Group A: Assignment No. 1

Title: Fibonacci Series

Objective: To find time and space complexity of fibonacci series algorithm.

Problem Statement: Write a non-recursive and recursive program to calculate fibonacci numbers and analyze their time and space complexity.

Software and Hardware Requirements:
1) Desktop/Laptop
2) Any Operating System
3) IDE

Theory:

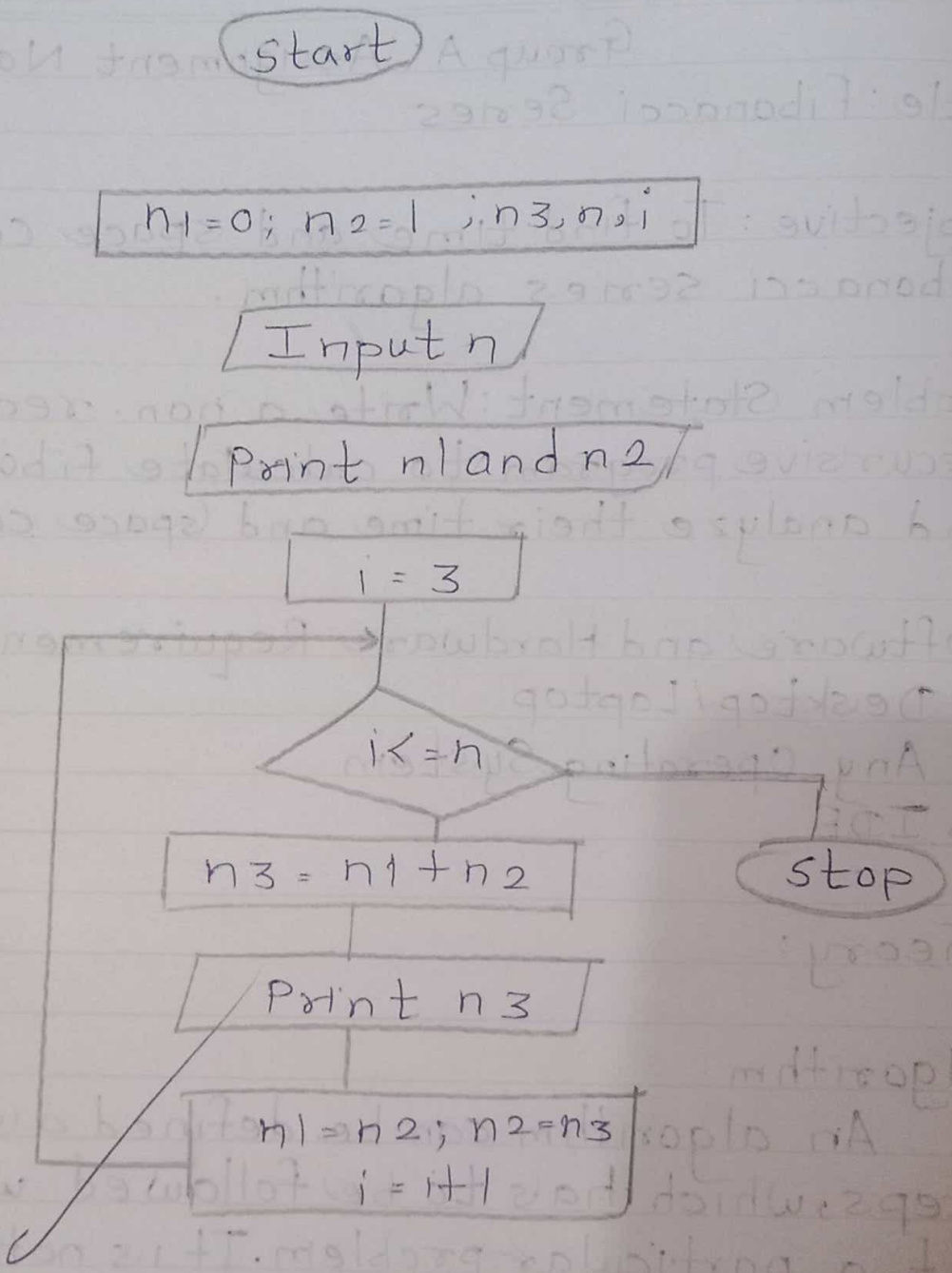
Algorithm:

An algorithm can be defined as finite set of steps, which has to be followed while carrying out a particular problem. It is nothing but a process of executing actions step by step. An algorithm is a distinct computational procedure that takes input as a set of values and results in the output as a set of value by solving the problem.

Asymptotic Notations:

Asymptotic notations are the mathematical notations used to describe the running time of an algorithm. when the input tends towards a particular value or a limiting value.

• Flowchart of fibonacci



For eg: In bubble sort, when the input array is sorted the time taken by the algorithm is linear i.e the best case when in reverse order the time taken is quadratic. When nather sorted nor reverse it is average.

There are mainly three Asymptatic Notations:

1) Big-O Notation (O):
Big-O Notation represents the upper bound of the running time of an algorithm. Thus it gives the worst case complexity of an algorithm.

2) Omega Notation (Ω)
Omega notation represents the lower bound of the running time of an algorithm. Thus it provides the best one case complexity of an algorithm.

3) Theta Notation (Θ):
Theta notation encloses the function from above and below. Since it represents the upper and lower bound of the running time of an algorithm it is used for analyzing the average case complexity of an algorithm.

Algorithm:
1) Iterative Approach
Step 1: Start
step 2: Declare variable i, f1, f2, f3.

- Analysis

Two cases (i) $n = 0$ or 1 and (j) $n > 1$

1) When $n = 0$ or 1 line 4 and 5 gext extended once each.

