value.

The problem states-

Which items should be placed into the knapsack such that-

- The value or profit obtained by putting the items into the knapsack is maximum.
- And the weight limit of the knapsack does not exceed.

1 Knapsack Problem Using Dyanamic Method-



Knapsack Problem

Consider-

- Knapsack weight capacity = w
- Number of items each having some weight and value = n

- 0/1 knapsack problem is solved using dynamic programming in the following steps-

Step-01:

- Draw a table say 'T' with (n+1) number of rows and (w+1) number of columns.
- Fill all the boxes of 0^{th} row and 0^{th} column with zeroes as shown-

	0	1	2	3	.....	W
0	0	0	0	0	.....	0
1	0					
2	0					
	.....					
n	0					

T-Table

Step-02:

Start filling the table row wise top to bottom from left to right. Use the following formula-

T (i , j) = max { T (i-1 , j) , valuei + T(i-1 , j – weighti) }

Here, T(i , j) = maximum value of the selected items if we can take items 1 to i and have weight restrictions of j.

- This step leads to completely filling the table.
- Then, value of the last box represents the maximum possible value that can be put into the knapsack.

Step-03:

- To identify the items that must be put into the knapsack to obtain that maximum profit,
- Consider the last column of the table.
- Start scanning the entries from bottom to top.
- On encountering an entry whose value is not same as the value stored in the entry immediately above it, mark the row label of that entry.
- After all the entries are scanned, the marked labels represent the items that must be put into the knapsack

Problem -

For the given set of items and knapsack capacity = 5 kg, find the optimal solution for the 0/1 knapsackproblem making use of a dynamic programming approach.

Item	Weight	Value
1	2	3
2	3	4
3	4	5
4	5	6

n = 4

w = 5 kg

(w1, w2, w3, w4) = (2, 3, 4, 5)

(b1, b2, b3, b4) = (3, 4, 5, 6)

Solution-

Given

- Knapsack capacity (w) = 5 kg
- Number of items (n) = 4

Step-01:

- Draw a table say 'T' with (n+1) = 4 + 1 = 5 number of rows and (w+1) = 5 + 1 = 6 number of columns.
- Fill all the boxes of 0th row and 0th column with 0.

Step-02:

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0					
2	0					
3	0					
4	0					

T-Table

Start filling the table row wise top to bottom from left to right using the formula-

$$T(i, j) = \max \{ T(i-1, j), value_i + T(i-1, j - weight_i) \}$$

Finding T(1,1)-

We have,

- i = 1
- j = 1
- $(value)_i = (value)_1 = 3$
- $(weight)_i = (weight)_1 = 2$

Substituting the values, we get-

T(1,1) = max { T(1-1 , 1) , 3 + T(1-1 , 1-2) }

T(1,1) = max { T(0,1) , 3 + T(0,-1) }

T(1,1) = T(0,1) { Ignore T(0,-1) }

T(1,1) = 0

Finding T(1,2)-

We have,

- i = 1
- j = 2
- $(value)_i = (value)_1 = 3$
- $(weight)_i = (weight)_1 = 2$

Substituting the values, we get-

T(1,2) = max { T(1-1 , 2) , 3 + T(1-1 , 2-2) }

T(1,2) = max { T(0,2) , 3 + T(0,0) }

T(1,2) = max {0 , 3+0}

T(1,2) = 3

Finding T(1,3)-

We have,

- i = 1
- j = 3
- $(value)_i = (value)_1 = 3$
- $(weight)_i = (weight)_1 = 2$

Substituting the values, we get-

T(1,3) = max { T(1-1 , 3) , 3 + T(1-1 , 3-2) }

T(1,3) = max { T(0,3) , 3 + T(0,1) }

T(1,3) = max {0 , 3+0}

T(1,3) = 3

Finding T(1,5)-

We have,

- i = 1
- j = 5
- $(value)_i = (value)_1 = 3$
- $(weight)_i = (weight)_1 = 2$

Substituting the values, we get-

T(1,5) = max { T(1-1 , 5) , 3 + T(1-1 , 5-2) }

T(1,5) = max { T(0,5) , 3 + T(0,3) }

T(1,5) = max {0 , 3+0}

T(1,5) = 3

Finding T(1,4)-

We have,

- i = 1
- j = 4
- $(value)_i = (value)_1 = 3$
- $(weight)_i = (weight)_1 = 2$

Substituting the values, we get-

T(1,4) = max { T(1-1 , 4) , 3 + T(1-1 , 4-2) }

T(1,4) = max { T(0,4) , 3 + T(0,2) }

T(1,4) = max {0 , 3+0}

T(1,4) = 3

Finding T(2,1)-

We have,

- i = 2
- j = 1
- $(value)_i = (value)_2 = 4$
- (weight)i = (weight)2 = 3

Substituting the values, we get-

T(2,1) = max { T(2-1 , 1) , 4 + T(2-1 , 1-3) }

T(2,1) = max { T(1,1) , 4 + T(1,-2) }

T(2,1) = T(1,1) { Ignore T(1,-2) }

T(2,1) = 0

Finding T(2,2)-

We have,

- i = 2
- j = 2
- $(value)_i = (value)_2 = 4$
- $(weight)_i = (weight)_2 = 3$

Substituting the values, we get-

T(2,2) = max { T(2-1 , 2) , 4 + T(2-1 , 2-3) }

T(2,2) = max { T(1,2) , 4 + T(1,-1) }

T(2,2) = T(1,2) { Ignore T(1,-1) }

T(2,2) = 3

Finding T(2,3)-

We have,

- i = 2
- j = 3
- $(value)_i = (value)_2 = 4$
- $(weight)_i = (weight)_2 = 3$

Substituting the values, we get-

T(2,3) = max { T(2-1 , 3) , 4 + T(2-1 , 3-3) }

T(2,3) = max { T(1,3) , 4 + T(1,0) }

T(2,3) = max { 3 , 4+0 }

T(2,3) = 4

Similarly, compute all the entries. After all the entries are computed and filled in the table, we get the following table-

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	3	3	3	3
2	0	0	3	4	4	7
3	0	0	3	4	5	7
4	0	0	3	4	5	7

T-Table

The last entry represents the maximum possible value that can be put into the knapsack.

So, maximum possible value that can be put into the knapsack = 7.

Identifying Items To Be Put Into Knapsack

Following Step-04,

We mark the rows labelled "1" and "2".

Thus, items that must be put into the knapsack to obtain the maximum value 7 are- Item-1 and Item-2

Time Complexity-

- Each entry of the table requires constant time θ(1) for its computation.
- It takes θ(nw) time to fill (n+1)(w+1) table entries.
- It takes θ(n) time for tracing the solution since tracing process traces the n rows.
- Thus, overall θ(nw) time is taken to solve 0/1 knapsack problem using dynamic programming

Conclusion-

Hence after successful Implementation of this assignment we have achieved CO5 and analyze Concept of 0/1 Knapsack using two different algorithm design strategies that is Dynamic Programming and branch and bound.

Group: A

Assignment no. 5

Title of the Assignment: -

Design n-Queens matrix having first Queen placed. Use backtracking to place remaining Queens to generate the final n-queen's matrix.

Objective of the Assignment:- Students should be able to understand and solve n-Queen Problem, and understand basics of Backtracking

Prerequisite:

- Basic of Python or Java Programming
- Concept of backtracking method
- Queen Problem

Theory:

1. Introduction to Backtracking
2. N-Queen Problem

Introduction to Backtracking

- Many problems are difficult to solve algorithmically. Backtracking makes it possible to solve at least some large instances of difficult combinatorial problems.
- Suppose we have to make a series of decisions among various choices, where
- We don't have enough information to know what to choose
- Each decision leads to a new set of choices.
- Some sequence of choices (more than one choices) may be a solution to your problem.

Backtracking:-

Backtracking is finding the solution of a problem whereby the solution depends on the previous steps taken. For example, in a maze problem, the solution depends on all the steps you

take one-by-one. If any of those steps is wrong, then it will not lead us to the solution. In a maze problem, we first choose a path and continue moving along it. But once we understand that the particular path is incorrect, then we just come back and change it. This is what backtracking basically is.

In backtracking, we first take a step and then we see if this step taken is correct or not i.e., whether it will give a correct answer or not. And if it doesn't, then we just come back and change our first step. In general, this is accomplished by recursion. Thus, in backtracking, we first start with a partial sub-solution of the problem (which may or may not lead us to the solution) and then check if we can proceed further with this sub-solution or not. If not, then we just come back and change it.

Thus, the general steps of backtracking are:

- Start with a sub-solution
- Check if this sub-solution will lead to the solution or not
- If not, then come back and change the sub-solution and continue again

Applications of Backtracking:-

- N Queens Problem
- Sum of subsets problem
- Graph coloring
- Hamiltonian cycles.

N queens on NxN chessboard

One of the most common examples of the backtracking is to arrange N queens on an NxN chessboard such that no queen can strike down any other queen. A queen can attack horizontally, vertically, or diagonally. The solution to this problem is also attempted in a similar way. We first place the first queen anywhere arbitrarily and then place the next queen in any of the safe places. We continue this process until the number of unplacedqueens becomes zero (a solution is found) or no safe place is left. If no safe place is left, then we change the position of the previously

placed queen.

N-Queens Problem:

A classic combinational problem is to place n queens on a n*n chess board so that no two attack, i.,e notwo queens are on the same row, column or diagonal.

N Queen Problem:-

- N Queen problem is the classical Example of backtracking. N-Queen problem is defined as, “given N x N chess board, arrange N queens in such a way that no two queens attack each other by being in the same row, column or diagonal”.

- For N = 1, this is a trivial case. For N = 2 and N = 3, a solution is not possible. So we start with N = 4and we will generalize it for N queens.

If we take n=4then the problem is called the 4 queens problem. If we take n=8 then the problem is called the 8 queens problem.

Algorithm

1. Start in the leftmost column
2. If all queens are place return true
3. Try all rows in the current column.

Do following for every tried row.

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.

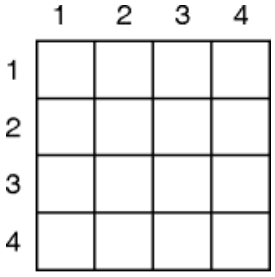
b If placing the queen in [row, column] leads to a solution then return true.

c If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and

d goto step (a) to try other rows.

4. If all rows have been tried and nothing worked,return false to trigger backtracking.

4-Queen Problem



4 x 4 Chessboard

Problem 1 : Given 4 x 4 chessboard, arrange four queens in a way, such that no two queens attack each other.That is, no two queens are placed in the same row, column, or diagonal.

- We have to arrange four queens, Q1, Q2, Q3 and Q4 in 4 x 4 chess board. We will put with queen in i^{th} row. Let us start with position (1, 1). Q1 is the only queen, so there is no issue. Partial solution is <1>
- We cannot place Q2 at positions (2, 1) or (2, 2). Position (2, 3) is acceptable. The partial solution is <1,3>.
- Next, Q3 cannot be placed in position (3, 1) as Q1 attacks her. And it cannot be placed at (3, 2), (3, 3)or (3, 4) as Q2 attacks her. There is no way to put Q3 in the third row. Hence, the algorithm backtracks and goes back to the previous solution and readjusts the position of queen Q2. Q2 is moved frompositions (2, 3) to (2, 4). Partial solution is <1, 4>
- Now, Q3 can be placed at position (3, 2). Partial solution is <1, 4, 3>.
- Queen Q4 cannot be placed anywhere in row four. So again, backtrack to the previous solution andreadjust the position of Q3. Q3 cannot be placed on (3, 3) or(3, 4). So the algorithm backtracks even further.
- All possible choices for Q2 are already explored, hence the algorithm goes back to partial solution <1> and moves the queen Q1 from (1, 1) to (1, 2). And this process continues until a solution is found.

All possible solutions for 4-queen are shown in fig (a) & fig. (b)

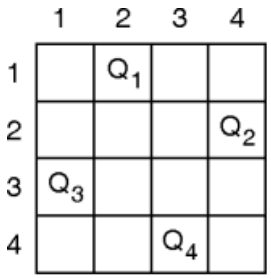


Fig. (a): Solution – 1

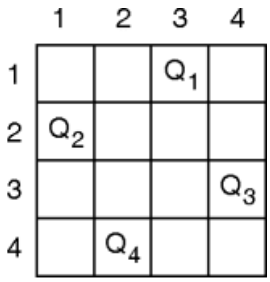
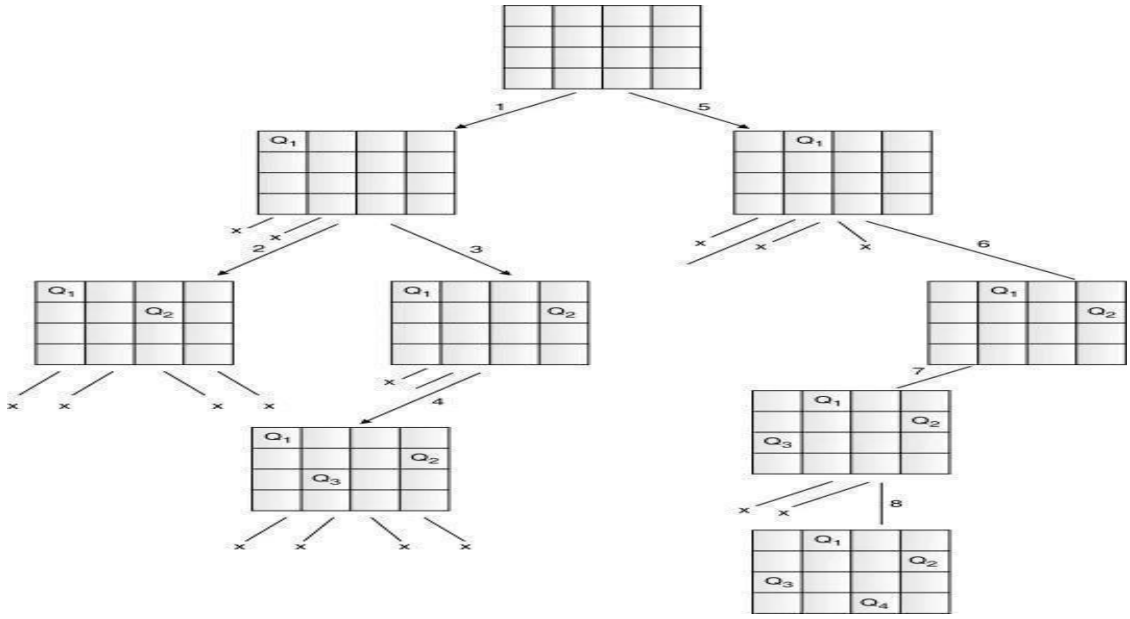
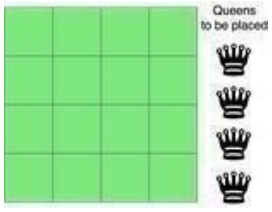


Fig. (b): Solution – 2

Following Figure describes the backtracking sequence for the 4-queen problem.



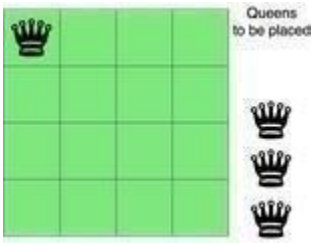
The solution of the 4-queen problem can be seen as four tuples (x1, x2, x3, x4), where x_i represents the column number of queen Q_i. Two possible solutions for the 4-queen Problem are



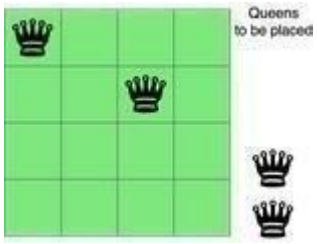
(2, 4, 1, 3) and (3, 1, 4,2).

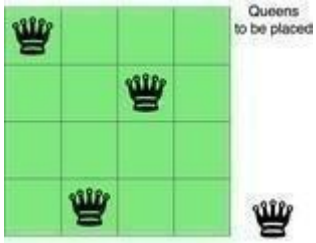
Explanation:

The above picture shows an NxN chessboard and we have to place N queens on it. So, we will start byplacing the first queen.



