Practical No. A1

Que :- Implement Depth First Search algorithm and Breadth First algorithm. Use an undirected graph and develop a recursive algorithm for searching all the vertices of the graph or tree data structure.

Solution :-

1. Depth First search :

Input =>

#include <iostream>

#include <map>

#include <list>

using namespace std;

class Graph {

public:

map<int, bool> visited;

map<int, list<int> > adj;

void addEdge(int v, int w);

void DFS(int v);

};

void Graph::addEdge(int v, int w) {

adj[v].push\_back(w);

}

void Graph::DFS(int v) {

visited[v] = true;

cout << v << " ";

list<int>::iterator i;

for (i = adj[v].begin(); i != adj[v].end(); ++i)

if (!visited[\*i])

DFS(\*i);

}

int main() {

Graph g;

g.addEdge(0, 1);

g.addEdge(0, 2);

g.addEdge(1, 2);

g.addEdge(2, 0);

g.addEdge(2, 3);

g.addEdge(3, 3);

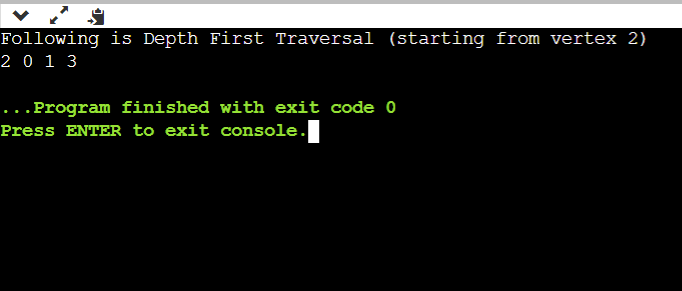
cout << "Following is Depth First Traversal (starting from vertex 2) \n";

g.DFS(2);

return 0;

}

Output =>



2. Breadth First Search :

Input =>

#include<iostream>

#include <list>

using namespace std;

class Graph {

int V;

list < int >\*adj;

public:

Graph (int V);

void addEdge (int v, int w);

void BFS (int s);

};

Graph::Graph (int V) {

this->V = V;

adj = new list < int >[V];

}

void Graph::addEdge (int v, int w) {

adj[v].push\_back (w);

}

void Graph::BFS (int s) {

bool \* visited = new bool[V];

for (int i = 0; i < V; i++)

visited[i] = false;

list < int >queue;

visited[s] = true;

queue.push\_back (s);

list < int >::iterator i;

while (!queue.empty ()) {

s = queue.front ();

cout << s << " ";

queue.pop\_front ();

for (i = adj[s].begin (); i != adj[s].end (); ++i) {

if (!visited[\*i]) {

visited[\*i] = true;

queue.push\_back (\*i);

}

}

}

}

int main () {

Graph g (4);

g.addEdge (0, 1);

g.addEdge (0, 2);

g.addEdge (1, 2);

g.addEdge (2, 0);

g.addEdge (2, 3);

g.addEdge (3, 3);

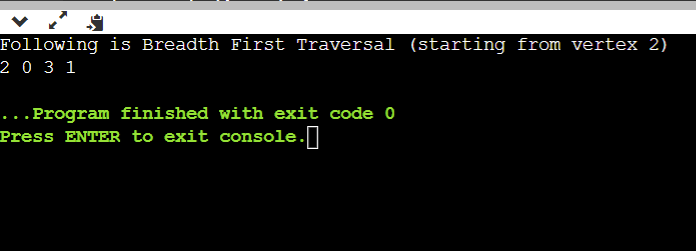
cout << "Following is Breadth First Traversal (starting from vertex 2) \n";

g.BFS (2);

return 0;

}

Output =>



Practical No. A2

Que :- Implement A star Algorithm for any game search problem.

Solution :-

Input =>

#include <bits/stdc++.h>

using namespace std;

#define ROW 9

#define COL 10

typedef pair < int, int > Pair;

typedef pair < double, pair < int, int >> pPair;

struct cell {

int parent\_i, parent\_j;

double f, g, h;

};

bool isValid (int row, int col) {

return (row >= 0) && (row < ROW) && (col >= 0) && (col < COL);

}

bool isUnBlocked (int grid[][COL], int row, int col) {

if (grid[row][col] == 1)

return (true);

else

return (false);

}

bool isDestination (int row, int col, Pair dest) {

if (row == dest.first && col == dest.second)

return (true);

else

return (false);

}

double calculateHValue (int row, int col, Pair dest) {

return ((double) sqrt ((row - dest.first) \* (row - dest.first)+(col - dest.second) \* (col - dest.second)));

}

void tracePath (cell cellDetails[][COL], Pair dest) {

printf ("\nThe Path is ");

int row = dest.first;

int col = dest.second;

stack < Pair > Path;

while (!(cellDetails[row][col].parent\_i == row && cellDetails[row][col].parent\_j == col)) {

Path.push (make\_pair (row, col));

int temp\_row = cellDetails[row][col].parent\_i;

int temp\_col = cellDetails[row][col].parent\_j;

row = temp\_row;

col = temp\_col;

}

Path.push (make\_pair (row, col));

while (!Path.empty ()) {

pair < int, int >p = Path.top ();

Path.pop ();

printf ("-> (%d,%d) ", p.first, p.second);

}

return;