2) when $n > 1$, lines 4, 8, 14 are extended once line 9 gets executed n times, and times 11 and 12 get extented n-1 time each.

Step 3: Initialize the variables.
f1=0, f2=1, f3=0.
Step 4: Enter the number of terms of fibonacci series to be printed.
Step 5: Print first two terms of series.
f1 and f2.
Step 6: Use loop for following steps.
f3 = f1+f2
f1 = f2
f2 = f3
Increase value of i each time by 1.
Print the value of f3.
Step 7: Stop.

2) Recursive Approach.

```
procedure Recursive_Fibo(n)
    int f0, f1
    f0 = 0
    f1 = 1
    if(n<=1):
        return 1
    return Recursive_fibo(n-1)+Recursive_fibo(n-2)
End Recursive_fibo.
```

Recursive:
Recursive is the process which comes into existence when a function calls a copy of itself to work on a smaller problem. Any function which calls itself is called recursive function and such function calls are called recursive calls.

Page No.
Date

Test Cases:

1) nth term of Fibonacci Series : 5
→ 3

2) Enter term of Fibonacci Series: 7
→ 8

3) Enter term of Fibonacci Series: 10
→ 55

4) Enter term of Fibonacci Series: 1
→ 0

Mathematical Model:

We can calculate the fibonacci Series by using the following formula.

$$f(n) = f(n-1) + F(n-2)$$

Above formula is used to find fibonacci series by using Recursion.

Time Complexity:

The time complexity of an algorithm quantifies the amount of time taken by an algorithm to run as function of length of the input.

Space Complexity:

The space complexity of an algorithm quantifies that the amount of space taken by an algorithm to run as a function of length of the

For Iterative Method:
1) Time Complexity is T(N) i.e Linear
2) Space Complexity is O(N).

For Recursive Method:
1) Time Complexity is T(2^N)
2) Space Complexity is O(N).

Conclusion: Successfully implemented program for fibonacci series using iterative and recursive approach and analyzed the time and space complexity for both approaches.

A

19/12

Group A - Assignment No. 2

Title: Huffman Encoding

Objective: To understand and implement huffman encoding using a greedy method.

Problem Statement: Write a program to implement Huffman encoding using a greedy strategy

Software and Hardware Requirement:

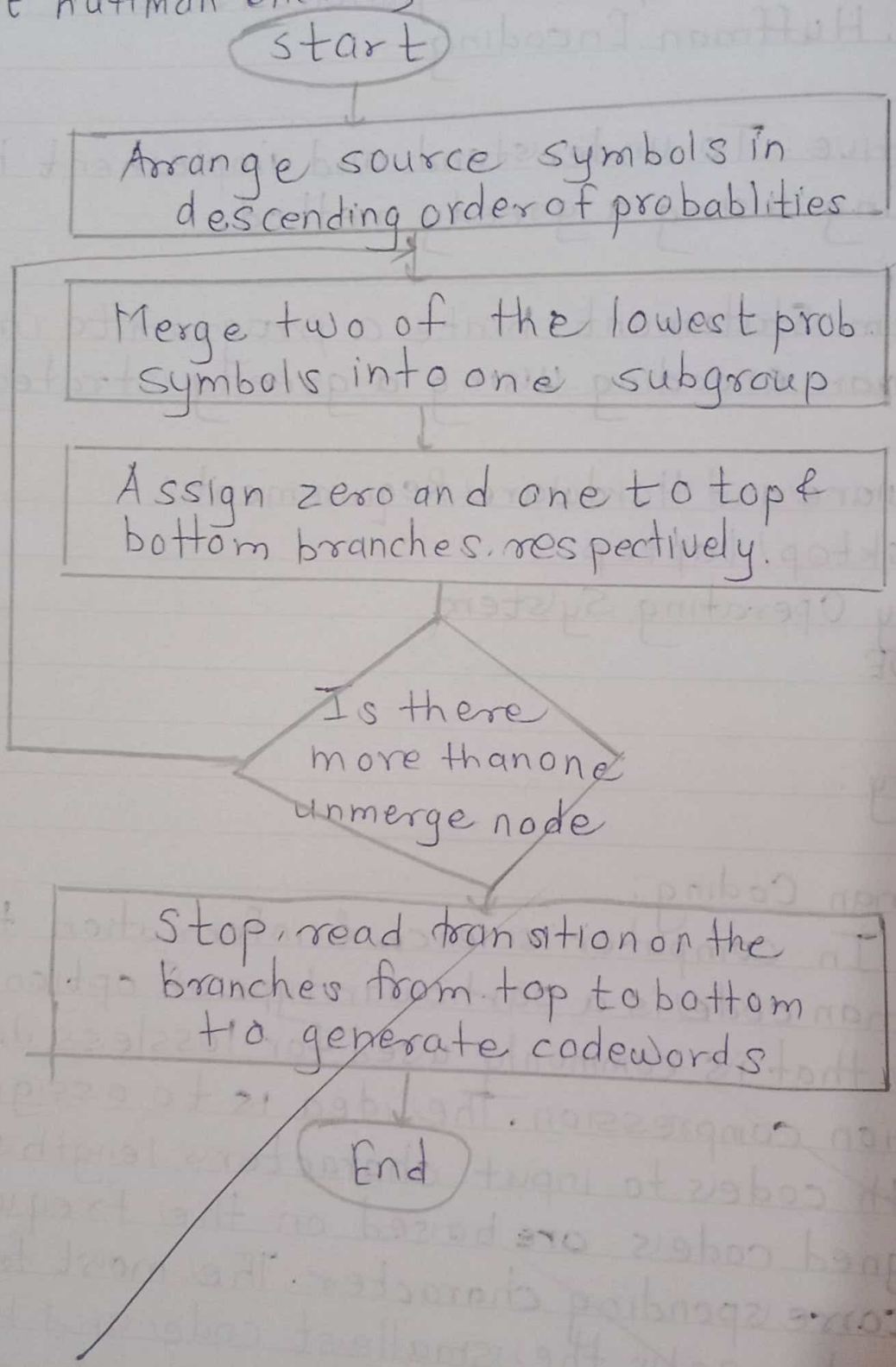
1) Desktop/ laptop
2) Any Operating System
3) IDE