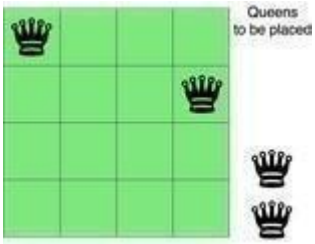
Now, the second step is to place the second queen in a safe position and then the third queen.



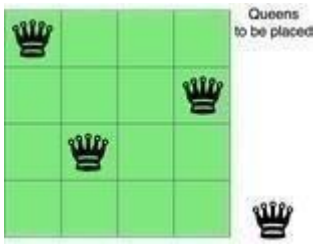


Now, you can see that there is no safe place where we can put the last queen. So, we will just change theposition of the previous queen. And this is backtracking.

Also, there is no other position where we can place the third queen so we will go back one more step andchange the position of the second queen.

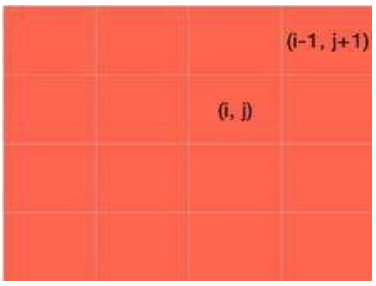


And now we will place the third queen again in a safe position until we find a solution.

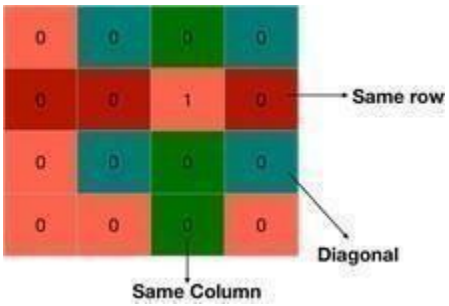


We need to check if a cell (i, j) is under attack or not. For that, we will pass these two in our function alongwith the chessboard and its size - IS-ATTACK(i, j, board, N).

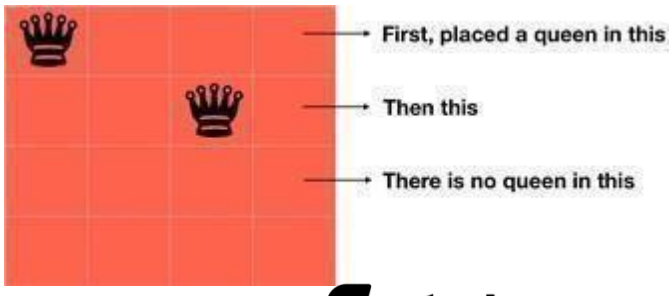
If there is a queen in a cell of the chessboard, then its value will be 1, otherwise, 0.



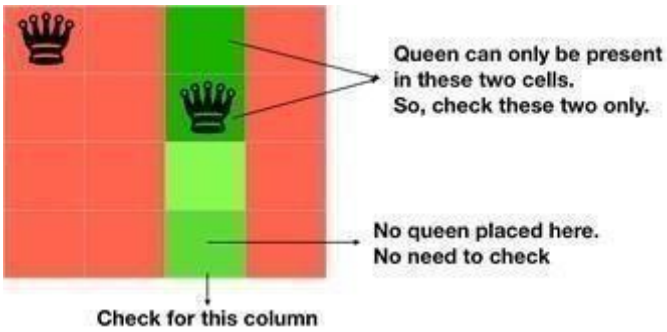
The cell (i,j) will be under attack in three condition - if there is any other queen in row i, if there is any otherqueen in the column j or if there is any queen in the diagonals.



We are already proceeding row-wise, so we know that all the rows above the current row(i) are filled but notthe current row and thus, there is no need to check for row i.



We can check for the column j by changing k from 1 to i-1 in board[k][j] because only the rows from 1 to i-1are filled.

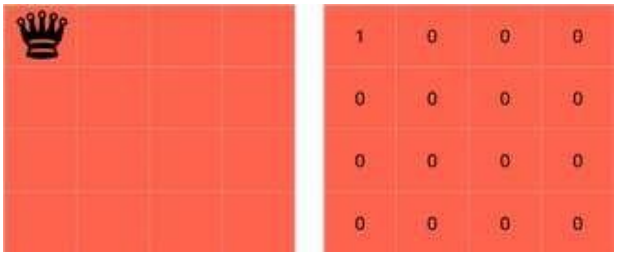


Now, we need to check for the diagonal. We know that all the rows below the row i are empty, so we need tocheck only for the diagonal elements which above the row i.

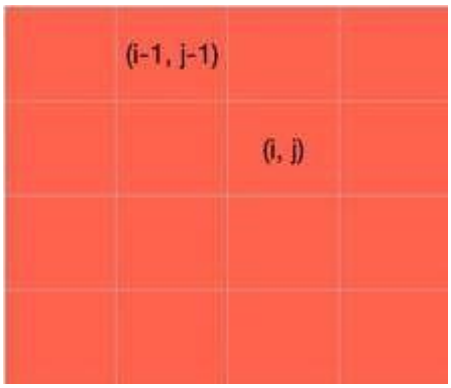
If we are on the cell (i, j), then decreasing the value of i and increasing the value of j will make us traverseover the diagonal on the right side, above the row i.

k=i-1

l=j+1



Also if we reduce both the values of i and j of cell (i, j) by 1, we will traverse over the left diagonal, above therow i.



At last, we will return false as it will be return true is not returned by the above statements and the cell (i, j) issafe.

Now, let's write the real code involving backtracking to solve the N Queen problem.

Our function will take the row, number of queens, size of the board and the board itself - N-QUEEN (row, n, N,board).

If the number of queens is 0, then we have already placed all the queens.

```
if n==0  return TRUE
```

Otherwise, we will iterate over each cell of the board in the row passed to the function and for each cell, we willcheck if we can place the queen in that cell or not. We can't place the queen in a cell if it is under attack.

```
for j in 1 to N

if !IS-ATTACK(row, j, board, N)

    board[row][j] = 1
```

After placing the queen in the cell, we will check if we are able to place the next queen with this arrangement ornot. If not, then we will choose a different position for the current queen.

```
for j in 1 to N
...
    if N-QUEEN(row+1, n-1, N, board)
        return TRUE
    board[row][j] = 0
```

if N-QUEEN(row+1, n-1, N, board) - We are placing the rest of the queens with the current arrangement. Also, since all the rows up to 'row' are occupied, so we will start from 'row+1'. If this returns true, then we are successful in placing all the queen, if not, then we have to change the position of our current queen. So, we are leaving the current cell board[row][j] = 0 and then iteration will find another place for the queen and this is backtracking.

Take a note that we have already covered the base case - if n==0 → return TRUE. It means when all queens will be placed correctly, then N-QUEEN (row, 0, N, board) will be called and this will return true.

At last, if true is not returned, then we didn't find any way, so we will return false.

```
N-QUEEN (row, n, N, board)
...
return FALSE
N-QUEEN(row, n, N, board)
  if n==0
    return TRUE
  for j in 1 to N
    if !IS-ATTACK(row, j, board, N)
                board[row][j] = 1
 if N-QUEEN(row+1, n-1, N, board)
                  return TRUE
board[row][j] = 0 //backtracking, changing current decision
  return FALSE
```

Conclusion:

Hence after successful Implementation of this assignment we have achieved CO5 and analyze Concept of Backtracking method and solve n-Queenproblem using backtracking method

Group: B

Assignment no. 1

Title of the Assignment: 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.
5. Evaluate the models and compare their respective scores like R2, RMSE, etc.

Dataset Description: The project is about on world's largest taxi company Uber Inc. In this project, we're looking to predict the fare for their future transactional cases. Uber delivers service to lakhs of customers daily. Now it becomes really important to manage their data properly to come up with new business ideas to get best results. Eventually, it becomes really important to estimate the fare prices accurately.

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

Contents of the Theory:

1. Data Preprocessing
2. Linear regression
3. Random forest regression models
4. Box Plot
5. Outliers
6. Haversine
7. Mathplotlib
8. Mean Squared Error

Data Preprocessing:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine Learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

Why do we need Data Preprocessing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning thedata and making it suitable for a machine learning model which also increases the accuracy and e ciency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

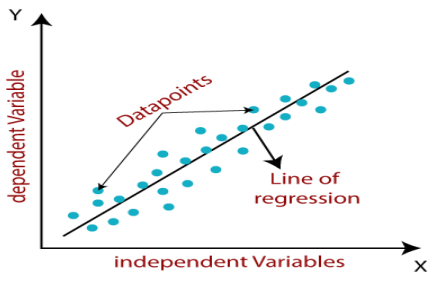
Linear Regression:

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as**sales, salary, age, product price,**etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or moreindependent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it nds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:

Random Forest Regression Models:



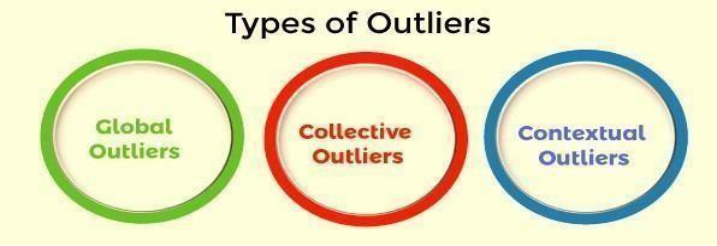
Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

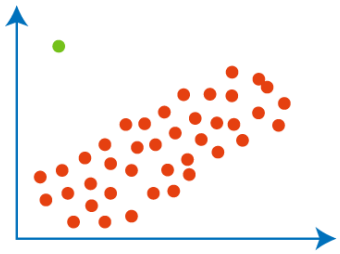
The greater number of trees in the forest leads to higher accuracy and revents the problem of Overfittring

Outliers:

As the name suggests, "outliers" refer to the data points that exist outside of what is to be expected. The major thing about the outliers is what you do with them. If you are going to analyze any task to analyze data sets, you will always have some assumptions based on how this data is generated. If you find some data points that are likely to contain some form of error then these are definitely outliers, and depending on the context, you want to overcome those errors. The data mining process involves the analysis and prediction of data that the data holds.

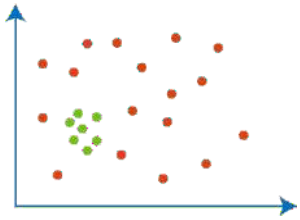


Global outliers are also called point outliers. Global outliers are taken as the simplest form of outliers. When data points deviate from all the rest of the data points in a given data set, it is known as the global outlier. In most cases, all the outlier detection procedures are targeted to determine the global outliers. The green data point is the global outlier.



Collective Outliers:

In a given set of data, when a group of data points deviates from the rest of the data set is called collective outliers. Here, the particular set of data objects may not be outliers, but when you consider the data objects as a whole, they may behave as outliers. To identify the types of different outliers, you need to go through background information about the relationship between the behaviors of outliers shown by different data objects. For example, in an Intrusion Detection System, the DOS package from one system to another is taken as normal behavior. Therefore, if this happens with the various computers simultaneously, it is considered abnormal behavior, and as a whole, they are called collective outliers. The green data points as a whole represent the collective outlier.

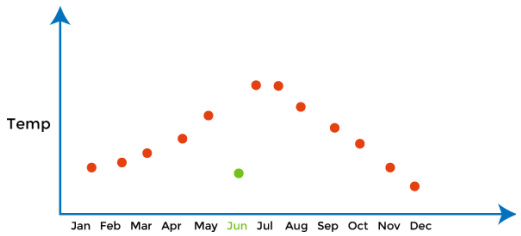


Contextual Outliers:

As the name suggests, "Contextual" means this outlier introduced within a context. For example, in the speech recognition technique, the single background noise. Contextual outliers are also known as Conditional outliers. These types of outliers happen if a data object deviates from the other data points because of any special condition in a given data set. As we know, there are two types of attributes of objects of data: contextual attributes and behavioral attributes. Contextual outlier analysis enables the users to examine outliers in different contexts and conditions, which can be useful in various applications. For example, A temperature reading of 45 degrees Celsius may behave as an outlier in a rainy season. Still, it will behave like a normal data point in the context of a summer season. In the given diagram, a green dot representing the low-temperature value in June is a contextual outlier since the same value in December is not an outlier.

Haversine:

The Haversine formula calculates the shortest distance between two points on a sphere using their latitudes and longitudes measured along the surface. It is important for use in navigation



Matplotlib:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Mean Squared Error:

The Mean Squared Error (MSE) or **Mean Squared Deviation (MSD)** of an estimator measures the average of error squares i.e., the average squared difference between the estimated values and true value. It is a risk function, corresponding to the expected value of the squared error loss. It is always non – negative and values close to zero are better. The MSE is the second moment of the error (about the origin) and thus incorporates both the variance of the estimator and its bias.

Conclusion: After Successful implementation of this assignment we are able to apply preprocessing techniques on datasets and Implement and evaluate linear regression and random forest regression models. Hence we have achieved CO1 and CO2

Group: B

Assignment no. 2

Title of the Assignment: Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months

Dataset Description: The case study is from an open-source dataset from Kaggle. The data set contains 10,000 sample points with 14 distinct features such as Customer Id, CreditScore, Geography, Gender, Age, Tenure, Balance, etc.

Link for Dataset:https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling

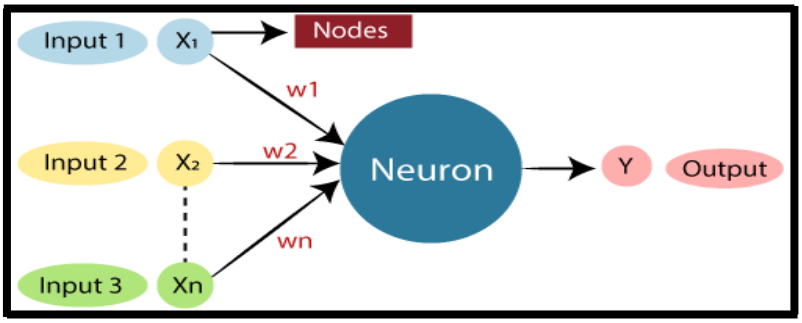
Perform following steps:

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.
5. Print the accuracy score and confusion matrix (5 points).

Contents of the Theory

1. Artificial Neural Network
2. Keras
3. tensorflow
4. Normalization
5. Confusion Matrix

The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another; artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes. The typical Artificial Neural Network looks something like the given figure.

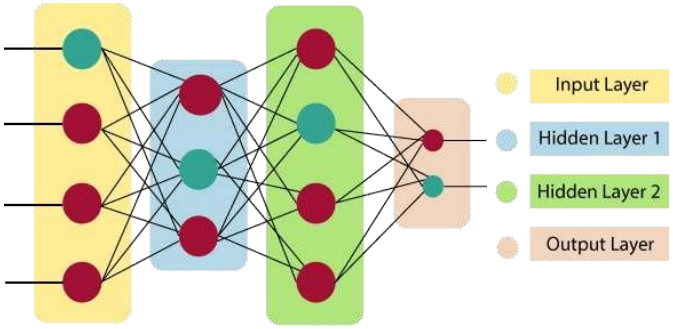


An Artificial Neural Network in the field of artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner. The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.

The architecture of an artificial neural network:

To understand the concept of the architecture of an artificial neural network, we have to understandwhat a neural network consists of. In order to de ne a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Lets us look at various types of layers available in an artificial neural network.

Artificial Neural Network primarily consists of three layers:



.

As the name suggests, it accepts inputs in several different formats provided by the programmer.

Hidden Layer:

The hidden layer presents in-between input and output layers. It performs all the It performs all the

calculations and hidden hidden features and patterns.

Output Layer:

The input goes through a series of transformations using the hidden layer which finally resultsin output that is conveyed using this layer. The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$\sum_{i=1}^{n} Wi * Xi + b$$

It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should feed or not. Only those who are fed make it to the output layer. There are distinctive activation functions available that can beapplied upon the sort of task we are performing.

Keras:

Keras is an open-source high-level Neural Network library, which is written in Python, is capable enough to run on Theano, TensorFlow, or CNTK. It was developed by one of the Google engineers, Francois Chollet. It is made user-friendly, extensible, and modular for facilitating faster experimentation with deep neural networks. It not only supports Convolution Networks and Recurrent Networks individually but also their combination.