}

void aStarSearch (int grid[][COL], Pair src, Pair dest) {

if (isValid (src.first, src.second) == false) {

printf ("Source is invalid\n");

return;

}

if (isValid (dest.first, dest.second) == false) {

printf ("Destination is invalid\n");

return;

}

if (isUnBlocked (grid, src.first, src.second) == false || isUnBlocked (grid, dest.first, dest.second) == false) {

printf ("Source or the destination is blocked\n");

return;

}

if (isDestination (src.first, src.second, dest) == true) {

printf ("We are already at the destination\n");

return;

}

bool closedList[ROW][COL];

memset (closedList, false, sizeof (closedList));

cell cellDetails[ROW][COL];

int i, j;

for (i = 0; i < ROW; i++) {

for (j = 0; j < COL; j++) {

cellDetails[i][j].f = FLT\_MAX;

cellDetails[i][j].g = FLT\_MAX;

cellDetails[i][j].h = FLT\_MAX;

cellDetails[i][j].parent\_i = -1;

cellDetails[i][j].parent\_j = -1;

}

}

i = src.first, j = src.second;

cellDetails[i][j].f = 0.0;

cellDetails[i][j].g = 0.0;

cellDetails[i][j].h = 0.0;

cellDetails[i][j].parent\_i = i;

cellDetails[i][j].parent\_j = j;

set < pPair > openList;

openList.insert (make\_pair (0.0, make\_pair (i, j)));

bool foundDest = false;

while (!openList.empty ()) {

pPair p = \*openList.begin ();

openList.erase (openList.begin ());

i = p.second.first;

j = p.second.second;

closedList[i][j] = true;

double gNew, hNew, fNew;

if (isValid (i - 1, j) == true) {

if (isDestination (i - 1, j, dest) == true) {

cellDetails[i - 1][j].parent\_i = i;

cellDetails[i - 1][j].parent\_j = j;

printf ("The destination cell is found\n");

tracePath (cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i - 1][j] == false && isUnBlocked (grid, i - 1, j) == true) {

gNew = cellDetails[i][j].g + 1.0;

hNew = calculateHValue (i - 1, j, dest);

fNew = gNew + hNew;

if (cellDetails[i - 1][j].f == FLT\_MAX || cellDetails[i - 1][j].f > fNew) {

openList.insert (make\_pair (

fNew, make\_pair (i - 1, j)));

cellDetails[i - 1][j].f = fNew;

cellDetails[i - 1][j].g = gNew;

cellDetails[i - 1][j].h = hNew;

cellDetails[i - 1][j].parent\_i = i;

cellDetails[i - 1][j].parent\_j = j;

}

}

}

if (isValid (i + 1, j) == true) {

if (isDestination (i + 1, j, dest) == true) {

cellDetails[i + 1][j].parent\_i = i;

cellDetails[i + 1][j].parent\_j = j;

printf ("The destination cell is found\n");

tracePath (cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i + 1][j] == false && isUnBlocked (grid, i + 1, j) == true) {

gNew = cellDetails[i][j].g + 1.0;

hNew = calculateHValue (i + 1, j, dest);

fNew = gNew + hNew;

if (cellDetails[i + 1][j].f == FLT\_MAX || cellDetails[i + 1][j].f > fNew) {

openList.insert (make\_pair(fNew, make\_pair (i + 1, j)));

cellDetails[i + 1][j].f = fNew;

cellDetails[i + 1][j].g = gNew;

cellDetails[i + 1][j].h = hNew;

cellDetails[i + 1][j].parent\_i = i;

cellDetails[i + 1][j].parent\_j = j;

}

}

}

if (isValid (i, j + 1) == true) {

if (isDestination (i, j + 1, dest) == true) {

cellDetails[i][j + 1].parent\_i = i;

cellDetails[i][j + 1].parent\_j = j;

printf ("The destination cell is found\n");

tracePath (cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i][j + 1] == false && isUnBlocked (grid, i, j + 1) == true) {

gNew = cellDetails[i][j].g + 1.0;

hNew = calculateHValue (i, j + 1, dest);

fNew = gNew + hNew;

if (cellDetails[i][j + 1].f == FLT\_MAX || cellDetails[i][j + 1].f > fNew) {

openList.insert (make\_pair(fNew, make\_pair (i, j + 1)));

cellDetails[i][j + 1].f = fNew;

cellDetails[i][j + 1].g = gNew;

cellDetails[i][j + 1].h = hNew;

cellDetails[i][j + 1].parent\_i = i;

cellDetails[i][j + 1].parent\_j = j;

}

}

}

if (isValid(i, j - 1) == true)

{

if (isDestination(i, j - 1, dest) == true)

{

cellDetails[i][j - 1].parent\_i = i;

cellDetails[i][j - 1].parent\_j = j;

printf("The destination cell is found\n");

tracePath(cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i][j - 1] == false && isUnBlocked(grid, i, j - 1) == true)

{

gNew = cellDetails[i][j].g + 1.0;

hNew = calculateHValue(i, j - 1, dest);

fNew = gNew + hNew;

if (cellDetails[i][j - 1].f == FLT\_MAX || cellDetails[i][j - 1].f > fNew)

{

openList.insert(make\_pair(fNew, make\_pair(i, j - 1)));

cellDetails[i][j - 1].f = fNew;

cellDetails[i][j - 1].g = gNew;

cellDetails[i][j - 1].h = hNew;

cellDetails[i][j - 1].parent\_i = i;

cellDetails[i][j - 1].parent\_j = j;