Theory:

Huffman Coding:

In computer science & information theory, a huffman code is a particular type of optional prefix code that is commonly used for lossless data communication compression. The idea is to assign variable-length codes to input characters lengths of the assigned codes are based on the frequencies of corresponding character. The most frequent character gets the smallest code and the least frequent character gets the largest code. The variable length codes assigned to input characters are Prefix codes means the codes are assigned to input in such a way that the code assigned to one character is not the prefix of code assigned to any other character.

• Flowchart Huffman encoding



There are mainly two major parts in Huffman coding
1) Build a Huffman tree from input characters.
2) Traverse the Huffman tree and assign code to characters.

Greedy Approach:
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.
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 global best result. Greedy algorithm are easy to describe.

Algorithm:
Step 1: Create a leaf node for each unique character and buid a min heap of all leaf nodes.

Step 2: Extract two nodes with the minimum frequency from the min heap.

Step 3: Create new internal node with a frequency equal to the sum of the two nodes frequencies.

Step 4: Make the first extracted node as its left child and the other extracted node as its right child. Add this node to the min heap.

Step 4: Repeat step 2 and step 3 until the heap contains only one node. The remaining node is the root node and the tree is complete.

Code:

Create a priority queue Q consisting of each unique character.
Sort then in ascending order of their frequencies for all the unique characters.
    Create a newnode.
    extract minimum value from Q and assign it to left child of newNode.
    extract minimum value from Q and assign it to rightChild of newNode.
    Calculate the sum of these two minimum values and assign it to the value newNode.
    Insert this newNode into the tree.
return rootNode.

Conclusion: We successfully understood the concept of Huffman encoding and implement it using greedy method.

20/9/22 (A)

Group A - Assignment No-3

Title: Knapsack Problem

Objective: To understand and implement knapsack problem using a greedy method.

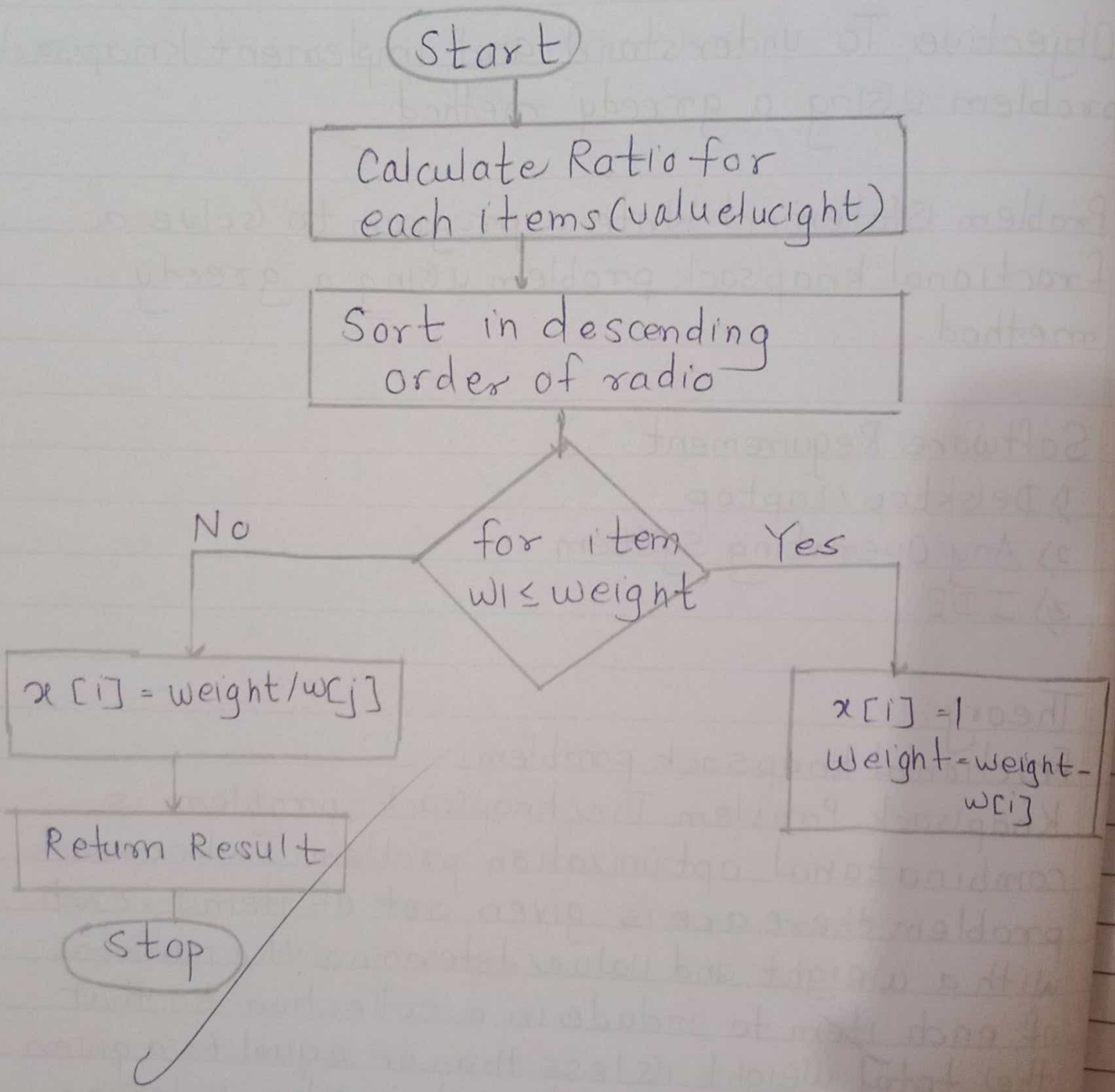
Problem Statement: Write a program to solve a fractional knapsack problem using a greedy method.