TensorFlow:

TensorFlow is a Google product, which is one of the most famous deep learning tools widely used in the research area of machine learning and deep neural network. It came into the market on 9th November 2015 under the Apache License 2.0. It is built in such a way that it can easily run-on multiple CPUs and GPUs as well as on mobile operating systems. It consists of various wrappers in distinct languages such as Java, C++, or Python.

Normalization:

Normalization is a scaling technique in Machine Learning applied during data preparation to change the values of numeric columns in the dataset to use a common scale. It is not necessary

for all datasets in a model. It is required only when features of machine learning models have different ranges.

Mathematically, we can calculate normalization with the below formula:

$$X_n = (X - X_{minimum}) / (X_{maximum} - X_{minimum})$$

Where,

- X_n = Value of Normalization
- $X_{maximum}$ = Maximum value of a feature
- $X_{minimum}$ = Minimum value of a feature

Min-Max Scaling : This technique is also referred to as scaling. As we have already discussed above, the Min-Max scaling method helps the dataset to shift and rescale the values of their attributes, so they end up ranging between 0 and 1.

Standardization scaling:

Standardization scaling is also known as **Z-score** normalization, in which values are centered on them with a unit standard deviation, which means the attribute becomes zero and the resultant distribution has a unit standard deviation. Mathematically, we can calculate the standardization by subtracting the feature value from the mean and dividing it by standard deviation.

Hence, standardization can be expressed as follows:

$$X' = \frac{X - \mu}{\sigma}$$

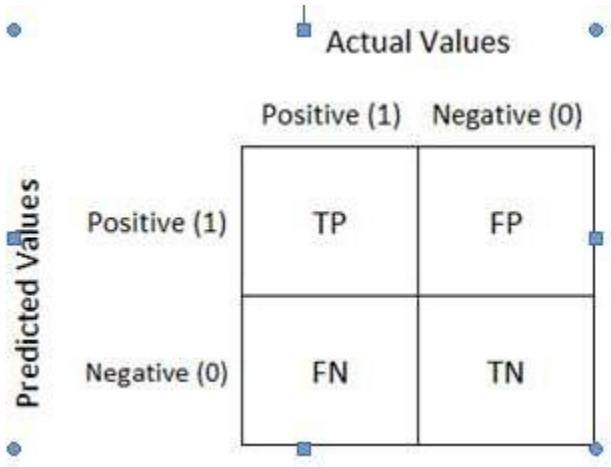
Here, μ represents the mean of feature value, and σ represents the standard deviation of feature values. However, unlike Min-Max scaling technique, feature values are not restricted to specific range in the standardization technique.

This technique is helpful for various machine learning algorithms that use distance measures such as KNN, K-means clustering, and Principal component analysis, etc. Further, it is also important that the model is built on assumptions and data is normally distributed.

Confusion Matrix:

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**. Some features of Confusion matrix are given below:

- For the 2 prediction classes of classifiers, the matrix is of 2*2 tables, for 3 classes, it is 3*3 tables, and so on.
- The matrix is divided into two dimensions, that are predicted values and actual values along with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.
- It looks like the below table:



True Positive (TP) — model correctly predicts the positive class (prediction and actual both are positive).

True Negative (TN) — model correctly predicts the negative class (prediction and actual both are

negative).

False Positive (FP) — model gives the wrong prediction of the negative class (predicted-positive, actual-negative).

FP is also called a **TYPE I** error.

False Negative (FN) — model wrongly predicts the positive class (predicted-negative, actual-positive). FN is alsocalled a **TYPE II** error.

Calculations using Confusion Matrix:

We can perform various calculations for the model, such as the model's accuracy, using this matrix. Thesecalculations are given below:

Classification Accuracy:

Accuracy = $\frac{TP+TN}{TP+FP+FN+TN}$

Error rate= $\frac{FP+FN}{TP+FP+FN+TN}$

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

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

F-measure = $\frac{\mathbf{2*Recall*precision}}{\mathbf{recall+precision}}$

Other important terms used in Confusion Matrix:

Null Error rate: It defines how often our model would be incorrect if it always predicted the majority class.

ROC Curve: The ROC is a graph displaying classifier's performance for all possible thresholds. The graph is plotted between the true positive rate (on the Y-axis) and the false Positive rate (on the x-axis)

Conclusion: After Successful implementation of this assignment we are able to apply preprocessing techniques on datasets and apply classification with the help of neural network and evaluate the performance of our Algorithm. Hence we have achieved CO1 and CO3

Group: B

Assignment no. 3

Title of the Assignment: 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

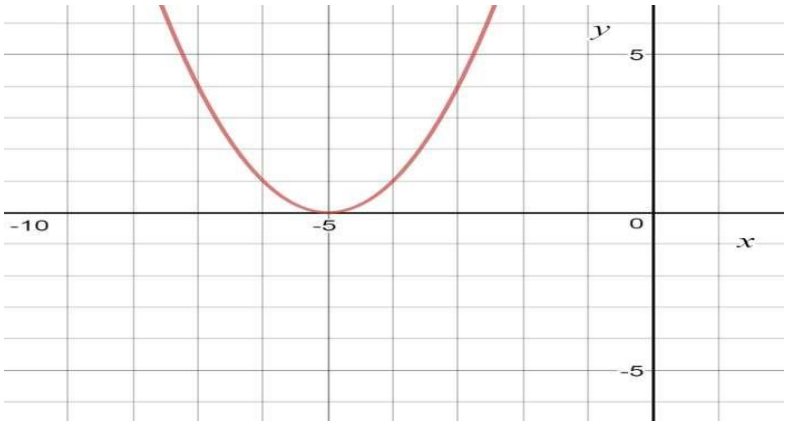
Steps should be made in proportion to the negative of the function gradient (move away from the gradient) at the current point to find local minima. Gradient Ascent is the procedure for approaching a local maximum of a function by taking steps proportional to the positive of the gradient (moving towards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

Initial Parameters:

X_0=3

Learning Rate=0.01

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

Iteration 1:

$$x_1 = x_0 - (learning\ rate) * \frac{dy}{dx}$$

$$\boldsymbol{x_1 = 3 - (0.01) * (2 * (3+5)) = 2.84}$$

Iteration 2:

$$x_2 = x_1 - (learning\ rate) * \frac{dy}{dx}$$

$$\boldsymbol{x_1 = 2.84 - (0.01) * (2 * (2.84 + 5)) = 2.6832}$$

Step 4 : We can observe that the X value is slowly decreasing and should converge to -5 (the local minima).However, how much iteration 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 !

Step 1: Initialize parameters.

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 whennumber of iterations exceeds 10,000

Conclusion: After Successful implementation of this assignment we are able to implement the Gradient Descent Algorithm and Analyze performance of an algorithm. Hence we have achieved CO4

Group: B

Assignment no. 4

Title of the Assignment: Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Computeconfusion matrix, accuracy, error rate, precision and recall on the given dataset.

Dataset Description: We will try to build a machine learning model to accurately predict whether or not thepatients in the dataset have diabetes or not?

The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

Objective:

- The Basic Concepts of k-NN algorithm.
- Compute confusion matrix, 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 familiarchoice.

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

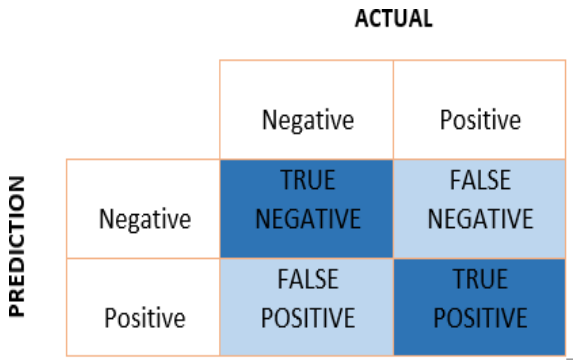
To select the K that's right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before.

- 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 about the 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:

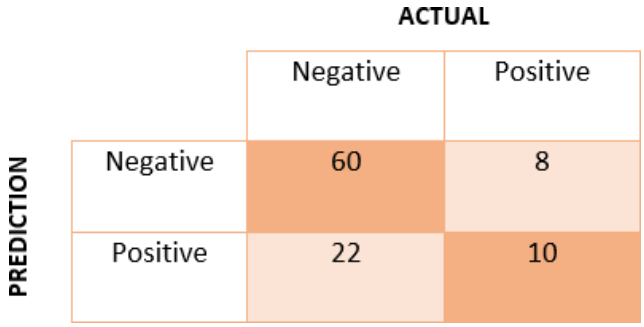
It is a matrix of size 2×2 for binary classification with actual values on one axis and predicted on another.



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'thavea 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),TrueNegative Rate (TNR), and False Negative Rate (FNR).

$$TPR = \frac{TP}{ACTUAL\ POSITIVE} = \frac{TP}{TP + FN}$$

$$TPR = \frac{FN}{ACTUAL\ POSITIVE} = \frac{FN}{TP + FN}$$

$$TNR = \frac{TN}{ACTUAL\ NEGATIVE} = \frac{TN}{TN + FP}$$

$$FPR = \frac{FP}{ACTUAL\ NEGATIVE} = \frac{FP}{TP + FP}$$

Even if data is imbalanced, we can figure out that our model is working well or not. For that, **the values of TPRand 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 **notretrieved** (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.

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

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).

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

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: After Successful implementation of this assignment we are able to apply preprocessing techniques on datasets and apply classification technique using K-Nearest Neighbors algorithm and evaluate the performance of our Algorithm. Hence we have achieved CO1 and CO3

Group: B

Assignment no. 5

Title of the Assignment: Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csvdataset. Determinethe 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:

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 centroid is 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 pointtothe 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 ofclusters (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.

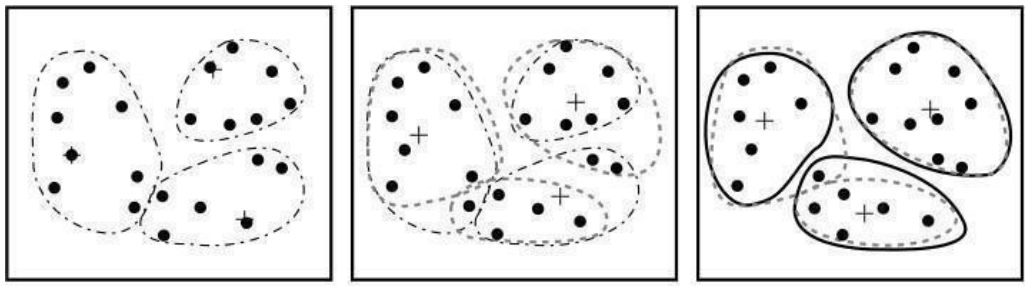
Typically, the square-error criterion is used, defined as:

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

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.



Above figure represent clustering of set of objects based on clustering methods

K-means Algorithm

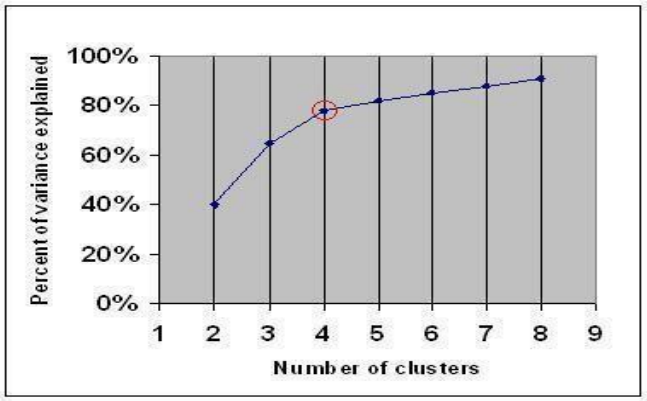
The k-means algorithm for partitioning, where is cluster center is represented by the mean value of the object in the cluster.

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 pickingthe 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 other cluster) when the centroids are recomputed. If variables are huge, then K-Means most of the times computationally faster than hierarchical clustering, if we keep k small

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: After Successful implementation of this assignment we are able to apply preprocessing techniques on datasets and apply Clustering technique using K-Means algorithm and evaluate the performance of our Algorithm. Hence we have achieved CO1 and CO3.

Group C

Assignment No: 1

Title of the Assignment: Installation of MetaMask and study spending Ether per transaction

Objective of the Assignment: Students should be able to learn new technology such as metamask. Its application and implementations

Prerequisite:

1. Basic knowledge of cryptocurrency
2. Basic knowledge of distributed computing concept
3. Working of blockchain

Contents for Theory:

1. **Introduction Blockchain**
2. **Cryptocurrency**
3. **Transaction Wallets**
4. **Ether transaction**
5. **Installation Process of Metamask**

Introduction to Blockchain

- Blockchain can be described as a data structure that holds transactional records and while ensuring security, transparency, and decentralization. You can also think of it asa chain or records stored in the forms of blocks which are controlled by no single authority.
- A blockchain is a distributed ledger that is completely open to any and everyone on the network. Once an information is stored on a blockchain, it is extremely difficult to change or alter it.
- Each transaction on a blockchain is secured with a digital signature that proves its authenticity. Due to the use of encryption and digital signatures, the data stored on the blockchain is tamper-proof and cannot be changed.
- Blockchain technology allows all the network participants to reach an agreement,

commonly known as consensus. All the data stored on a blockchain is recorded digitally and has a common history which is available for all the network participants. This way, the chances of any fraudulent activity or duplication of transactions is eliminated without the need of a third-party.

Blockchain Features

The following features make the revolutionary technology of blockchain stand out:

- **Decentralized**

  Blockchains are decentralized in nature meaning that no single person or group holds the authority of the overall network. While everybody in the network has the copy of the distributed ledger with them, no one can modify it on his or her own. This unique feature of blockchain allows transparency and security while giving power to the users.

- **Peer-to-Peer Network**

  With the use of Blockchain, the interaction between two parties through a peer-to-peer model is easily accomplished without the requirement of any third party. Blockchain uses P2P protocol which allows all the network participants to hold an identical copy of transactions, enabling approval through a machine consensus. For example, if you wish to make any transaction from one part of the world to another, you can do that with blockchain all by yourself within a few seconds. Moreover, any interruptions or extra charges will not be deducted in the transfer.

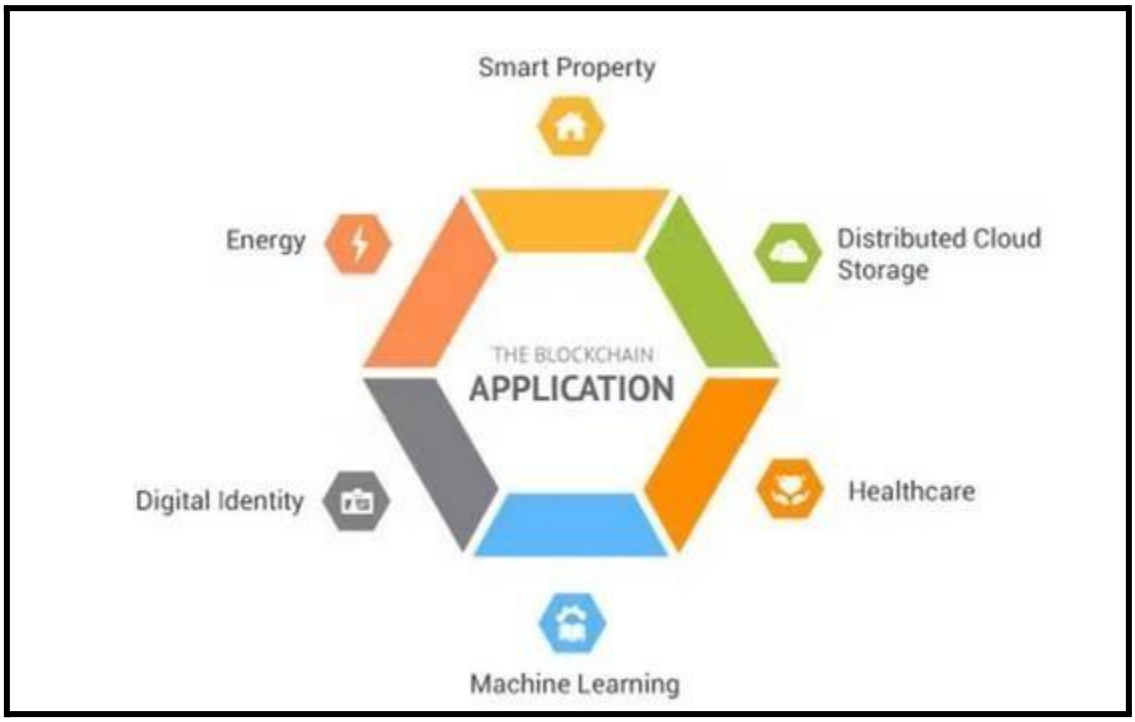
- **Immutable**

  The immutability property of a blockchain refers to the fact that any data once written on the blockchain cannot be changed. To understand immutability, consider sending email as an example. Once you send an email to a bunch of people, you cannot take it back. In order to find a way around, you'll have to ask all the recipients to delete your email which is pretty tedious. This is how immutability works.