}

}

}

if (isValid(i - 1, j + 1) == true)

{

if (isDestination(i - 1, j + 1, dest) == true)

{

cellDetails[i - 1][j + 1].parent\_i = i;

cellDetails[i - 1][j + 1].parent\_j = j;

printf("The destination cell is found\n");

tracePath(cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i - 1][j + 1] == false && isUnBlocked(grid, i - 1, j + 1) == true)

{

gNew = cellDetails[i][j].g + 1.414;

hNew = calculateHValue(i - 1, j + 1, dest);

fNew = gNew + hNew;

if (cellDetails[i - 1][j + 1].f == FLT\_MAX || cellDetails[i - 1][j + 1].f > fNew)

{

openList.insert(make\_pair(fNew, make\_pair(i - 1, j + 1)));

cellDetails[i - 1][j + 1].f = fNew;

cellDetails[i - 1][j + 1].g = gNew;

cellDetails[i - 1][j + 1].h = hNew;

cellDetails[i - 1][j + 1].parent\_i = i;

cellDetails[i - 1][j + 1].parent\_j = j;

}

}

}

if (isValid(i - 1, j - 1) == true)

{

if (isDestination(i - 1, j - 1, dest) == true)

{

cellDetails[i - 1][j - 1].parent\_i = i;

cellDetails[i - 1][j - 1].parent\_j = j;

printf("The destination cell is found\n");

tracePath(cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i - 1][j - 1] == false && isUnBlocked(grid, i - 1, j - 1) == true)

{

gNew = cellDetails[i][j].g + 1.414;

hNew = calculateHValue(i - 1, j - 1, dest);

fNew = gNew + hNew;

if (cellDetails[i - 1][j - 1].f == FLT\_MAX || cellDetails[i - 1][j - 1].f > fNew)

{

openList.insert(make\_pair(fNew, make\_pair(i - 1, j - 1)));

cellDetails[i - 1][j - 1].f = fNew;

cellDetails[i - 1][j - 1].g = gNew;

cellDetails[i - 1][j - 1].h = hNew;

cellDetails[i - 1][j - 1].parent\_i = i;

cellDetails[i - 1][j - 1].parent\_j = j;

}

}

}

if (isValid(i + 1, j + 1) == true)

{

if (isDestination(i + 1, j + 1, dest) == true)

{

cellDetails[i + 1][j + 1].parent\_i = i;

cellDetails[i + 1][j + 1].parent\_j = j;

printf("The destination cell is found\n");

tracePath(cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i + 1][j + 1] == false && isUnBlocked(grid, i + 1, j + 1) == true)

{

gNew = cellDetails[i][j].g + 1.414;

hNew = calculateHValue(i + 1, j + 1, dest);

fNew = gNew + hNew;

if (cellDetails[i + 1][j + 1].f == FLT\_MAX || cellDetails[i + 1][j + 1].f > fNew)

{

openList.insert(make\_pair(fNew, make\_pair(i + 1, j + 1)));

cellDetails[i + 1][j + 1].f = fNew;

cellDetails[i + 1][j + 1].g = gNew;

cellDetails[i + 1][j + 1].h = hNew;

cellDetails[i + 1][j + 1].parent\_i = i;

cellDetails[i + 1][j + 1].parent\_j = j;

}

}

}

if (isValid(i + 1, j - 1) == true)

{

if (isDestination(i + 1, j - 1, dest) == true)

{

cellDetails[i + 1][j - 1].parent\_i = i;

cellDetails[i + 1][j - 1].parent\_j = j;

printf("The destination cell is found\n");

tracePath(cellDetails, dest);

foundDest = true;

return;

}

else if (closedList[i + 1][j - 1] == false && isUnBlocked(grid, i + 1, j - 1) == true)

{

gNew = cellDetails[i][j].g + 1.414;

hNew = calculateHValue(i + 1, j - 1, dest);

fNew = gNew + hNew;

if (cellDetails[i + 1][j - 1].f == FLT\_MAX || cellDetails[i + 1][j - 1].f > fNew)

{

openList.insert(make\_pair(fNew, make\_pair(i + 1, j - 1)));

cellDetails[i + 1][j - 1].f = fNew;

cellDetails[i + 1][j - 1].g = gNew;

cellDetails[i + 1][j - 1].h = hNew;

cellDetails[i + 1][j - 1].parent\_i = i;

cellDetails[i + 1][j - 1].parent\_j = j;

}

}

}

}

if (foundDest == false)

printf("Failed to find the Destination Cell\n");

return;

}

int main()

{

int grid[ROW][COL] = {

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

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

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

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

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

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

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

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

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

};

Pair src = make\_pair(8, 0);