Software Requirement:
1) Desktop / Laptop
2) Any Operating System
3) IDE

Theory:
Fractional knapsack problem:
Knapsack Problem: The knapsack problem is combinatorial optimization problem. In knapsack problem there are is given set of items, each with a weight and value determine the numbers of each item to include in a collection so that the total weight is less than or equal to a given limit data and the total number value is large as possible.

The fractional knapsack problem is also one of the techniques which are used to solve the knapsack problem. In fractional knapsack, the internet are broken in order to maximize the profit.

- Flowchart of knapsack



Page No.
Date

Greedy Approach to solve fractional knapsack Problem:

In greedy approach we calculate the ratio of profit/weight and accordingly, we will select the item. The item with the highest ratio would be selected first.

There are basically three approaches to solve the problem:

1) First approach is to select the item based on the maximum profit.
2) Second approach is to select the item based on the minimum weight.
3) Third approach is to calculate the ratio of profit/weight.

Algorithm:

Greedy Fractional knapsack:

```
for i=1 to n
    do x[i]=0

weight=0
for i=1 to n
if weight w[i]<w then
    x[i]=1
  weight=weight+w[i]
else
    x[i]=(w-weight)/w[i]
```

weight = w
break
return x

Analysis:
If the provided items are already sorted into a decreasing order of $\frac{Pi}{Wi}$ then the

while loop takes a time is O(n).

Therefor, the total time including the sort is in O(n log n).

Test Cases:

1) Enter the number of items: 3
enter values: 24 15 25
enter weights: 15 10 18
Maximum Capacity: 20

Maximum value of items that can be 3.15

2) No of Items: 4
Enter values: 5 12 8 4
Enter weights: 12 15 8 20
Enter capacity: 25

Maximum value of Items that can be carried is 27.

The fraction in which items should be taken [1, 0, 33, 1, 0].

Mathematical Model:

$$\text{Ratio} = \frac{\text{Value}}{\text{Weight}}$$

Conclusion: We have successfully implemented fractional knapsack problem using greedy approach.

29/09/22 (A)

Group A : Assignment No: 4

Title: 0-1 knapsack Problem

Objective: To Understand and implement 0-1 knapsack problem using dynamic programming

Problem Statement: Write a program to solve a 0-1 knapsack problem using dynamic programming or branch and bound strategy.

Software Requirements:
1) Desktop/Laptop
2) Any Operating System
3) IDE

Theory:

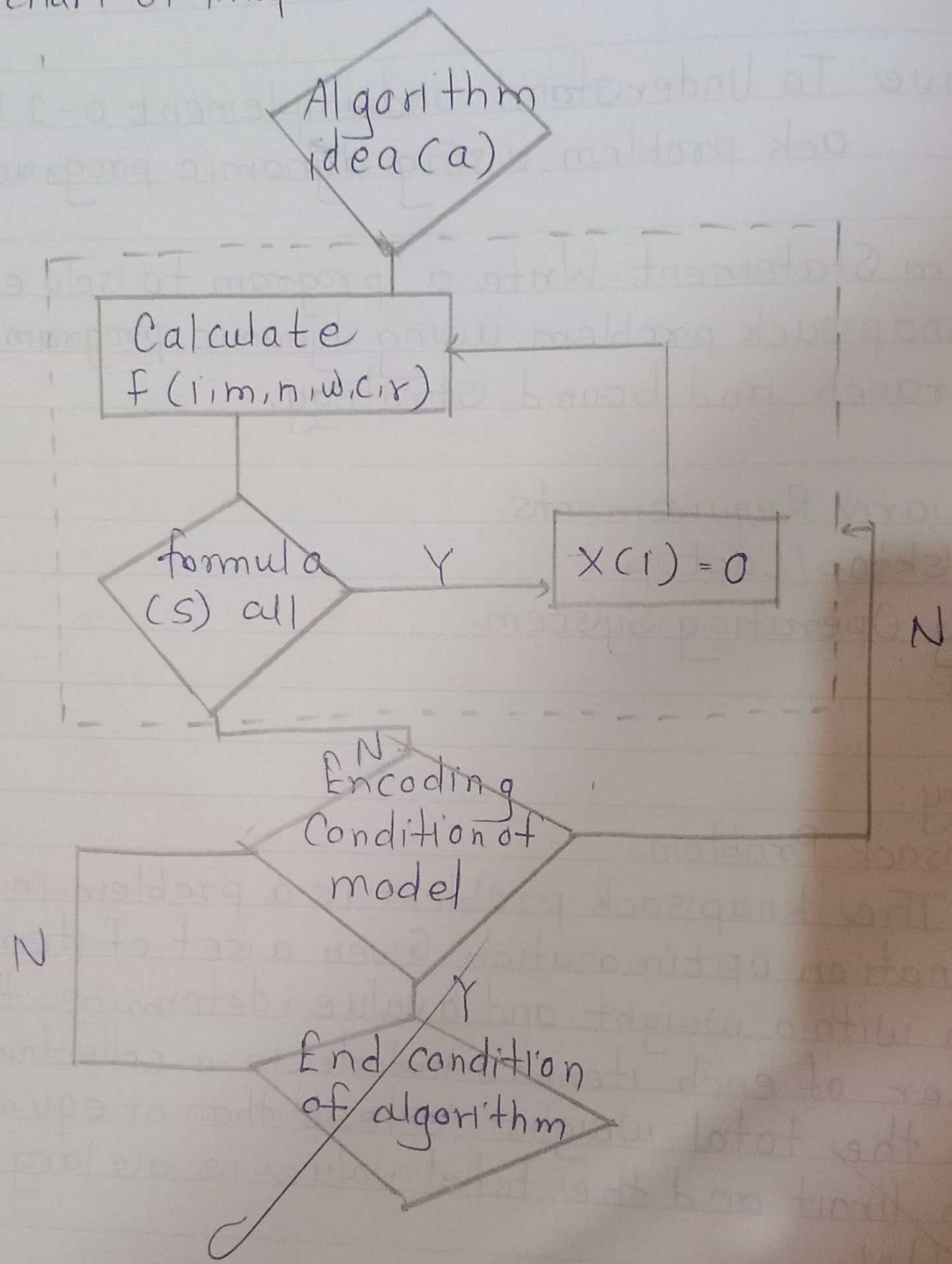
Knapsack Problem:

The knapsack problem is a problem in a combination optimazation. Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.

0/1 knapsack Problem:

The 0/1 knapsack problem means that the items are either completely or no items are filled in a knapsack. The 0/1 knapscakproblem is solved by the dynamic programming.

- Flowchart of knapsack



for eg We have two items having weights 2kg and 3kg respectively. If we pick the 2kg item then we cannot pick 1kg Item from 2kg item. We have to pick 2kg item completely.

Dynamic Programming Approach:

Let i be the highest numbered item in optimal solution S for W dolars. Then S = S - {i} is an optimal solution for w - wi dollars and value to the solution S is Vi plus the value of the sub problem.

Algorithm:

```
dynamic_knapsack(v, w, n, W)
for w = 0 to W do
    c[0, w] = 0
for i = 1 to n do
    c[i, 0] = 0
    for w = 1 to W do
        if wi ≤ w then
            if Vi + c[i-1, w-wi] then
                c[i, w] = Vi + c[i-1, w-wi]
            else
                c[i, w] = c[i-1, w]
        else
            c[i, w] = c[i-1, W]
```

Analysis:

The algorithm takes $\theta(n, w)$ times as a table C has $(n+1)(W+1)$ entries, where each entry requires $\theta(1)$ time to complete.

Page No.
Date

Time Complexity : O(N*W)
N = Number of weight element
W = Capacity

Space Complexity : O(N*W)
The use of 2-D array of size N*W.

Test cases:

1) Enter Number of Items: 3
Enter values: 10, 15, 40
Enter weight : 1, 2, 3
Enter Capacity : 6

We take maximum of (25, 40 + DP[2][6-3])
which is 65.

Mathematical Model:
Number of Items : 4
Weights : {3, 4, 6, 5}
Profit : {1, 3, 1, 4}
Capacity : 8

First we create a matrix 8×8
In matrix columns represent the weight i.e 8
Rows represent the profits and weight of items.

First we write the weights in the ascending order and profits according to their weights

Wi = {3, 4, 5, 6}

Pi = {2, 3, 4, 1}

The first row and the first column would be 0 as there is no item for w=0.

When i = 1, W = 1

W = 3 Since we have only one item in the set having weight 3, but the capacity of the knapsack is 1. We cannot fill the item of 3 kg in the knapsack of capacity 1 kg so add 0 at M[1][1]

Then we have to check weight, Profit and according to the knapsack capacity we have to add values into the matrix.

Lastly two weights are selected i.e 3 and 4 to maximize the profit.

Conclusion: We have successfully implemented 0-1 knapsack problem using dynamic approach.

20/9/22 (A)

Group A
Assignment No: 5

Title: N-Queens Problem

Objective: To solve N-queens problem by using backtracking.