- **Tamper-Proof**

  With the property of immutability embedded in blockchains, it becomes easier to detect tampering of any data. Blockchains are considered tamper-proof as any change in even one single block can be detected and addressed smoothly. There are two key ways of detecting tampering namely, hashes and blocks.

Popular Applications of Blockchain Technology



Benefits of Blockchain Technology:

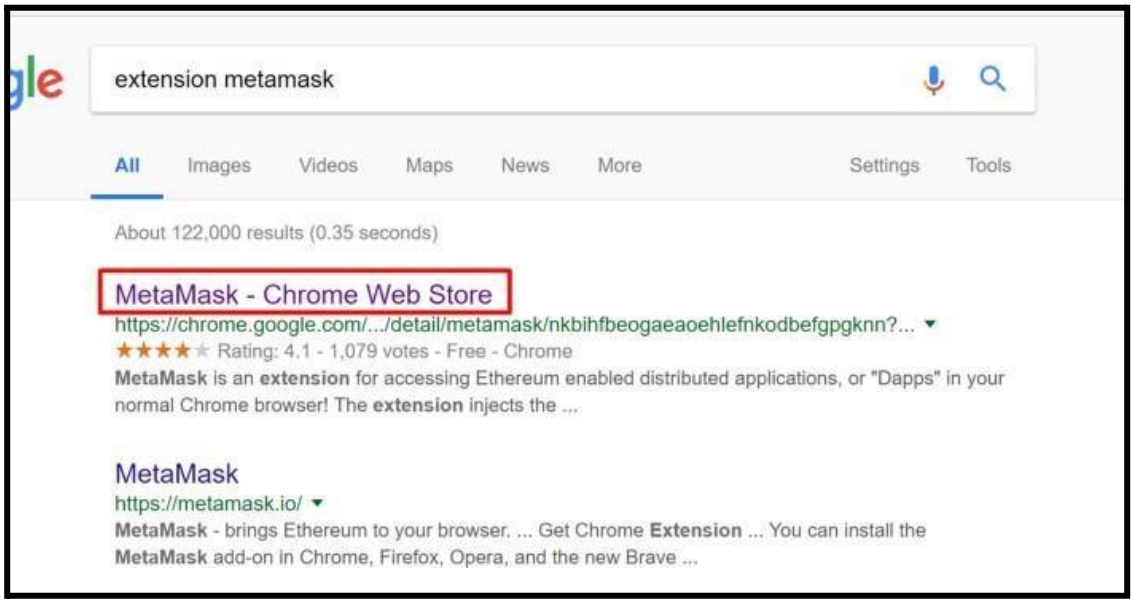
- **Time-saving:** No central Authority verification needed for settlements making theprocess faster and cheaper.
- **Cost-saving:** A Blockchain network reduces expenses in several ways. No need for third-party verification. Participants can share assets directly. Intermediaries are reduced. Transaction efforts are minimized as every participant has a copy of shared ledger.
- **Tighter security:** No one can temper with Block chain Data as it is shared among millions of participants. The system is safe against cybercrimes and Fraud.
- In finance market trading, Fibonacci retracement levels are widely used in technicalanalysis.

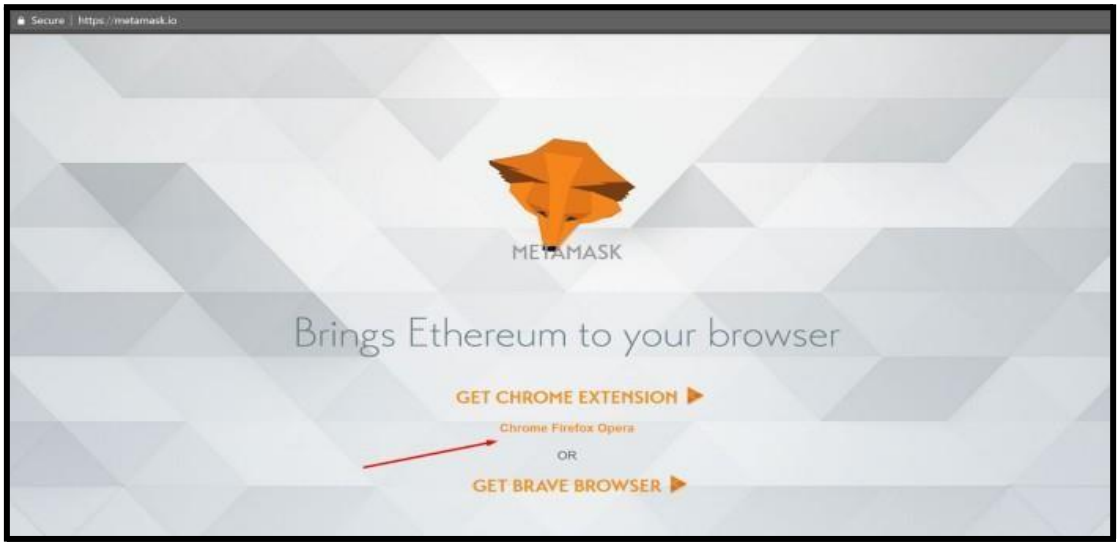
How to use MetaMask: A step by step guide

MetaMask is one of the most popular browser extensions that serves as a way of storing your Ethereum and other ERC-20 Tokens. The extension is free and secure, allowing web applications to read and interact with Ethereum's blockchain.

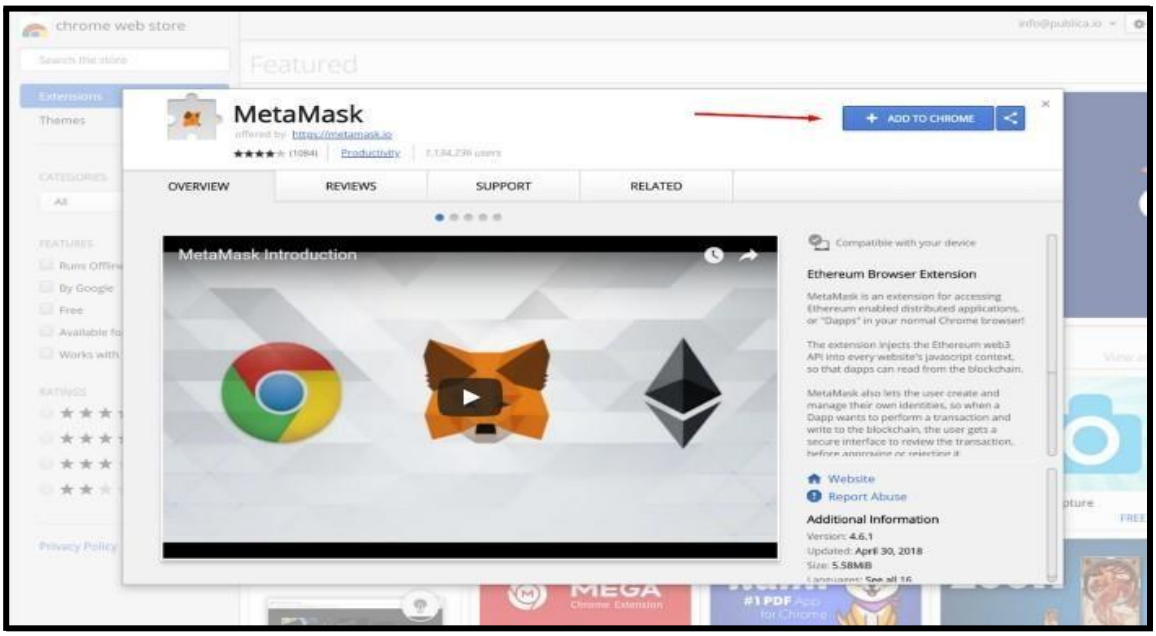
Step 1. Install MetaMask on your browser.

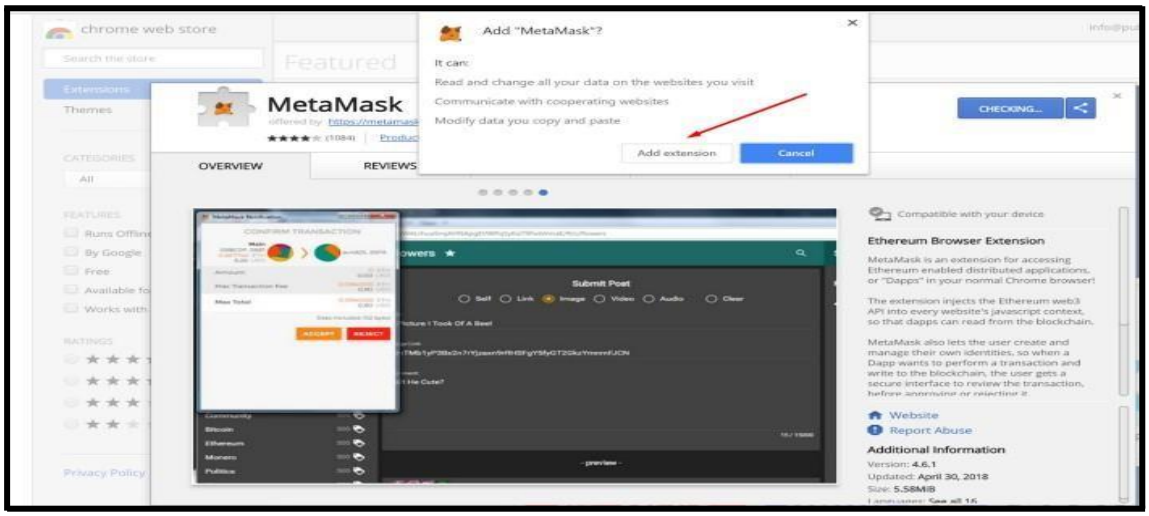
To create a new wallet, you have to install the extension first. Depending on your browser, there are different marketplaces to find it. Most browsers have MetaMask on their stores, so it's not that hard to see it, but either way, here they are Chrome, Firefox, and Opera.





- Click on **Install MetaMask** as a Google Chrome extension.
- Click **Add to Chrome**.
- Click **Add Extension**.





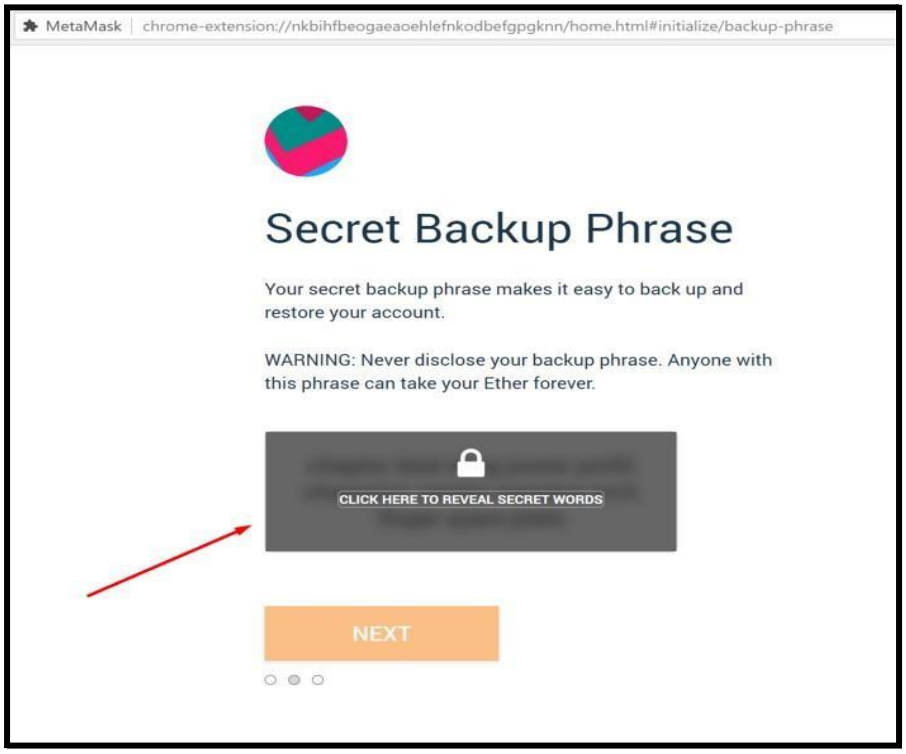
nd it's as easy as that to install the extension on your browser, continue reading the next stepto figure out how to create an account.

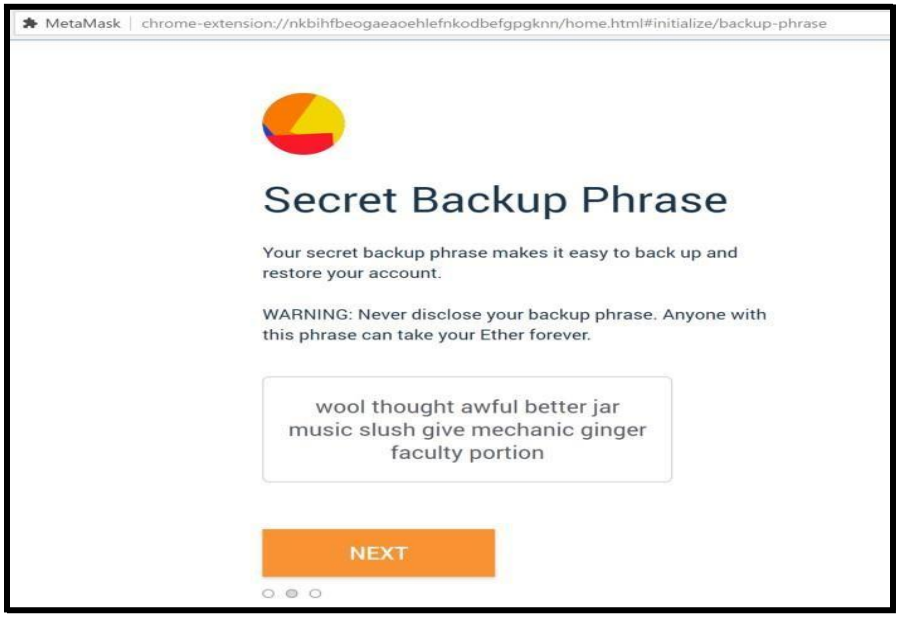
Step 2. Create an account.

- Click on the extension icon in the upper right corner to open MetaMask.
- To install the latest version and be up to date, **click Try it now**.
- **Click Continue**.
- You will be prompted to create a new password. **Click Create**.

- Proceed by **clicking Next** and accept the Terms of Use.

Click Reveal Secret Words. There you will see a 12 words seed phrase. This is reallyimportant and usually not a good idea to store digitally, so take your time and write it down



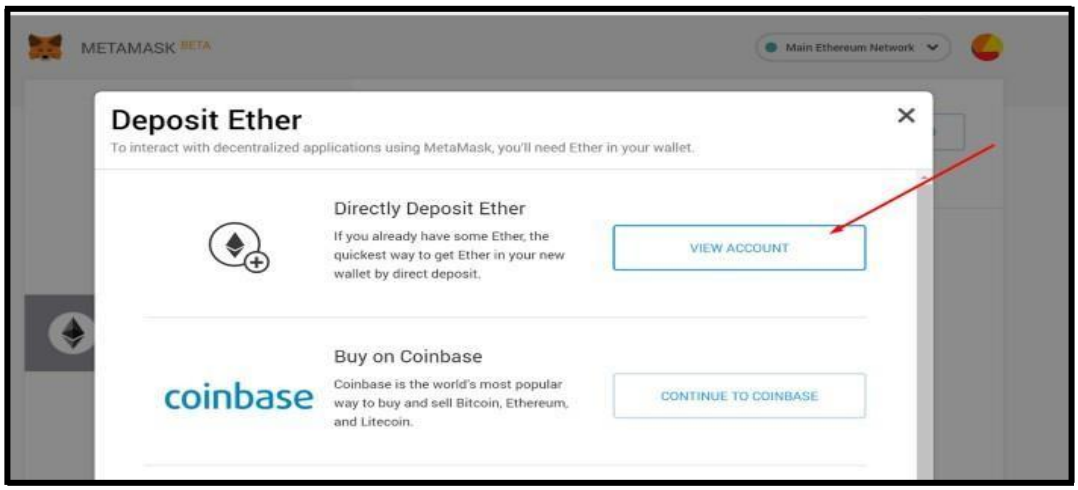


- Verify your secret phrase by selecting the previously generated phrase in order. **ClickConfirm**.