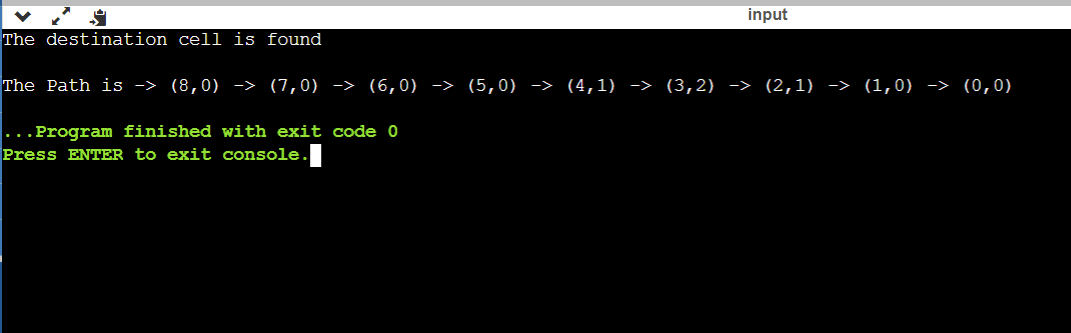
Pair dest = make\_pair(0, 0);

aStarSearch(grid, src, dest);

return (0);

}

Output =>



Practical No. A3

Que :- Implement Greedy search algorithm for any of the following application:

Prim's Minimal Spanning Tree Algorithm

Solution :-

Input =>

#include <bits/stdc++.h>

using namespace std;

#define V 5

int minKey(int key[], bool mstSet[]) {

int min = INT\_MAX, min\_index;

for (int v = 0; v < V; v++)

if (mstSet[v] == false && key[v] < min)

min = key[v], min\_index = v;

return min\_index;

}

void printMST(int parent[], int graph[V][V]) {

cout<<"Edge \tWeight\n";

for (int i = 1; i < V; i++)

cout<<parent[i]<<" - "<<i<<" \t"<<graph[i][parent[i]]<<" \n";

}

void primMST(int graph[V][V]) {

int parent[V];

int key[V];

bool mstSet[V];

for (int i = 0; i < V; i++)

key[i] = INT\_MAX, mstSet[i] = false;

key[0] = 0;

parent[0] = -1;

for (int count = 0; count < V - 1; count++){

int u = minKey(key, mstSet);

mstSet[u] = true;

for (int v = 0; v < V; v++)

if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v])

parent[v] = u, key[v] = graph[u][v];

}

printMST(parent, graph);

}

int main() {

int graph[V][V] = { { 0, 2, 0, 6, 0 },

{ 2, 0, 3, 8, 5 },

{ 0, 3, 0, 0, 7 },

{ 6, 8, 0, 0, 9 },

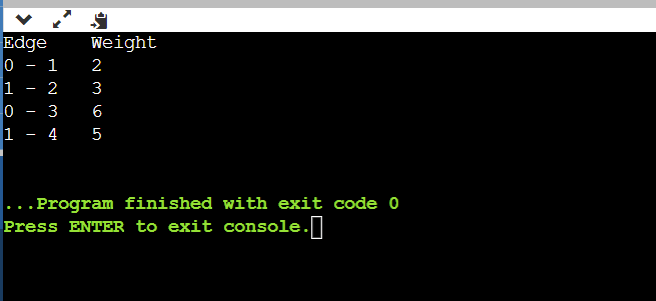
{ 0, 5, 7, 9, 0 } };

primMST(graph);

return 0;

}

Output =>



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Website URL** | **Purpose of Website** | **Things liked in the website** | **Things disliked in the website** | **Overall evaluation of the website (Good/Bad)** |
| 1 | https://www.wikipedia.org/ | Online Encyclopedia | Simple UI design, User Friendly | N/A | Good |
| 2 | https://www.ebay.com/ | Ecommerce | Interactive, Simple design, Easy Navigation | N/A | Good |
| 3 | https://www.apple.com/in/ | Ecommerce | Interactive, User Friendly | N/A | Good |
| 4 | https://arngren.net/ | Ecommerce | N/A | Cluttered | Bad |
| 5 | https://www.art.yale.edu/ | Educational | User Friendly | Poor Presentation, Cluttered, Unattractive | Bad |

Practical No. 1

Practical No. 2

Input :-

page1.html :

<!DOCTYPE html>

<html>

<head>

<title> WT Lab 2 </title>

<meta name="description" content="Page Description Here"/>

<meta http-equiv="Content-Type" content="text/html"; charset="utf-8"/>

<meta name="viewport" content="width=device-width, initial-scale=1"/>

<style>

body {

margin:0;

padding:0;

line-height: 1.5em;

}

/\* ##### Navigation Bar ##### \*/

ul {

list-style-type: none;

margin: 500;

padding: 500;

background-color: #000;

}

li {

float: left;

border-right:1px solid #bbb;

}

li:last-child {

border-right: none;

}

li a {

display: block;

color: white;

text-align: left;

padding: 14px 120px;

text-decoration: none;

}

li a:hover:not(.active) {

background-color: #808080;

}

.active {

background-color: #0099ff;

}

</style>

</head>

<body>

<a id="home"></a>

<table width="100%" height="60" border="0" cellspacing="5" cellpadding="5" bgcolor="#000000">

<tr>

<td align="left">

<ul style="display: inline;font face="Verdana; text-align="center";">

<li><a class="active" href="#home">Home</a></li>

<li><a href="C:\Users\OJUS\OneDrive\Desktop\⠀\WT\Lab\Practical 2\about Us.html" target="blank\_">About Us</a></li>

<li><a href="#services">Services</a></li>

<li><a href="#clients">Clients</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</td>

</tr>

</table>

<a id="mail"></a>

<table width="100%" height="150" border="0" cellspacing="5" cellpadding="5" bgcolor="#F0F0F0">