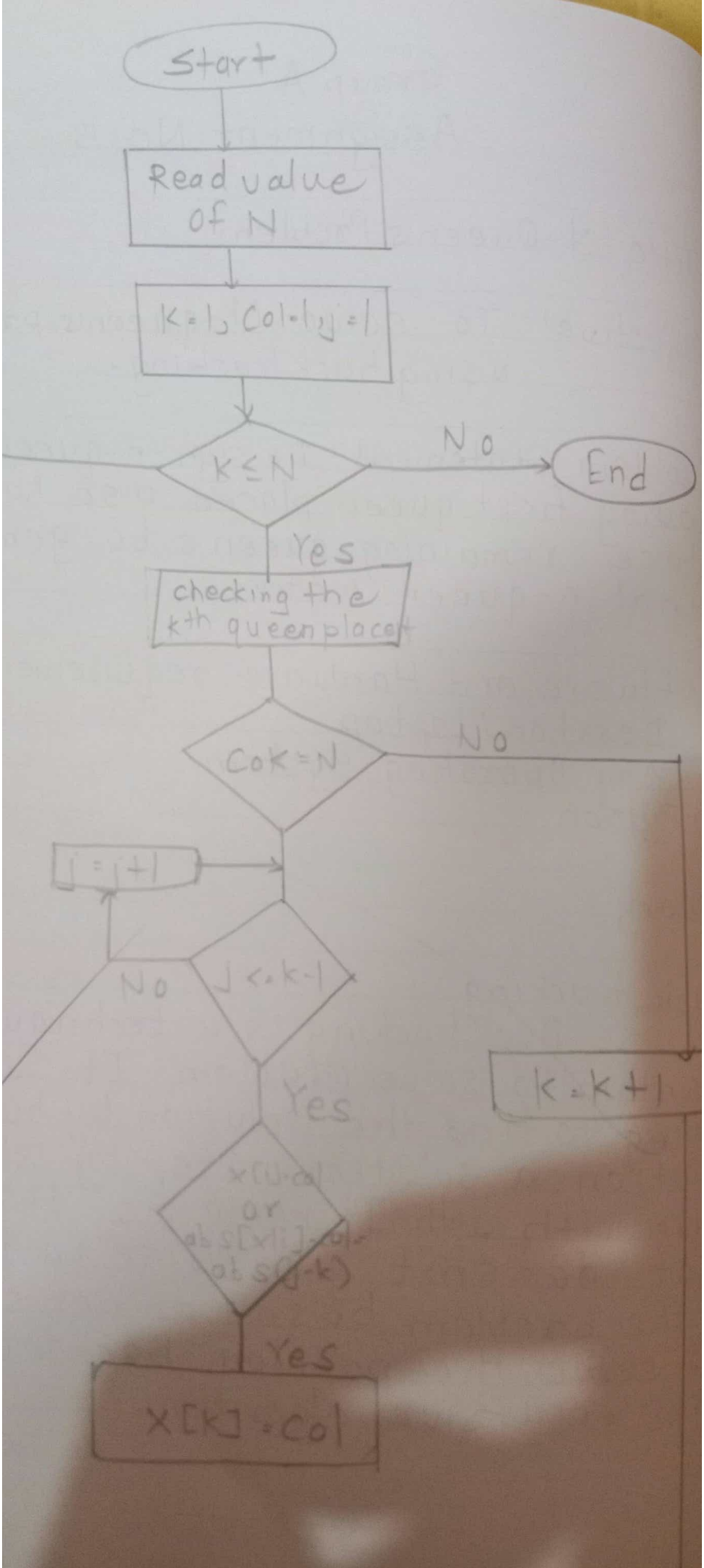
Problem Statement: Design n-queens matrix having first queen placed use backtracking to place remaining queens to generate the final n-queen matrix.

Software and Hardware requirement:
1. Desktop / Laptop
2. Any Operating System
3. Python

Theory:

Back tracking:

Backtracking is a technique based on algorithm to solve problem. It user recursive calling to find the solution by building a solution a solution step by step increasing value with wth. It removes the solution that doesn't give rise to the solution of the problem based on constraints given to solve the problem. Backtraking algorithm uses recursive algorithm.

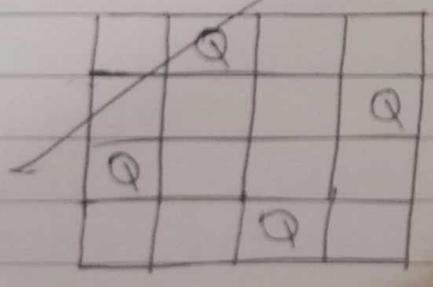

Start
Read value of N
K=1, Col=1, j=1
K≤N
No
End
Yes
checking the kth queen place
Col=N
No
j=j+1
No
J<k-1
Yes
k=k+1
Yes
X[k]=Col

Backtracking Algorithm:
Step 1: If current position is goal, return success
step 2: else
Step 3: if current position is an end point, return failed.
Step 4: Else if current-position is not end point, explore and repeat above steps.

N-queen-problem-

In n queen problem, we are given an N×N chessboard and we have to place a way that no two queens attack each other. A queen will attack another queen if it is placed in horizantal, vertical or diagonal points in its way.

e.g. 4-queen problem, the solution will be



Here, the binary output For n queen problem with is a queen.

{0,1,0,0}
{0,0,0,1}
{1,0,0,0}
{0,0,1,0}

N-queens algorithm:
Step1: Sort from 1st position in the array.
Step2: Place queen in the board and do.
2.1: After placing the queen, mark the position as a part of solution & then recursively chelk if will lead to a solution.
2.2: How if placing the queen doesn't lead to a solution and trackback and goto step(a).
Step3: If all queens are placed return true.

Step4: If all rows are tried and no solution found.

Time analysis:
Every call in n-queen can make O(n) calls.
It can be represented as
$T(n) = nT(n-1) + O(n)$
After As in nth row we make n to next row hence
$n + (n-1)$

and every call has a for loop

$O(n)$ - worst

$T(n) = nT(n-1) + O(n)$

This further decomposes to

Time Complexity of $O(n)$

Space complexity of $O(n^2)$.

Time complexity: $O(n^2)$

Test Cases:

No	input	expected	actual	Result
1	1	{1}	{1}	Pass
2	2	No sol	No sol	Pass
3	4	{0,1,0,0}	{0,1,0,0}	Pass
		{0,0,0,1}	{0,0,0,1}	
		{1,0,0,0}	{1,0,0,0}	
		{0,0,1,0}	{0,0,1,0}	

Conclusion: Successfully implemented solution for n-queens problem by using backtracking algorithm.

Group C: Assignment 1

Title: Installation of MetaMask

Objective: To understand the installation of Metamask and learn the study of spending either per transactions.

Problem statement: Installation of Metamask and spending ether per transaction.