And that's it; now you have created your MetaMask account successfully. A new Ethereum wallet address has just been created for you. It's waiting for you to deposit funds, and if you want to learnhow to do that, look at the next step below.

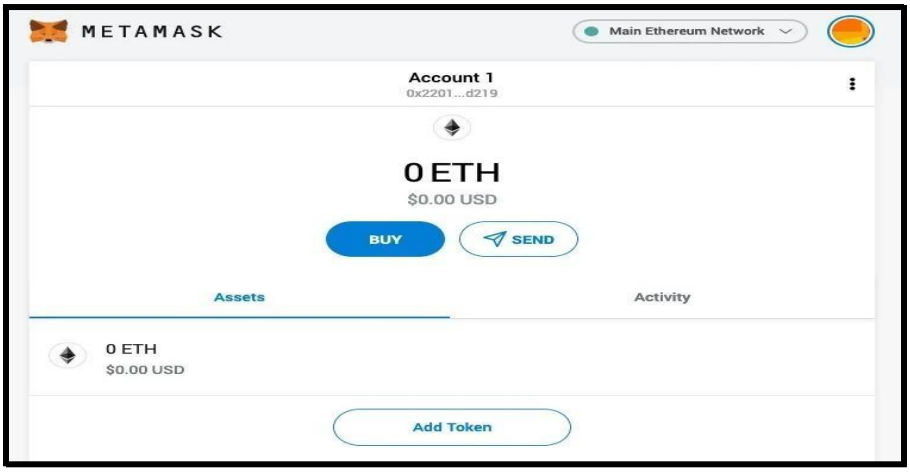
Step 3. Depositing funds.

- Click on **View Account**.



You can now see your public address and share it with other people. There are some methods to buy coins offered by MetaMask, but you can do it differently as well; you just need your address.

If you ever get logged out, you'll be able to log back in again by clicking the MetaMask icon, which will have been added to your web browser (usually found next to the URL bar).



You can now access your list of assets in the 'Assets' tab and view your transaction history in the 'Activity' tab.

- Sending crypto is as simple as clicking the 'Send' button, entering the recipient address and amount to send, and selecting a transaction fee. You can also manually adjust the transaction fee using the 'Advanced Options' button, using information from ETH Gas Station or similar platforms to choose a more acceptable gas price.
- After clicking 'Next', you will then be able to either confirm or reject the transaction on the subsequent page.

- To use MetaMask to interact with a dapp or smart contract, you'll usually need to find a 'Connect to Wallet' button or similar element on the platform you are trying to use. Afterclicking this, you should then see a prompt asking whether you want to let the dapp connect to your wallet.

What advantages does MetaMask have?

- **Popular** - It is commonly used, so users only need one plugin to access a wide range ofdapps.
- **Simple** - Instead of managing private keys, users just need to remember a list of words, andtransactions are signed on their behalf.
- **Saves space** - Users don't have to download the Ethereum blockchain, as MetaMask sendsrequests to nodes outside of the user's computer.
- **Integrated** - Dapps are designed to work with MetaMask, so it becomes much easier to sendEther in and out.

Conclusion-In this way we have explored Concept Blockchain and metamask wallet for transaction of digital currency and also mapped with CO6.

Group C

Assignment No: 2

Title of the Assignment: Create your own wallet using Metamask for crypto transactions

Objective of the Assignment: Students should be able to learn about cryptocurrencies and learn howtransaction done by using different digital currency

Prerequisite:

1. Basic knowledge of cryptocurrency
2. Basic knowledge of distributed computing concept
3. Working of blockchain

Contents for Theory:

1. Cryptocurrency
2. Transaction Wallets
3. Ether transaction

Introduction to Cryptocurrency

- Cryptocurrency is a digital payment system that doesn't rely on banks to verify transactions. It's a peer-to-peer system that can enable anyone anywhere to send and receive payments. Instead of being physical money carried around and exchanged in the real world, cryptocurrency payments exist purely as digital entries to an online database describing specific transactions. When you transfer cryptocurrency funds, the transactions are recorded in a public ledger. Cryptocurrency is stored in digital wallets.
- Cryptocurrency received its name because it uses encryption to verify transactions. This means advanced coding is involved in storing and transmitting cryptocurrency data between wallets and to public ledgers. The aim of encryption is to provide security and safety.
- The first cryptocurrency was Bitcoin, which was founded in 2009 and remains the best known today. Much of the interest in cryptocurrencies is to trade for profit, with speculatorsat times driving prices skyward.

How does cryptocurrency work?

- Cryptocurrencies run on a distributed public ledger called blockchain, a record of all transactions updated and held by currency holders.
- Units of cryptocurrency are created through a process called mining, which involves using computer power to solve complicated mathematical problems that generate coins. Users can also buy the currencies from brokers, then store and spend them using cryptographic wallets.
- If you own cryptocurrency, you don't own anything tangible. What you own is a key that allows you to move a record or a unit of measure from one person to another without atrusted third party.
- Although Bitcoin has been around since 2009, cryptocurrencies and applications of blockchain technology are still emerging in financial terms, and more uses are expected in thefuture. Transactions including bonds, stocks, and other financial assets could eventually be traded using the technology.

Cryptocurrency examples

There are thousands of cryptocurrencies. Some of the best known include:

- **Bitcoin:**
  Founded in 2009, Bitcoin was the first cryptocurrency and is still the most commonly traded. The currency was developed by Satoshi Nakamoto – widely believed to be a pseudonym for an individual or group of people whose precise identity remains unknown.

- **Ethereum:**

  Developed in 2015, Ethereum is a blockchain platform with its own cryptocurrency, called Ether (ETH) or Ethereum. It is the most popular cryptocurrency after Bitcoin.

- **Litecoin:**

  This currency is most similar to bitcoin but has moved more quickly to develop new innovations, including faster payments and processes to allow more transactions.

- **Ripple:**

Ripple is a distributed ledger system that was founded in 2012. Ripple can be used to track different kinds of transactions, not just cryptocurrency. The company behind it has worked with various banks and financial institutions.

- Non-Bitcoin cryptocurrencies are collectively known as “altcoins” to distinguish them from the original.

How to store cryptocurrency

- Once you have purchased cryptocurrency, you need to store it safely to protect it from hacks or theft. Usually, cryptocurrency is stored in crypto wallets, which are physical devices or online software used to store the private keys to your cryptocurrencies securely. Some exchanges provide wallet services, making it easy for you to store directly through the platform. However, not all exchanges or brokers automatically provide wallet services for you.
- There are different wallet providers to choose from. The terms “hot wallet” and “cold wallet”are used:
- **Hot wallet storage:** "hot wallets" refer to crypto storage that uses online software to protectthe private keys to your assets.
- **Cold wallet storage:** Unlike hot wallets, cold wallets (also known as hardware wallets) relyon offline electronic devices to securely store your private keys.

Conclusion-In this way we have explored Concept Cryptocurrency and learn how transactions are done using digital currency mapped with CO6.

.

Group C

Assignment No: 3

Title of the Assignment: Write a smart contract on a test network, for Bank account of a customer for following operations: Deposit money Withdraw Money Show balance

Objective of the Assignment: Students should be able to learn about smart contract for banking application and able to perform some operation like Deposit money Withdraw Money Show balance

Prerequisite:

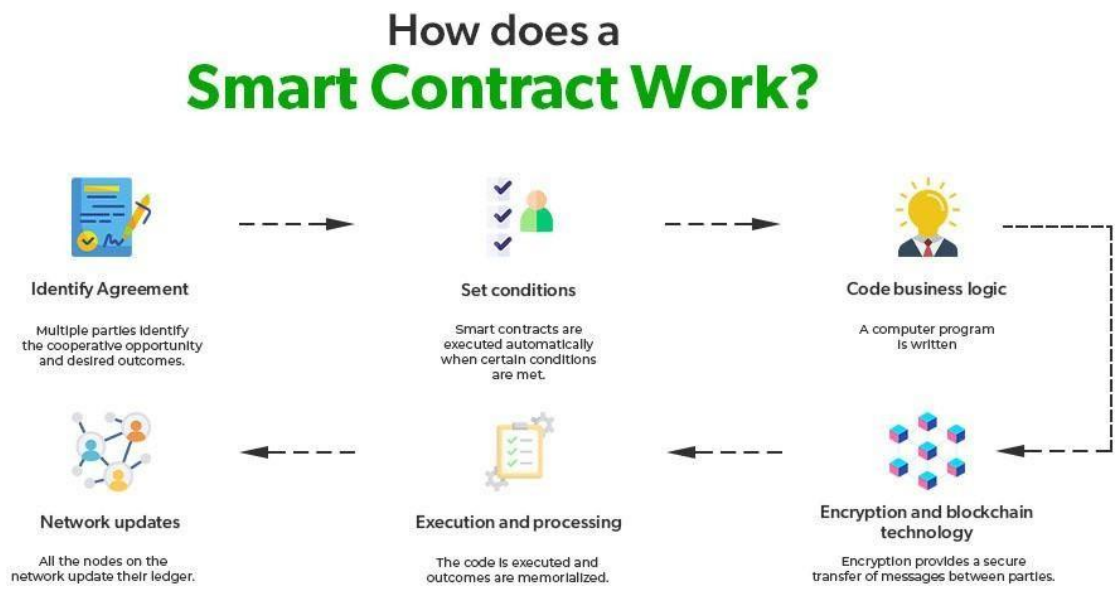
1. Basic knowledge of smart contract
2. Remix IDE
3. Basic knowledge of solidity

Contents for Theory:

1. Smart contract
2. Remix IDE
3. Solidity Basics
4. Ether transaction
5.

Introduction to Smart Contract

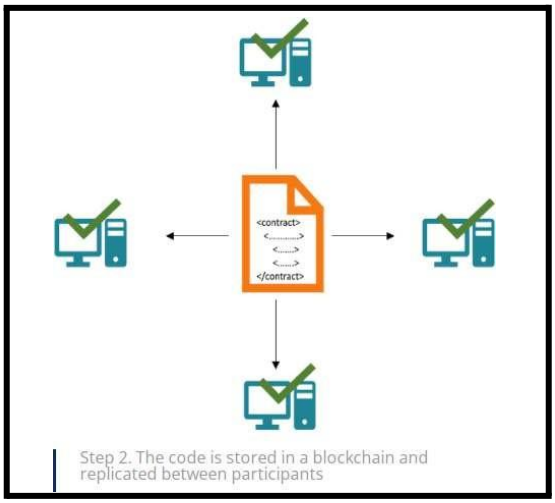
- Smart contracts are self-executing lines of code with the terms of an agreement between buyer and seller automatically verified and executed via a computer network.
- Nick Szabo, an American computer scientist who invented a virtual currency called "Bit Gold" in 1998,1 defined smart contracts as computerized transaction protocols that execute terms of a contract.2
- Smart contracts deployed to blockchains render transactions traceable, transparent, and irreversible.



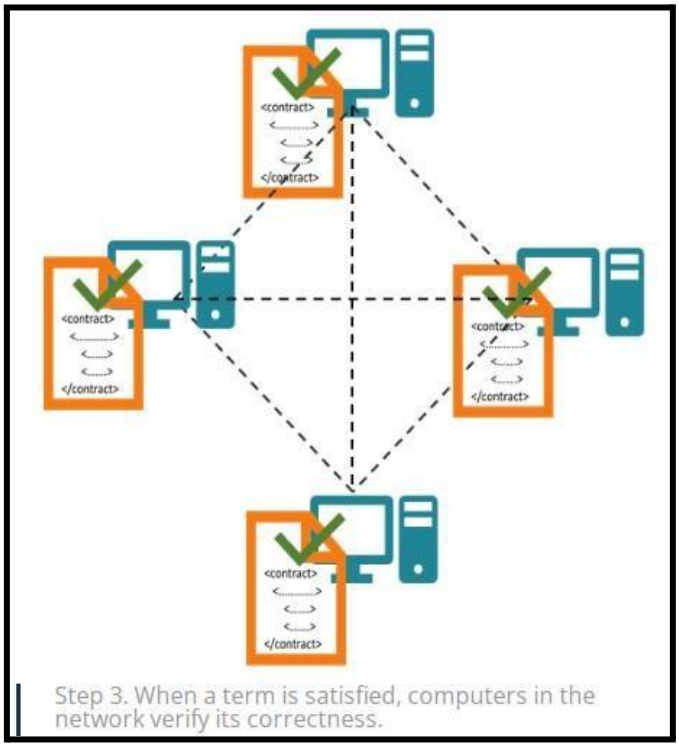
How does Smart Contract work?

- First, the contractual parties should determine the terms of the contract. After the contractual terms are finalized, they are translated into programming code. Basically, the code represents a number of different conditional statements that describe the possible scenarios of a future transaction.

- When the code is created, it is stored in the blockchain network and is replicated among the participants in the blockchain.



- Then, the code is run and executed by all computers in the network. If a term of the contract is satisfied and it is verified by all participants of the blockchain network, then the relevant transaction is executed.

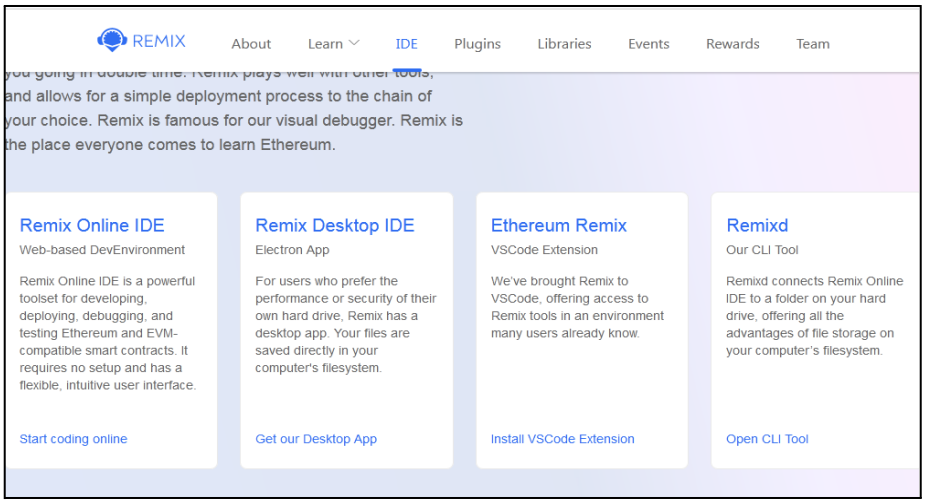


Remix IDE : Installation steps

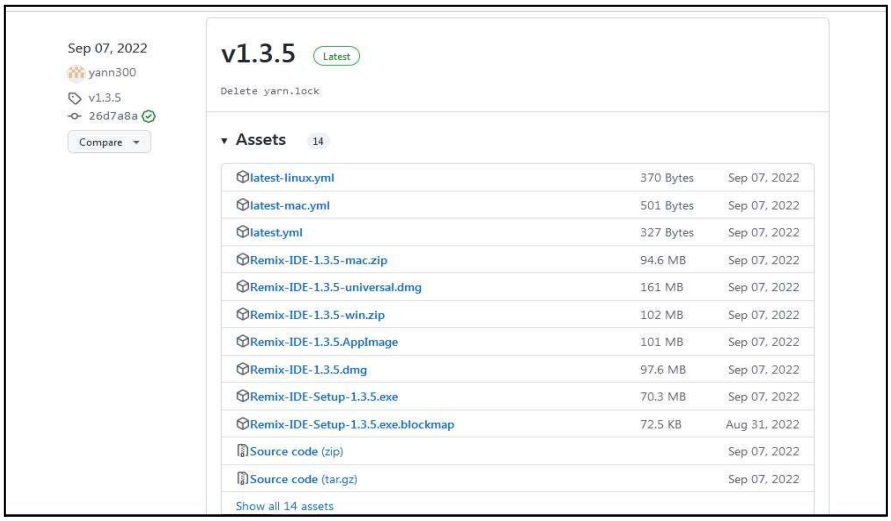
- First and foremost, head on over to the release page of the official Remix Desktop repository and grab the binary suited for your host system. This will be the .exe file for Windows users,

the .dmg for macOS and the .deb for Debian-derivative GNU/Linux systems. For Ubuntu and other AppImage setups, the .AppImage will be what you are looking for.