<tr>

<td align="center">

<table width="708" border="0" cellspacing="1" cellpadding="1">

<tr>

<td width="400" align="left"><font face="Verdana, Geneva, sans-serif" color="black" size="5"><strong>My Business</strong><br>Web Page</font></td>

<td width="291" align="right"><font face="Verdana, Geneva, sans-serif" color="black" size="5"><strong>Contact Us Today!</strong><br>webdev@webdev.com</font></td>

</tr>

</table>

</td>

</tr>

</table>

<table width="100%" height="150" border="0" cellspacing="5" cellpadding="5" bgcolor="#F0F0F0">

<tr>

<td align="center">

<table width="600" border="0" cellspacing="1" cellpadding="1">

<tr>

<td align="center"><img src="D:\Projects\Web Development\Downloads\Section 10 - HTML Development\Lecture 98 - HTML Project l Image\website2.jpg" width="1000" height="500"></td>

</tr>

</table>

</td>

</tr>

</table>

<table width="100%" height="80" border="0" cellspacing="5" cellpadding="5" bgcolor="white">

<tr>

<td height="70" align="center" valign="top">

<table width="730" border="0" cellspacing="1" cellpadding="1" bgcolor="#FF9900">

<tr>

<td width="800" align="center" nowrap><font face="Verdana, Geneva, sans-serif" color="white" size="6"><a href="C:\Users\OJUS\OneDrive\Desktop\⠀\WT\Lab\Practical 2\page2.html" target="blank\_"><strong>Demo Web Page</strong></a></font></td>

</tr>

</table>

</td>

</tr>

</table>

<table width="100%" height="80" border="0" cellspacing="5" cellpadding="5" bgcolor="white">

<tr>

<td height="70" align="center" valign="top">

<table width="730" border="0" cellspacing="1" cellpadding="1">

<tr>

<td width="800" align="center"><font face="Verdana, Geneva, sans-serif" color="black" size="6">My Company Web Page<strong><br>Dream. Plan. Create.</strong></font></td>

</tr>

</table>

</td>

</tr>

</table>

<a id="contact"></a>

<table width="100%" height="80" border="0" cellspacing="5" cellpadding="5" bgcolor="white">

<tr>

<td height="70" align="center" valign="top">

<table width="730" border="0" cellspacing="1" cellpadding="1" bgcolor="#FF9900">

<tr>

<td width="800" align="center" nowrap><font face="Verdana, Geneva, sans-serif" color="white" size="6"><a href="#mail"><strong>Contact Us Today!</strong></a></font></td>

</tr>

</table>

</td>

</tr>

</table>

<table width="100%" height="80" border="0" cellspacing="5" cellpadding="5" bgcolor="white">

<tr>

<td width="800" align="center" nowrap valign="top"><font face="Verdana, Geneva, sans-serif" color="black" size="5"><strong>Register for our Newsletter:</strong></font><br>

<form name="contact" method="post" action="send.php">

<input type="text" name="email" />

<input type="submit" name="button" value="Submit" />

</form>

</td>

</tr>

</table>

<a id="services"></a>

<table width="100%" height="236" border="0" cellspacing="5" cellpadding="5" bgcolor="white">

<tr>

<td height="70" align="center">

<table width="900" border="1" bordercolor="#CCCCCC" cellspacing="1" cellpadding="1" align="center">

<tr>

<td width="300"><font face="Verdana, Geneva, sans-serif" color="black"><strong><u>ABOUT US</u></strong></font></td>

<td width="300"><font face="Verdana, Geneva, sans-serif" color="black"><strong><u>OUR VISION</u></strong></font></td>

<td width="300"><font face="Verdana, Geneva, sans-serif" color="black"><strong><u>SERVICES</u></strong></font></td>

</tr>

<tr>

<td width="300"><font face="Verdana, Geneva, sans-serif" color="black" size="2" >Web development company</font></td>

<td width="300"><font face="Verdana, Geneva, sans-serif" color="black" size="2" >Web development for profit</font></td>

<td width="300">

<font face="Verdana, Geneva, sans-serif" color="black" size="2" >

<ul>

<li>IT Management <br> Globalization</li>

</ul>

</font>

</td>

</tr>

</table>

</td>

</tr>

</table>

<table width="100%" height="60" border="0" cellspacing="5" cellpadding="5" bgcolor="#000000">

<tr>

<td align="center"><font face="Verdana, Geneva, sans-serif" color="white">Home - About - Services - Clients - Contact<br><br>Made By :-<br>MyBusinessWebDev</font></td>

</tr>

</table>

</body>

</html>

page2.html :

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Demo Web Page</title>

<style>

body {

background: #222;

margin-top: 20px;

font-family: arial;

color: white;

}

.thumbnails img {

height: 80px;

border: 4px solid #555;

padding: 1px;

margin: 0 10px 10px 0;

}

.thumbnails img:hover {

border: 4px solid #00ccff;

}

.preview img {

border: 4px solid #444;

padding: 1px;

width: 800px;

}

</style>

</head>

<body>

<div id="container" align="center">

<h1>My Web Page Gallery</h1>

<br />

<div class="thumbnails">

<img onmouseover="preview.src=img1.src" name="img1" src="images/img1.jpg" />

<img onmouseover="preview.src=img2.src" name="img2" src="images/img2.jpg" />

<img onmouseover="preview.src=img3.src" name="img3" src="images/img3.jpg" />

<img onmouseover="preview.src=img4.src" name="img4" src="images/img4.jpg" />

<img onmouseover="preview.src=img5.src" name="img5" src="images/img5.jpg" />

</div>

<br/>

<div class="preview" align="center">

<img name="preview" src="images/img1.jpg" />

</div>

</div>

</body>

</html>

About Us.html :