Software & Hardware Requirements:
1) PC / Laptop
2) Any Operating System

Theory:

Blockchain:

A blockchain is a distributed database or ledger that is shared among the nodes of a computer network. As a database a blockchain stores information electraically in digital format. Blockchain are best known for their crucial role in cryptocurrency systems such as Bitcoin for maintains a secure and decentralized record of transaction. A database usually structures its data into tables whereas a blockchain as its name implies structure its data into chunks that are struck together. Blockchain provides high level of security to every node.

Page No.
Date

MetaMask:

MetaMask is a popular cryptocurrency wallet known for its easy of use, availability on both desktops and mobile devices the ability to buy, send and receive cryptocurrency from within the wallet and collect non-fungible tokens accross two blockchains. Meta Mask is a (software) cryptocurrency wallet used to interact with ethereum blockchain.

Ethereum Per Transaction:

An ethereum transactions refers to an action initiates by on externaly-owned account. In other words an account managed by a human, not a contract.

for. If Bob sends Alice 1 ETH, Bob's account must be debited and Alice must be credited. This (state changing action takes place within a transaction.

Steps:

Step 1: Go to chrome web store Extensions Section

Step 2: Search Metamask.

Step 3: Check the number of downloads to make (sure that the legitimate). Metamask is being installed as hackeres might try to make clones of it.

Step 4: Click the Add to chrome button.

Step 5: Once installation is complete this page will be displayed. Click on the Get started button.

Step 6: This is the first time creating a wallet so click the create a wallet button. If there is already a wallet then imort the already created using the import wallet button.

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

Step 8: Create Password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask.

Step 9: Click on the dark area which says click here to reveal secret words to get your secret phrase.

Step 10: This is most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer.

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

Step 12: Click on the confirm button Please follow the tips mentioned.

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

Step 14: One can access MetaMask in the browser by clicking the foxface in icon on the top right.

Conclusion: We successfully understand and implemented the installation of MetaMask.

V.A. Mejor (R)

Group : C
Assignment No-2

Page No. 72
Date

Title: Create Wallet Using Metamask

Objective: To learn how to create own wallet using Metamask for crypto transactions.

Problem Statement: Create your own wallet using Metamask for crypto transactions.

Software & Hardware Requirements:
1) PC/ Laptop
2) Any Operating System
3) Any Browser

Theory:

Blockchain:

Blockchain is a system of recording information in a way that makes it difficult or impossible to change, hack or cheat the system. A blockchain is essentially a digital ledger of transactions that is duplicated and distributed across the entire network of computer system on the blockchain. Each block in the chain contains a number of transactions & every time a new transaction occurs on the blockchain, a record of that transaction is added to every participants ledger.

Page No.
Date

Meta-Mask:

Metamask is a free crypto wallet software that people use to interact in the crypto world. It lets you buy, sell & trade crypto asserts for the Ftherum blockchain, much like row a real wallet lets you purchase items in the real world.

Metamask is free to use and can be installed on browser as an extension on the internet browsers like Google chrome, Firefox, Brave etc or we can download it on a smartphone application both on ios and Android. with over 30 million users, Metamask is one of the most popular crypto currency wallets everyday.

Features of Blockchain:

- Buy, receive, send and swap ether.
- Connect to Ethereum Dapps.
- Connect to other crypto wallets.
- Play blockchain based games.
- Access different networks such as the BNB smart contracts chain and other test nets.

Steps:

To use metamask you will either chrome, brave, firefox or any browser.

Page No.
Date

Step 1: First you will need to download and install the official metamask extension for your choosen browser.

Step 2: One installed "Welcome to Metamask" splash screen will appear. Click on "Get Started" button to begin creating our etherum wallet using Metamask.

Step 3: Click on "Create Wallet" button.

Step 4: You will then asked if you want to help improve Metamask. Click on "No Thanks button".

Step 5: Then create your own new password for your metamask wallet.

Step 6: Read and accept the terms of use and click 'Create' button, once password has been set.

Step 7: Metamask will then present you with your 12-word backup phase. You will need to write this phrases carefully.

Click Next once you have written.

Page No.
Date

Step 8: Confirm your backup phrase on the next screen by entering the words in the same order saved previously and these then click 'Confirm' once done.

Step 9: Click on "All Done" on the final page and the you will be automatically in to Metamask.

Step 10: Successfully created wallet using Metamask.

Conclusion: We have successfully understand and create the Metamask Wallet.

VBR

Group C

Page No. 76
Date

Assignment No-3

Title: Smart Contract

Objective: To understand the concept of smart contract and implement smart contract on a test network.

Problem Statement: Write a smart contract on a test network for bank account of a customer for following operations.
- Deposite Money
- Withdraw Money
- Show Balance

Software and Hardware Requirements:
1) PC/ Laptop
2) Any Operating System

Theory:

Smart Contracts:

A smart contracts is a computer program that directly and automatically controls the transfer of digital assets between the parties under certain conditions. A smart contract while also automatically enforcing the contract. Smart contracts are programs that execute exactly as they are set up (programmed) by their.

Page No.
Date

Creators: Just like a traditional contract is enforceable by law, smart contracts are enforceable by code.

. The bitcoin network was the first to use some sort of smart contract by using them to transfer is actually available in the sender account.

. Later the ethereum platform emeged which was considered more powerful, precisely because the developers could make custom contracts in a turing complete language.

. It is to be noted that the contracts written in the case of the bitcoin network were written in a the bitcoin network.

Features of smart contracts:

1) Distributed: Everyone on the network is guranted to have a copy of all the conditions of the smart contract and they cannot be changed by one of the parties. A smart contract implementation in the bitcoin network.