- Go to https://remix-project.org/
- Go to IDE option .Select Desktop IDE



- Select The file As per your choice



- Install Executable file,by double clicking on file(Applicable to .exe file Windows OS)

What is Solidity?

- Solidity is a rather simple language deliberately created for a simplistic approach to tackle real world solutions. Gavin Wood initially proposed it in August of 2014. Several developers of the Ethereum chain such as Christian Reitwiessner, Alex Beregszaszi, Liana Husikyan,

Yoichi Hirai and many more contributed to creating the language. The Solidity language can be executed on the Ethereum platform, that is a primary Virtual Machine implementing the blockchain network to develop decentralized public ledgers to create smart contract systems.

Solidity Basics

- To get started with the language and learn the basics let's dive into coding. We will begin by understanding the syntax and general data types, along with the variable data types. Solidity supports the generic value types, namely:
- Booleans: Returns value as either true or false. The logical operators returning Boolean data types are as follows:
- ! Logical negation
- && logical conjunction, "and"
- || logical disjunction, "or"
- == equality
- != inequality

Integers: Solidity supports int/unit for both signed and unsigned integers respectively. These storage allocations can be of various sizes. Keywords such as uint8 and uint256 can be used to allocate a storage size of 8 bits to 256 bits respectively. By default, the allocation is 256 bits. That is, uint and int can be used in place of uint256 and int256. The operators compatible with integer data types are:

- Comparisons: <=, <, ==, !=, >=, >. These are used to evaluate to bool.
- Bit operators: &, |, ^ bitwise exclusive 'or', ~ bitwise negation, "not".
- Arithmetic operators: +, -, unary -, unary +, *, /, % remainder, ** exponentiation, << left shift, >> right shift.

The EVM returns a Runtime Exception when the modulus operator is applied to the zero of a "divide by zero" operation.

Address: An address can hold a 20 byte value that is equivalent to the size of an Ethereum address. These address types are backed up with members that serve as the contract base.

String Literals: String literals can be represented using either single or double quotes (for example, "foo" or 'bar'). Unlike in the C language, string literals in Solidity do imply trailing value zeroes. For instance, “bar” will represent a three byte element instead of four. Similarly, in the case of integer literals, the literals are convertible inherently using the corresponding fit, that is, byte or string.

Modifier: In a smart contract, modifiers are used to ensure the coherence of the conditions defined before executing the code.

Solidity provides basic arrays, enums, operators, and hash values to create a data structure known as **“mappings**.” These mappings are used to return values associated with a given storage location. An Array is a contiguous memory allocation of a size defined by the programmer where if the size is initialized as K, and the type of element is instantiated as T, the array can be written as T[k].

Arrays can also be dynamically instantiated using the notation uint[][6]. Here the notation initializes a dynamic array with six contiguous memory allocations. Similarly, a two dimensional array can be initialized as arr[2][4], where the two indices point towards the dimensions of the matrix.

We will begin our programming venture with a simple structure of a contract. Consider the following code:

```
pragma solidity^0.4.0;
contract StorageBasic {
  uint storedValue;
  function set(uint var) {
     storedValue= var;
  }
  function get() constant returns (uint) {
   return storedValue;
  }
}
```

- The word “**Pragma**” refers to the instructions given to a compiler to sequentially execute the source code.
- Solidity is statically typed language. Therefore, each variable type irrespective of their scope can be instantiated at compile time. These elementary types can be further combined to create

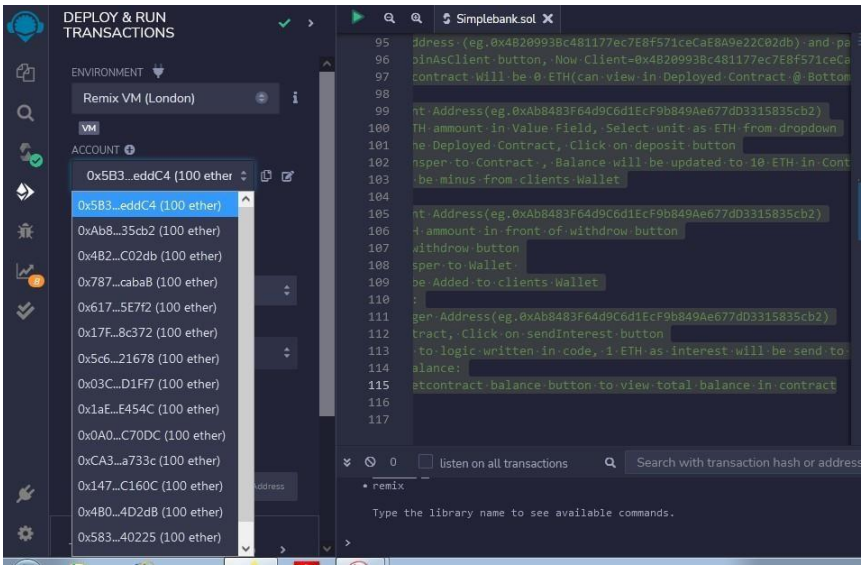
complex data types. These complex data types then synchronize with each other according to their respective preferences.

Steps to Run Banking Application

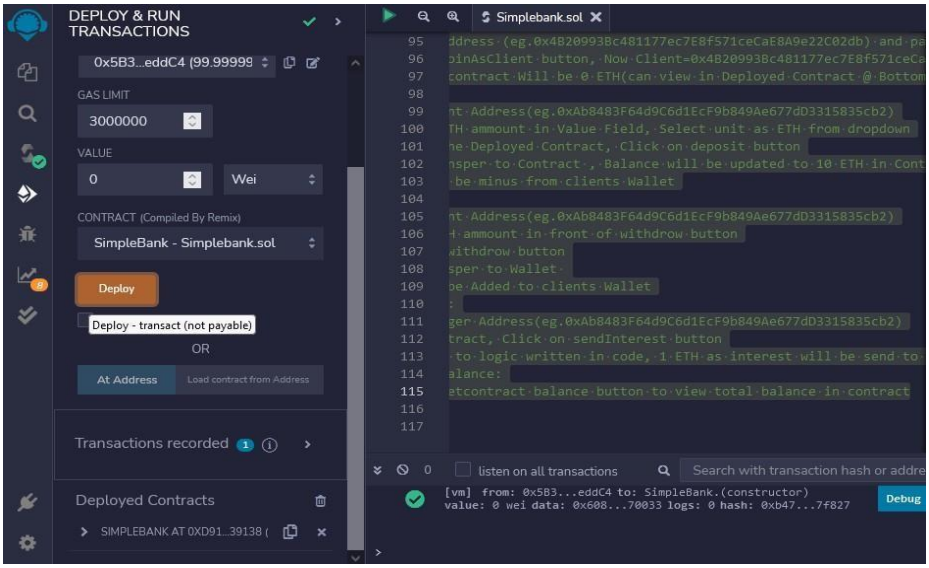
1.//Output steps

//Initially All Account has 100 fake ether

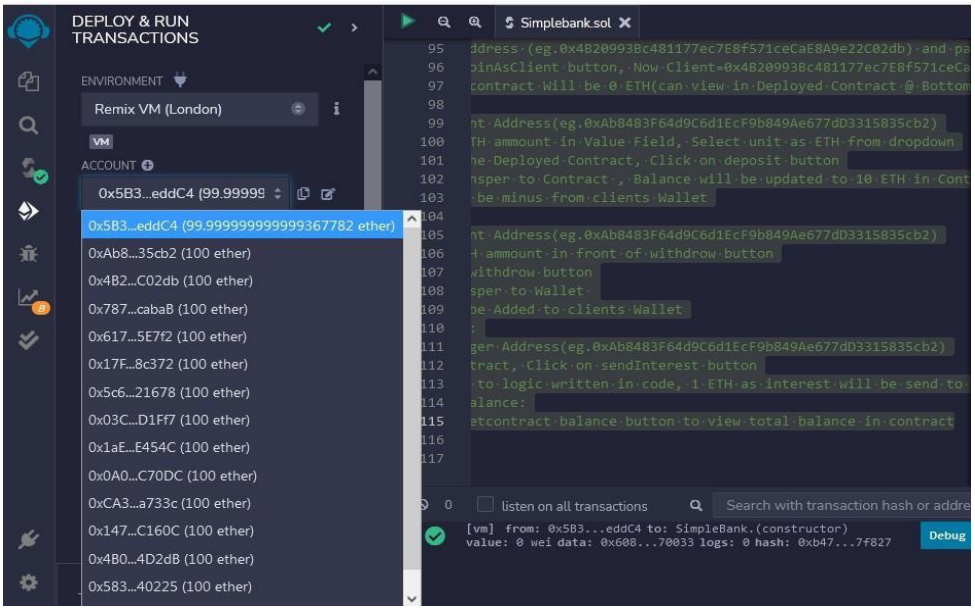
//Step 1: Select first Address (eg.0x5B38Da6a701c568545dCfcB03FcB875f56beddC4)



//Step 2: Click on Deploy button(Contract Created,Can view under Deployed Contract)



//After deploying contract 100 ETH turns to 99.99999. ....ETH

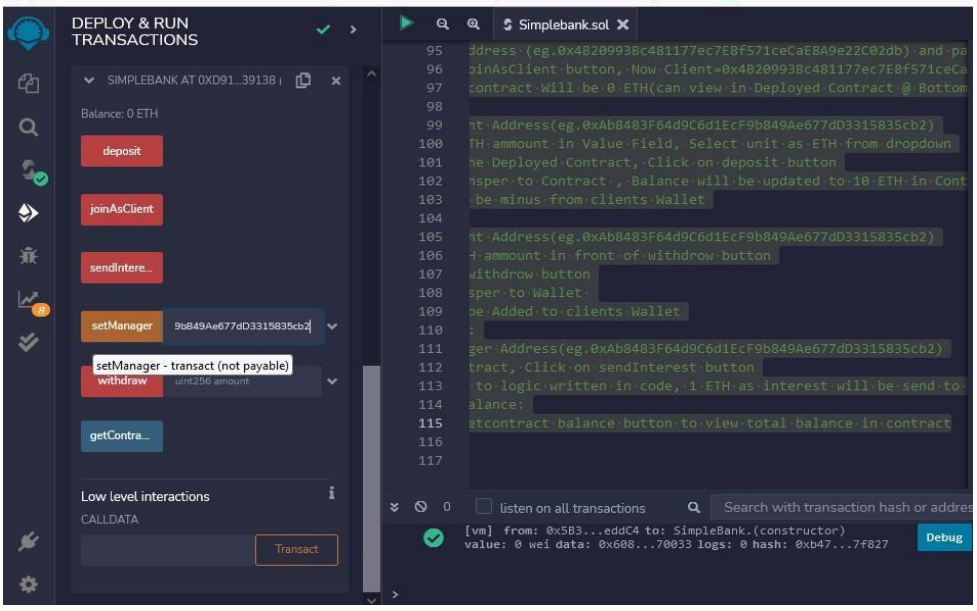


//Step 3: Set Manager: Follow Following instructions

// i.Select Onother Address(eg.0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2)

// ii.Copy this address (eg.0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2) and paste it in contract, infront of set Manager button

// iii. click on set manager button, Now Manager=0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2



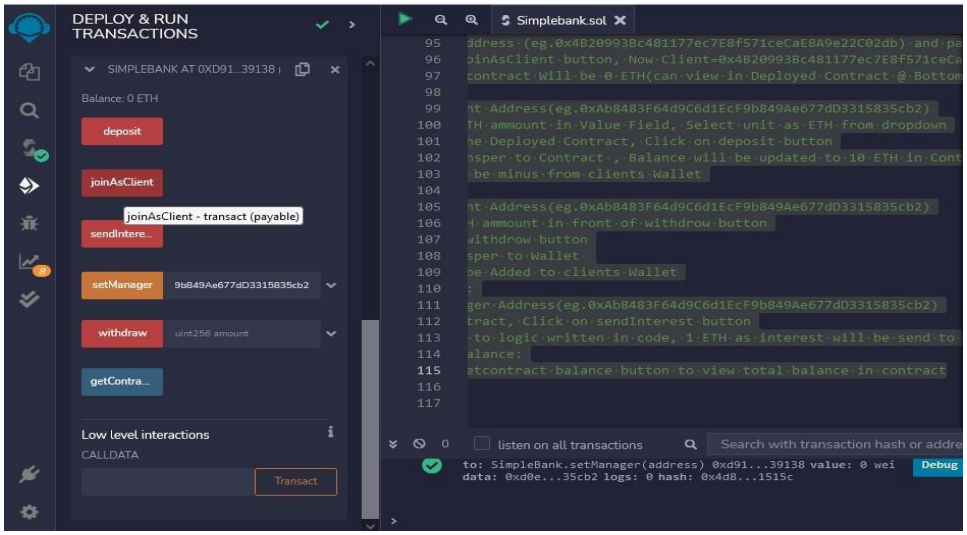
//Step 4: join as Client: Follow Following instructions

// i.Select Onother Address(eg.0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db)

// ii.Copy this address (eg.0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db) and paste it in

contract, infront of joinAsClient button

// iii.click on joinAsClient button, Now Client=0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db



//Initially Balance in contract Will be 0 ETH(can view in Deployed Contract @ Bottom)

//Step 5: Deposit:

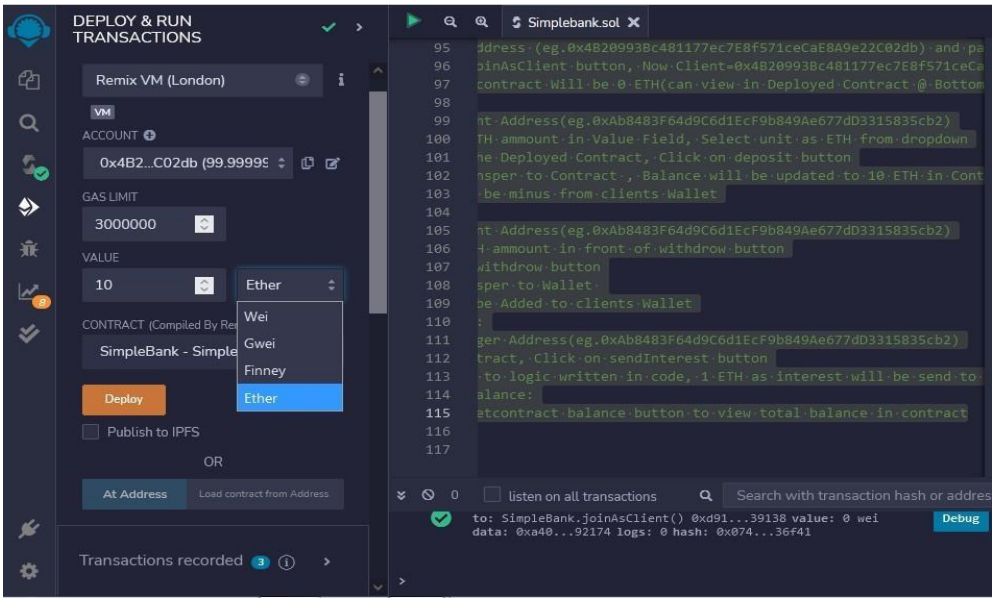
// i.Select Client Address(eg.0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2)