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>About Us</title>

<style>

body {

margin:0;

padding:0;

line-height: 1.5em;

}

#header {

background: black;

font-family: Arial;

height: 50px;

}

#contentwrapper {

float: left;

width: 100%;

}

#contentcolumn {

margin: 0 230px 0 230px;

}

#leftcolumn {

float: left;

width: 230px;

margin-left: -100%;

background: #24C8FF;

}

#rightcolumn {

float: left;

width: 230px;

margin-left: -230px;

background: #FFFB00;

}

.innertext {

margin: 20px;

font-family: Arial;

color: #5E5E5E;

}

#footer {

clear: left; /\* Cannot float on Left \*/

width: 100%;

background: black;

color: #FFF;

font-family: Arial;

text-align: center;

padding: 4px;

}

/\* ###### Responsive layout CSS ###### \*/

@media (max-width: 840px){ /\* Drop Right Column Down \*/

#leftcolumn {

margin-left: -100%;

}

#rightcolumn {

float: none;

width: 100%;

margin-left: 0;

clear: both;

}

#contentcolumn {

margin-right: 0; /\*Set margin to LeftColumnWidth\*/

}

}

@media (max-width: 600px){ /\* Drop Left Column Down \*/

#leftcolumn {

float: none;

width: 100%;

clear: both;

margin-left: 0;

}

#contentcolumn {

margin-left: 0;

}

}

</style>

</head>

<body>

<div id="container">

<div id="header">

<p style="color:white;text-align:center;font-size:40px;padding:10px">About Us</p>

</div>

<div id="contentwrapper">

<div id="contentcolumn">

<div class="innertext">

<b>Welcome!</b> <br /> Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

</div>

</div>

</div>

<div id="leftcolumn">

<div class="innertext">

<b>Left Column: <em>230px</em></b><br /> It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

</div>

</div>

<div id="rightcolumn">

<div class="innertext">

<b>Right Column: <em>230px</em></b><br /> Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

</div>

</div>

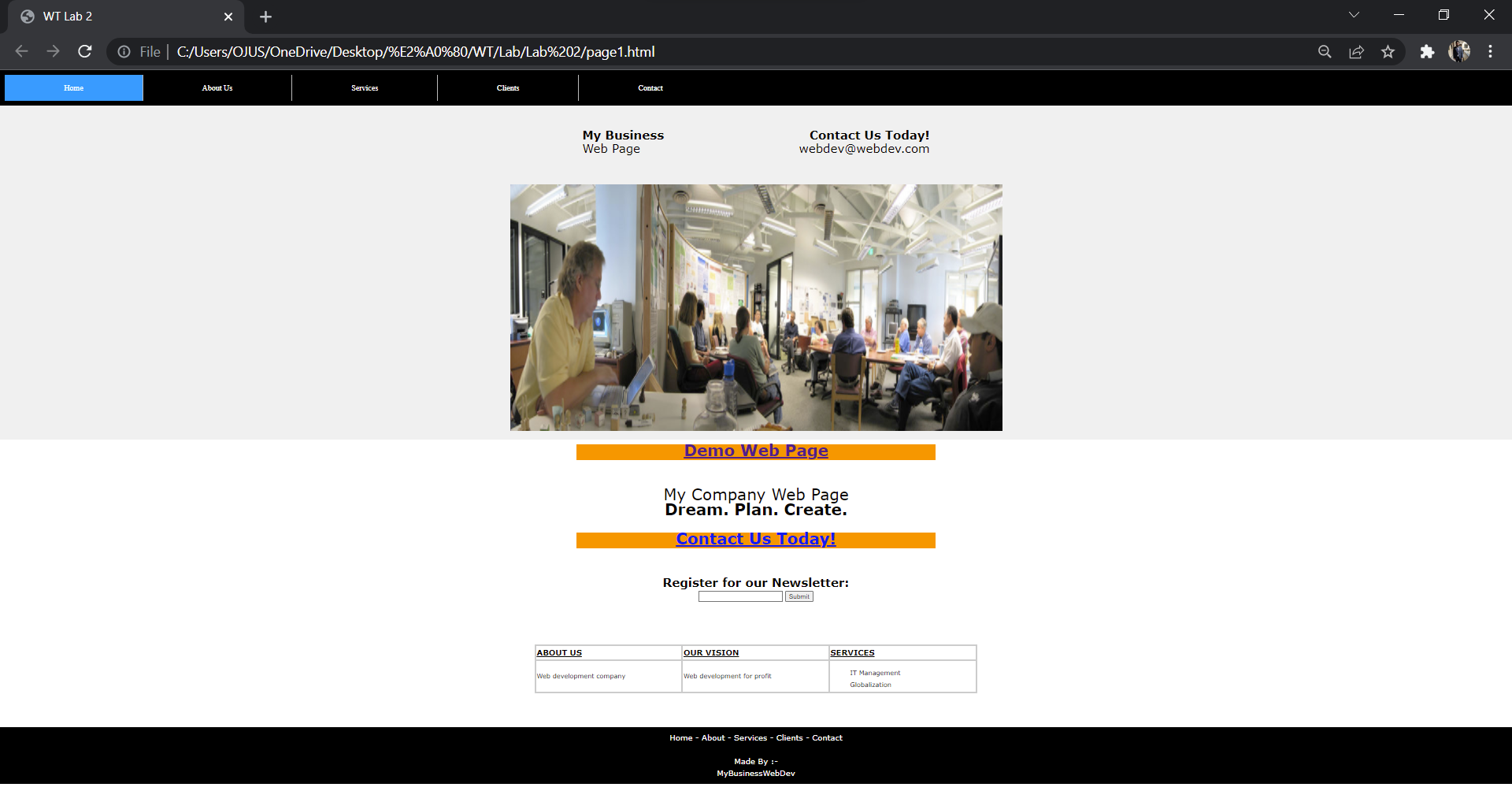
<div id="footer">Contact Now for Web Development</div>