Features of smart contracts:
1) Distributed: Everyone on the network is guranteed to have a copy of all the conditions of the

smart contract and they cannot be changed by one of the parties. A smart contract is replicated and distributed by all the nodes connected to the network.

2) Deterministic: Smart contracts can only perform functions for which they are designed only when the required conditions are met. The final outcome will not vary, no matter who executes the smart contract.

3) Immutable: Once deployed smart contract cannot be changed, it can only be removed as long as the functionality is implemented previously.

4) Autonomy: There is no third party involved. The contract is made by you and shared between the parties. No intermediaries are involved which minimizes bullying and grants full authority to the dealing parties.

5) Transparent: Smart Contracts are always stored on a public distributed ledger called blockchain due to which the code is visible to everyone whether or not they are participants in the smart contracts.

Steps:

Step 1: Navigate to remix and the size will open

a browser with a default contract.

Step 2: The first thing we need to do is to create a new contract by selecting from remix is left menu bar.

Step 3: Then provide a name for a new solidity file that has an extension '.sal' Name the contract. Hello world and click on 'ok' to continue. This will create a blank contract.

Step 4: Type the code in empty authoring pane to create your first contract. The contract is created using the contract keyword.

Step 5: Look at the action window to the right of Remix, It has got several tabs - compile, Run settings, Debugger, Analysis & support.

Step 6: However the important action is deployment of a contract & that can be done using the 'create' button to deploy the Contract.

Step 7: Click on create button to deploy the contract to the browser. The lower panel of Remix will show the results of execution.

Conclusion: We have successfully created smart contract.

Page No. 80
Date

Title: Solidity in Blockchain

Objective: To understand the concepts of solidity in blockchain.

Problem Statement: Write a program in solidity to create student data. Use the following constructs.
- structures
- Arrays
- fallback

Deploy this as smart contract on Etherum and observe the transactions fee and Gas value.

Software and Hardware Requirements:
1) PC/Laptop
2) Any Operating System

Theory:

Smart Contract:

A smart contract is a computer program that directly & automatically controls the transfer of digital assests between the parties under certain conditions. A smart contract works in the same way as a traditional contruct while also automatically enforcing the contruct. Smart contract are programs that execute extactly as they are set up by their creators.

Page No.
Date

Solidity in Blockchain:

Solidity is an object oriented programming language for implementing smart contracts on various blockchain platforms, most notably Ethercum. Solidity is a brand new programming language developed by Ethereum the second largest cryptocurency market by capitalization.

It is used to create smart contracts that implement business logic and generate a chain of transaction records in the blockchain system. It acts as tool for creating machine level code and compiling it on Ethereum Virtual machine (EVM).

Properties of fallback function:

1) Has no name or arguments
2) Cannot return anything
3) If it is not marked payable, the contract will throw an exceptation if it receives plain ether without data.
4) Can be defined once per contract.
5) It is mandatory to mark it external.

Steps

Remix IDE is generally used to compile and run solidity smart contracts.

Page No.
Date

Step 1: Open Remix IDE on any of your browsers select on the new file & click on solidity to choose the environment.

Step 2: Write the smart constracts in the code section & click the compile button under the compiler window to compile the contruct

Step 3: To execute the code click on the Deploy button under deploy and Run transactions window.

Step 4: After Deploying the code click on the method calls under the drop down of deployed contracts to run the program and for output, check to click on the drop-down on the console.

Step 5: For deploying click on the Debug button corresponding to the method call in the console. Here you can check each function call and variable assignments.

Conclusion: We have successfully understand the concept of solidity programming in blockchain.

VIVA B

Group C: Assignment No-5

Title: Use cases of Blockchain.

Objectives: To understand the concept of types of blockchain and different use cases of blockchain.

Problem Statement: Write a survey Report on types of blockchain and its real time use cases.

Software and Hardware Requirements:
1) PC/laptop
2) Any Operating System

Theory:

Types of Blockchain:
1) Public Blockchain
2) Private Blockchain
3) Hybrid Blockchain
4) Cosortium Blockchain

1) Public Blockchain:

As the name is public this blockchain is open to public, which means it is not owned by anyone. Anyone having internet and a computer with good hardware can participate in the public blockchain. All the computer in the network hold the copy of other nodes or block present in the network.

2) Private Blockchain:

These are not as open as a public blockchain. They are open to some authorised users only. These blockchain are operated in the closed network.

Use Case: Private blockchain are used for internal auditing voting and assert management. eg. Hyperledger or Corda.

3) Hybrid Blockchain

It is a combination of both public & private blockchain permission-based and permissionaless systems are used user acess information via smart contract. Even a primary entity owns a hydrid blockchain it cannot alter transaction.

Use Case. It provides a greater solution to the health care industry, government, real state and financial companies.

4) Consortium Blockchain

It is a creative approach that solves the needs of the organization. This blockchain validates the transaction and also initiates.

or receives transactions. It is also known as federated Blockchain. In this blockchain some part is public and some part is private. In this type more than one organization manages the blockchain.

Use Case: It has high potential in business, banks and other payment processors. For eg. Tendermint and Multichain.

Real Time Use Cases of Blockchain:

1) Blockchain in Money Transfer: Pioneered by Bitcoin, cryptocurrency transfer apps are exploding in popularity right now.

2) Blockchain and IOT: The Internet of Things is the next logical boom in blockchain applications. Blockchain infused IOT adds a higher level of security to prevent data branches by utilizing transparrency & virtual incorruptibility of the technology to keep things "smart".

Conclusion: We have successfully understand different real time use cases of blockchain.