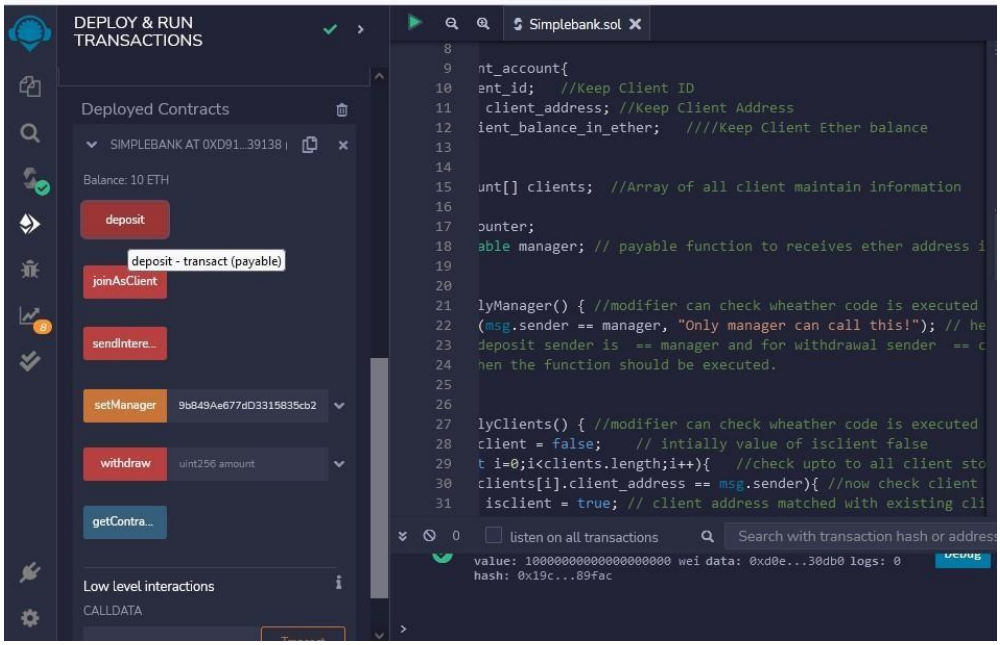
// ii. Enter 10 ETH ammount in Value Field, Select unit as ETH from dropdown

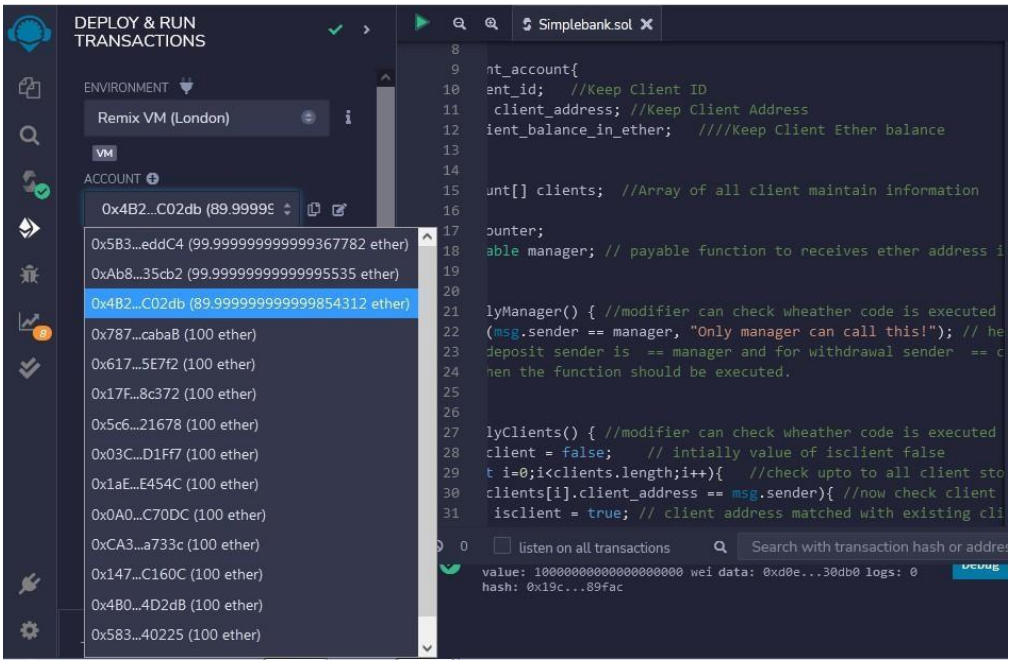
// iii. Come to the Deployed Contract, Click on deposit button

// iv. 10 ETH transper to Contract , Balance will be updated to 10 ETH in Contract

// V. 10 ETH will be minus from clients Wallet







//Step 6: Withdraw:

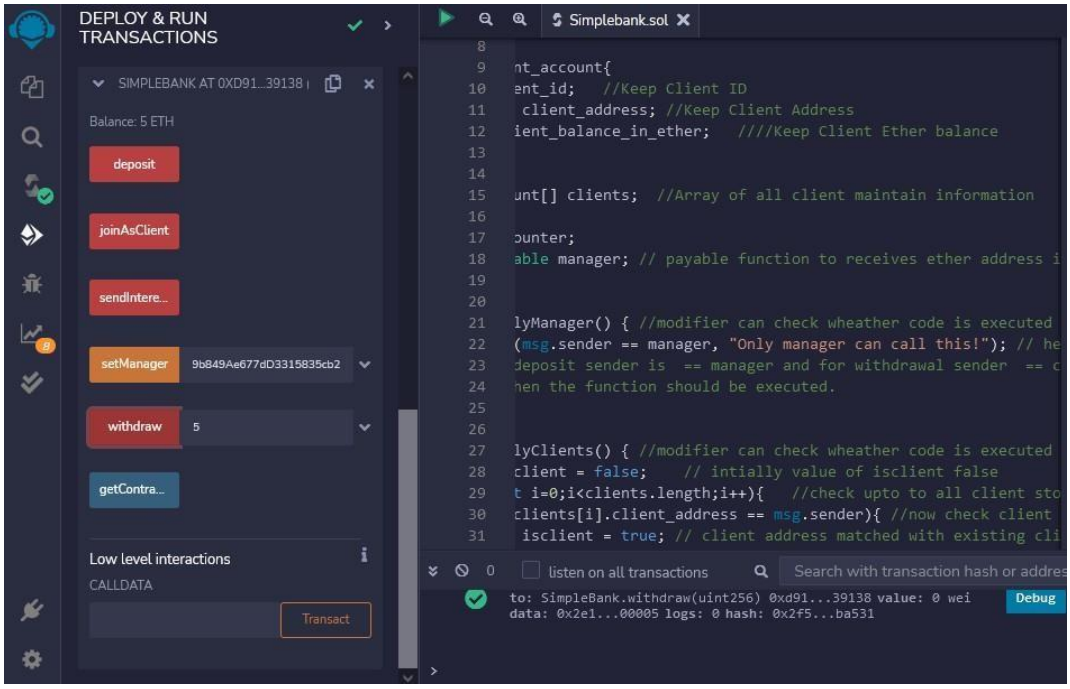
// i.Select Client Address(eg.0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2)

// ii. Enter 5 ETH ammount in front of withdrow button

// iii. Click on withdrow button

// iv. 5 ETH transper to Wallet

// V. 5 ETH will be Added to clients Wallet

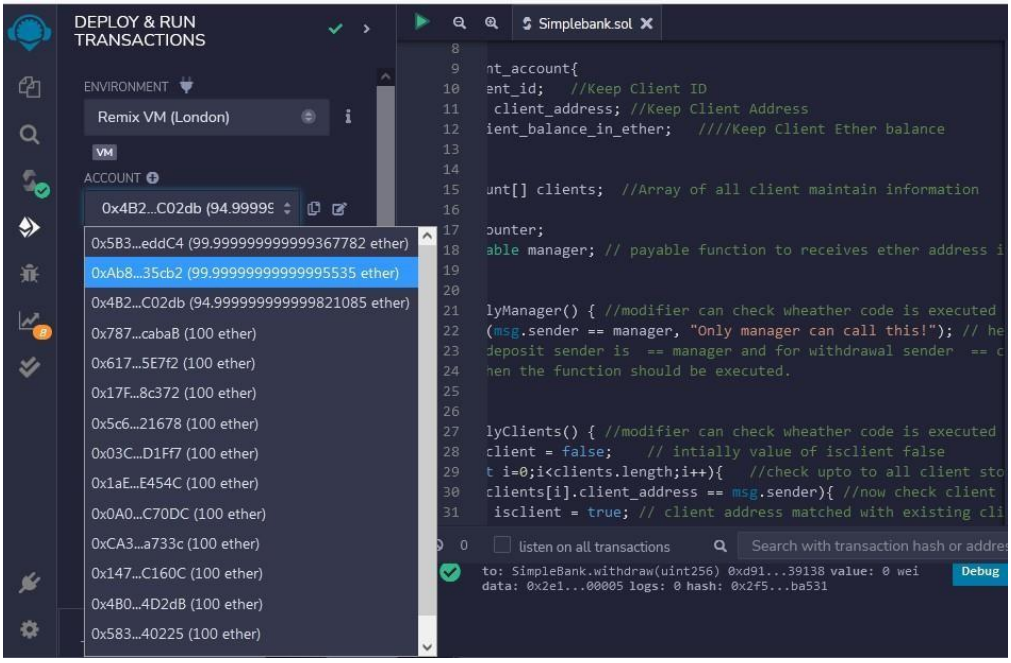


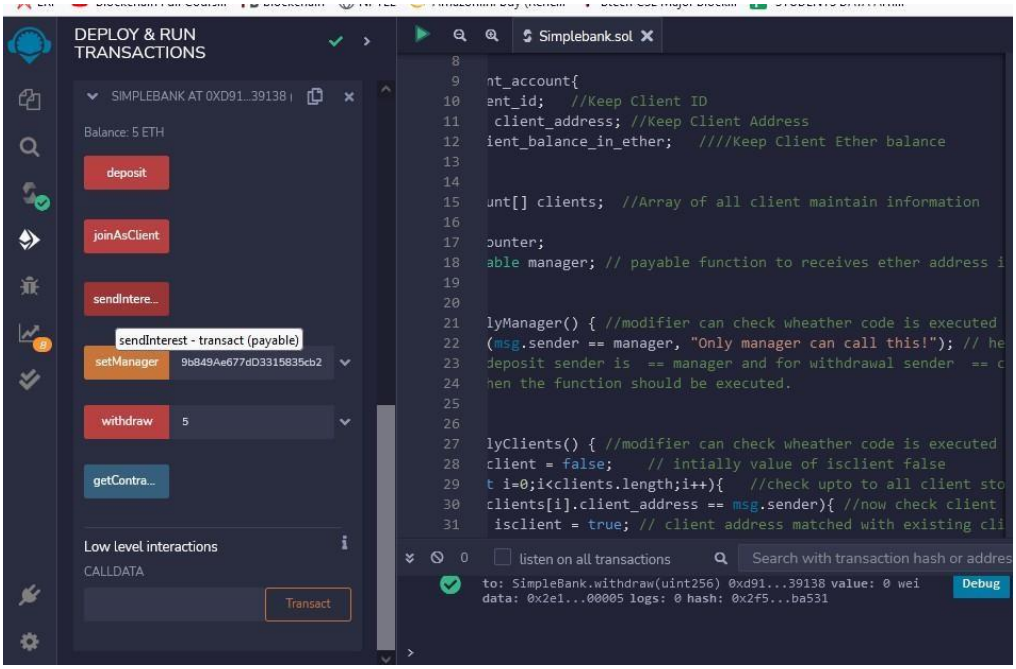
//Step 7: Send Interest:

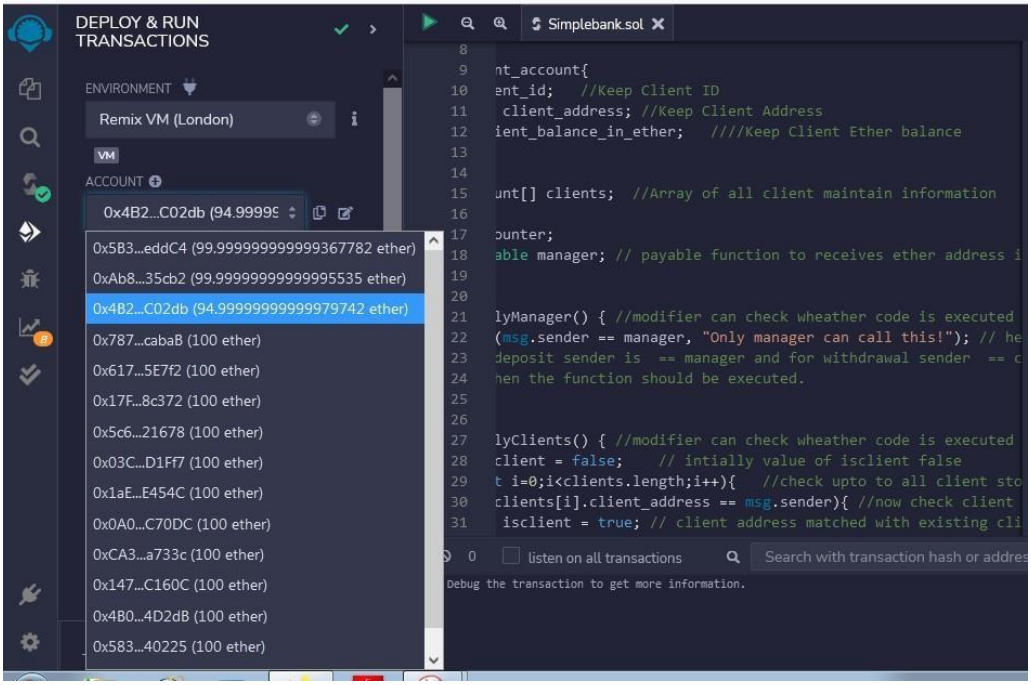
// i.Select Manager Address(eg.0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2)

// ii.Come to contract, Click on sendInterest button

// iii. According to logic written in code, 1 ETH as interest will be send to Client Wallet

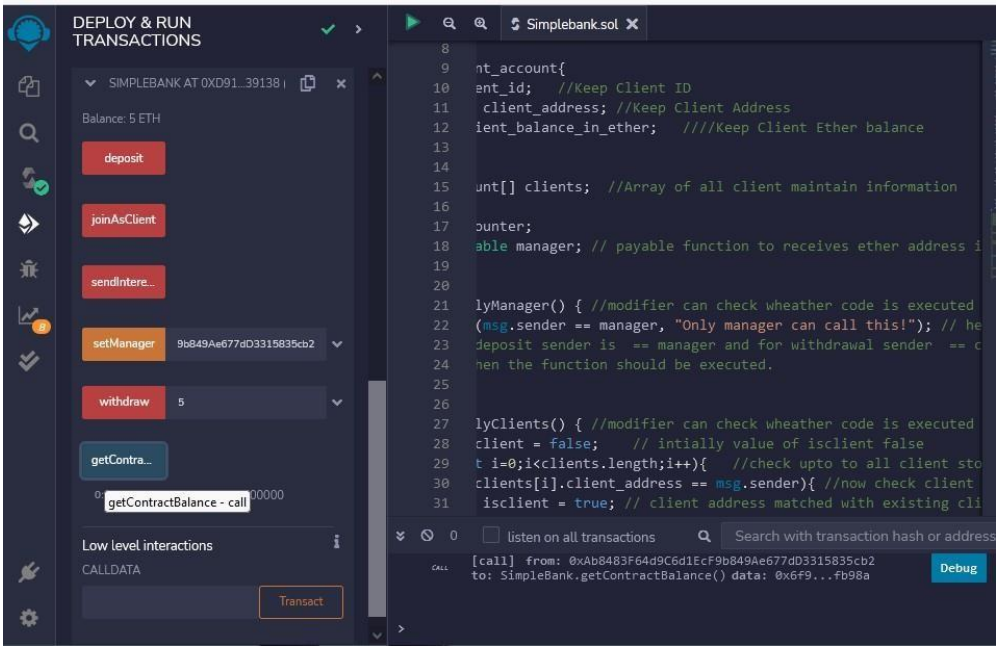






//Step 8: getContract Balance:

// i.Click on get contract balance button to view total balance in contract



Conclusion-In this way we have explored Concept u s i n g smart contract on a test network, and mapped with CO6

Group C

Assignment No: 4

Title of the Assignment: 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.

Objective of the Assignment: Students should be able to learn about Solidity. Its datatypes and implementations.

Prerequisite:

1. Basic Programming Logic
2. Basic knowledge of Solidity

Contents for Theory:

1. Solidity - Arrays

2. Solidity - Structures

3. Solidity – Fallback

4. Implementation

1. **Solidity – Arrays :**

Arrays are data structures that store the fixed collection of elements of the same data types in which each and every element has a specific location called index. Instead of creating numerous individual variables of the same type, we just declare one array of the required size and store the elements in the array and can be accessed using the index. In Solidity, an array can be of fixed size or dynamic size. Arrays have a continuous memory location, where the lowest index corresponds to the first element while the highest represents the last.

Creating an Array

To declare an array in Solidity, the data type of the elements and the number of elements should be specified. The size of the array must be a positive integer and data type should be a valid Solidity type

Syntax:

<data type> <array name>[size] = <initialization>

Fixed-size Arrays :

The size of the array should be predefined. The total number of elements should not exceed the size of the array. If the size of the array is not specified then the array of enough size is created which is enough to hold the initialization.

Dynamic Array:

The size of the array is not predefined when it is declared. As the elements are added the size of array changes and at the runtime, the size of the array will be determined.

Array Operations :

- **Accessing Array Elements:**
  The elements of the array are accessed by using the index. If you want to access ith element then you have to access (i-1)th index.

- **Length of Array:**

  Length of the array is used to check the number of elements present in an array. The size of the memory array is fixed when they are declared, while in case the dynamic array is defined at runtime so for manipulation length is required.

- **Push:**

  Push is used when a new element is to be added in a dynamic array. The new element is always added at the last position of the array.

- **Pop:**

  Pop is used when the last element of the array is to be removed in any dynamic array.

- **Solidity – Structures:**

  Structs in Solidity allows you to create more complicated data types that have multiple properties. You can define your own type by creating a **struct**.
  They are useful for grouping together related data.

  Structs can be declared outside of a contract and imported in another contract. Generally, it is used to represent a record. To define a structure *struct* keyword is used, which creates a new data type.

  Syntax:

```
struct <structure_name> {
   <data type> variable_1;
   <data type> variable_2;
}
```

For accessing any element of the structure, 'dot operator' is used, which separates the struct variable and the element we wish to access. To define the variable of structure data type structure name is used.

2. Solidity – Fallback :

The solidity fallback function is executed if none of the other functions match the function identifier or no data was provided with the function call. Only one unnamed function can be assigned to a contract and it is executed whenever the contract receives plain Ether without any data. To receive Ether and add it to the total balance of the contract,

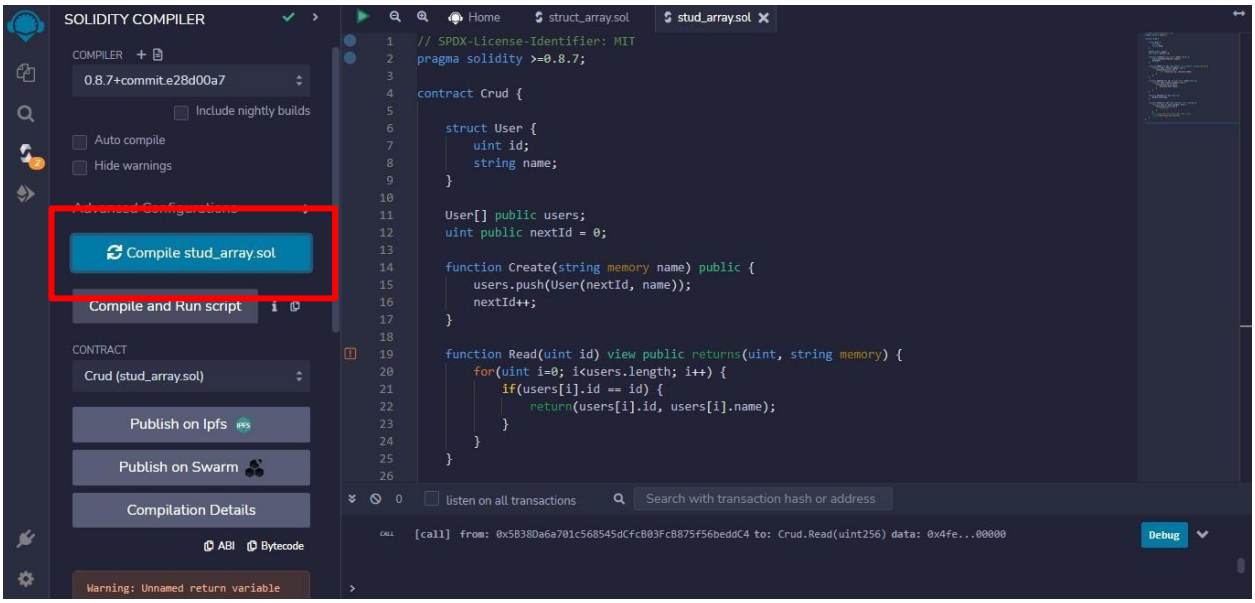
the fallback function must be marked payable. If no such function exists, the contract cannot receive Ether through regular transactions and will throw an exception.

Properties of a fallback function:

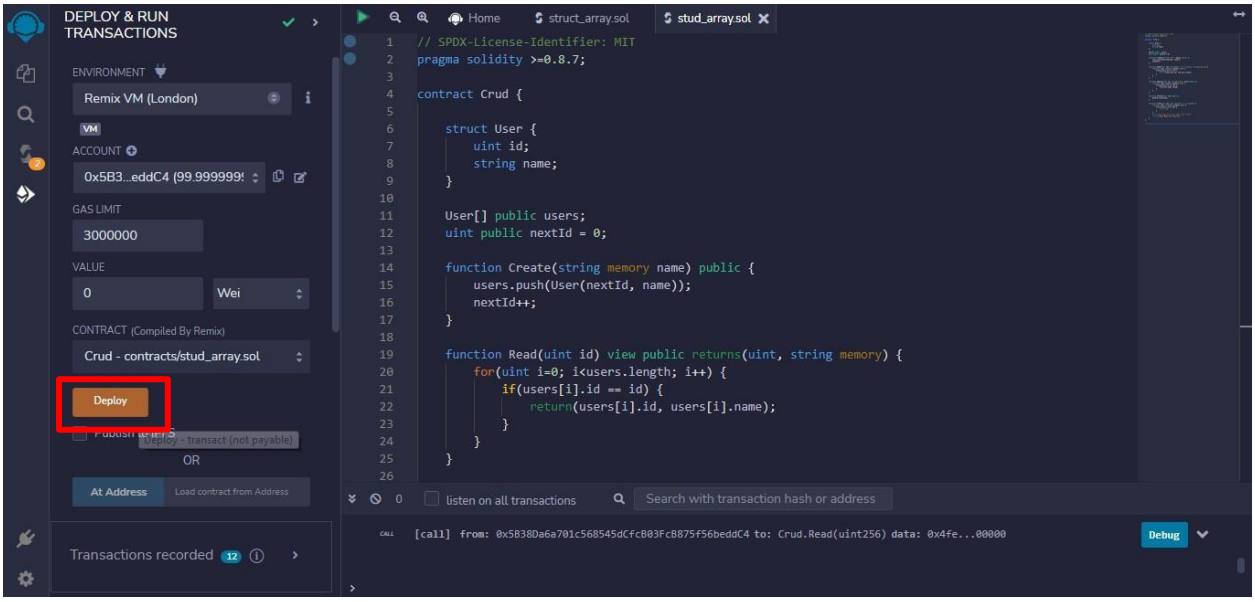
1. Has no name or arguments.
2. If it is not marked **payable**, the contract will throw an exception if it receives plain ether without data.
3. Cannot return anything.
4. Can be defined once per contract.
5. It is also executed if the caller meant to call a function that is not available
6. It is mandatory to mark it external.
7. It is limited to 2300 gas when called by another function. It is so for as to make this function call as cheap as possible.

Output –

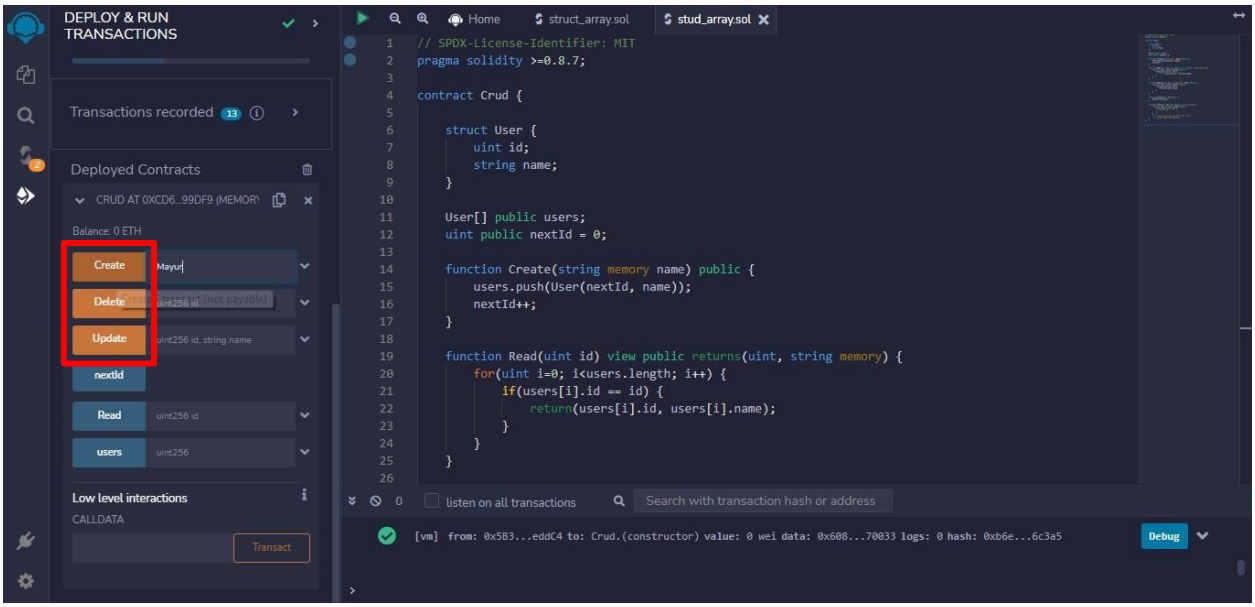
Step 1 – Compile the program by clicking on compile button.



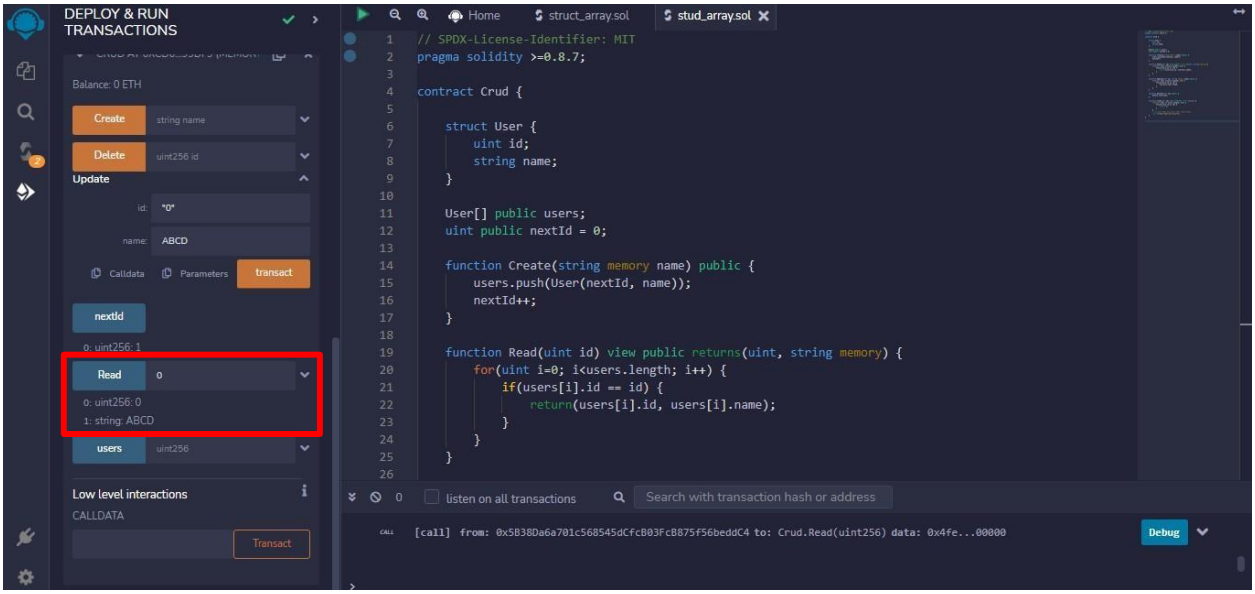
Step 2 – After the Successful compilation, Deploy the contract to see the output.


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (London)
VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT (Compiled By Remix)
Crud - contracts/stud_array.sol
Deploy
OR
At Address
Load contract from Address
Transactions recorded 12
Home
struct_array.sol
stud_array.sol
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.7;
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);
}
}
}
listen on all transactions
Search with transaction hash or address
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Crud.Read(uint256) data: 0x4fe...00000
Debug

Step 3 - Now you can check the output. You can insert, update and delete the student data using your smart contract.



Step 4 – After entering your data, you can read the data using ID.



Conclusion-

In this way we have created array, structure and used fallback function in solidity and mapped with CO6.

Group C

Assignment No: 5

Title of the Assignment: Write a survey report on types of Blockchains and its real time use cases.

Objective of the Assignment: Students should be able to learn about different use cases/ real timeapplication of Blockchain and perform survey on one of the case study.

Contents for Theory:

A blockchain is a digital ledger of all cryptocurrency transactions. It is constantly growing as "completed" blocks are added to it with a new set of recordings. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. Bitcoin nodes use the block chain to differentiate legitimate Bitcoin transactions from attempts to re-spend coins that have already been spent elsewhere.

Types of Blockchain

There are three types of blockchains- public, private and consortium.

1. **Public Blockchain:** A public blockchain has absolutely no access restrictions. Anyone with an internet connection can send transactions to it as well as become a validator (i.e. participate in the consensus process). Bitcoin is the best example of a public blockchain.
2. **Private Blockchain:** A private blockchain is a little more centralized. Here, a central authority controls who can access the network and who can become a validator. Validators on a private blockchain are typically vetted by the central authority. Permissioned blockchains are often used in enterprise settings where centralized control is necessary. An example of a private blockchain is Hyperledger Fabric.
3. **Consortium Blockchain:** A consortium blockchain is a hybrid of the public and private blockchain. Here, a group of companies or organizations control who can access the network and who can become a validator. The selection of validators is typically done through a voting process. Consortium blockchains are often used in cases where multiple parties need to collaborate but no party is fully trusted. An example of a consortium blockchain is the R3 Corda platform.

Real-Time

1. **Supply Chain Management** Blockchain can be used to create an immutable record of all the transactions in a supply chain. This can help to increase transparency and traceability in the supply chain.
2. **Identity Management** Blockchain can be used to create a digital identity for individuals, organizations, and devices. This can be used for KYC (know your customer) and AML (anti-money laundering) compliance.
3. **Payments** Blockchain can be used to process payments. This can be done using cryptocurrencies or fiat currencies.
4. **Data Management** Blockchain can be used to store data in a tamper-proof and decentralized manner. This can be used for data sharing and data security.
5. **IoT** Blockchain can be used to create a decentralized network of IoT devices. This can be used for data sharing and data security.
6. **Predictive Analytics** Blockchain can be used to create a decentralized network of predictive analytics models. This can be used for data sharing and data security.

Conclusion: Hence, we have studied to write a survey report on types of Blockchains and its real time application and mapped with CO6.

Group C

Assignment No: 6

Mini Project

Points Expected in Report:

1. Introduction of topic
2. Literature survey
3. System Architecture
4. Working/Algorithm
5. Result/Output
6. Conclusion
7. References