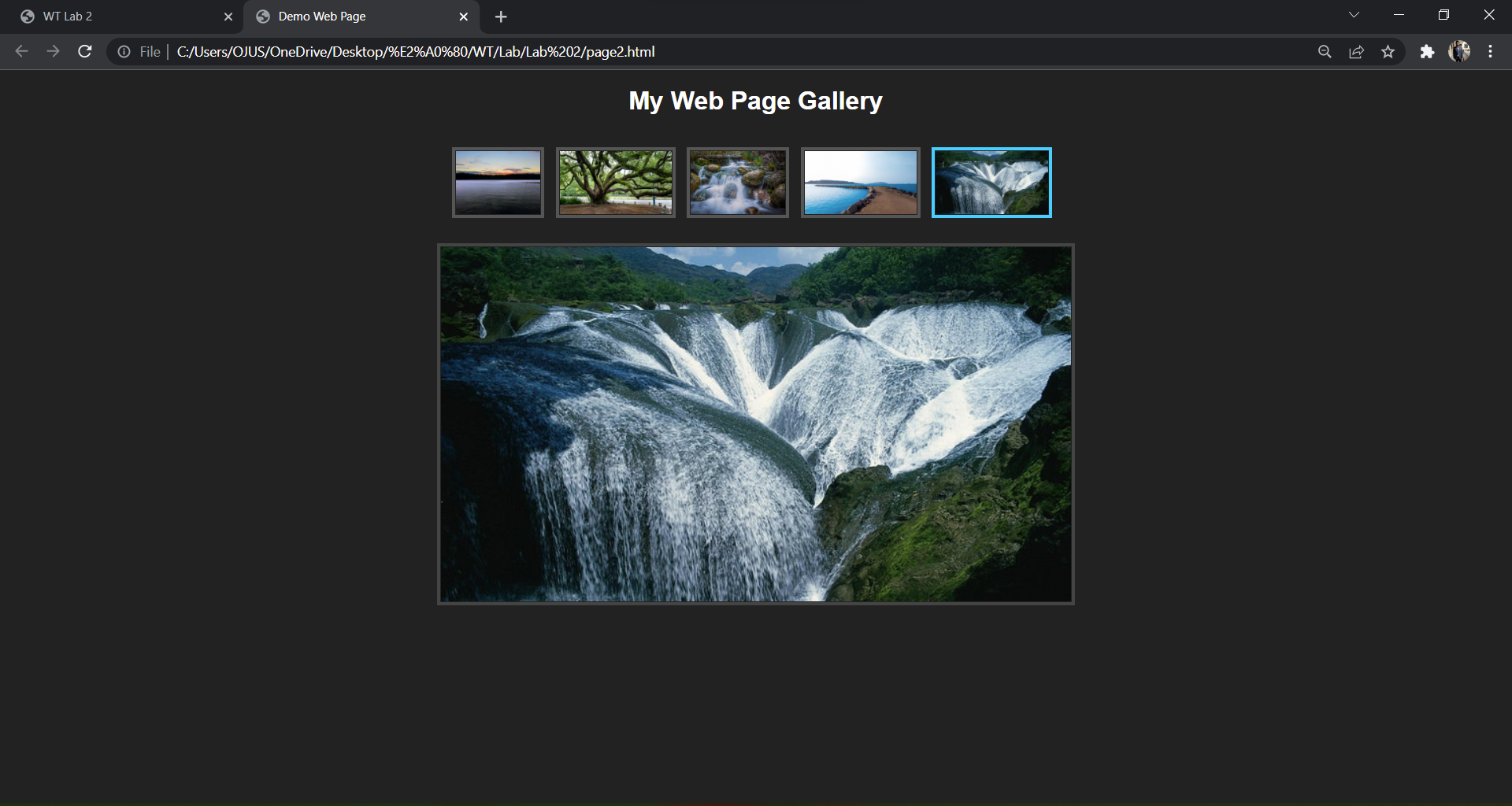
</div>

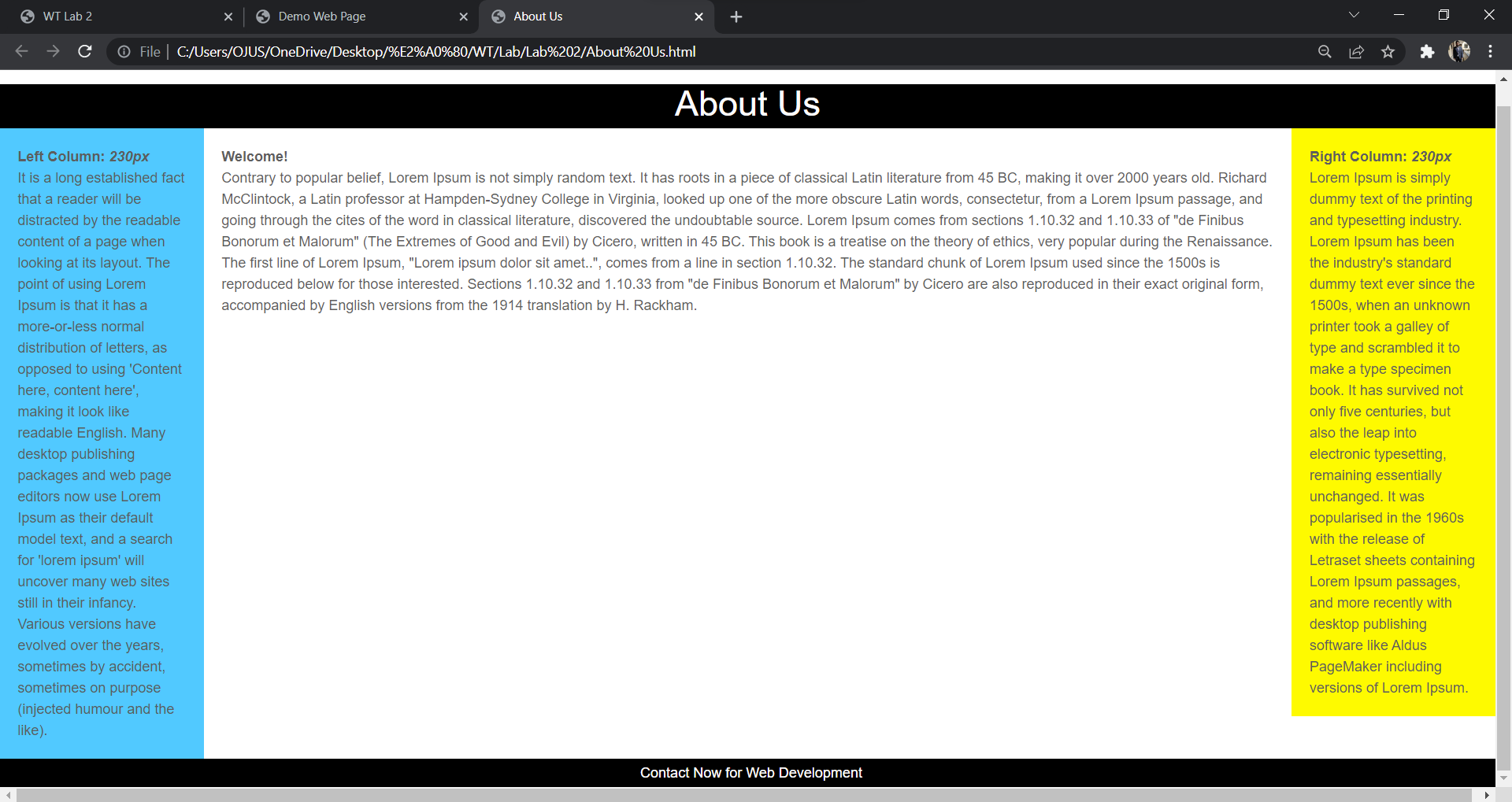
</body>

</html>

Output :-







Practical No. 3

Input :-

Employee.xml :

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="C:\Users\Acer\Downloads\style.xsl"?>

<!DOCTYPE catalog [

<!ELEMENT catalog (cd)>

<!ELEMENT cd (Emp\_Name,Address,Age,company,Experience)>

<!ELEMENT Emp\_Name (#PCDATA)>

<!ELEMENT Address (#PCDATA)>

<!ELEMENT Age (#PCDATA)>

<!ELEMENT company (#PCDATA)>

<!ELEMENT Experience (#PCDATA)>

]>

<catalog>

<cd>

<Emp\_Name>Rohan</Emp\_Name>

<Address>Ravet</Address>

<Age>26</Age>

<company>Infosys</company>

<Experience>10 years</Experience>

</cd>

<cd>

<Emp\_Name>Riya</Emp\_Name>

<Address>Akurdi</Address>

<Age>25</Age>

<company>TCS</company>

<Experience>9 year</Experience>

</cd>

<cd>

<Emp\_Name>Meena</Emp\_Name>

<Address>Chinchwad</Address>

<Age>30</Age>

<company>RCA</company>

<Experience>11 year</Experience>

</cd>

<cd>

<Emp\_Name>Meena</Emp\_Name>

<Address>Chinchwad</Address>

<Age>30</Age>

<company>RCA</company>

<Experience>11 year</Experience>

</cd>

<cd>

</catalog>

Style.xsl :

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2>Employee Details</h2>

<table border="1">

<tr bgcolor="Red">

<th style="text-align:left">Emp\_Name</th>

<th style="text-align:left">Address</th>

</tr>

<xsl:for-each select="catalog/cd">

<tr>

<td><xsl:value-of select="Emp\_Name"/></td>

<td><xsl:value-of select="Address"/></td>

</tr>

</xsl:for-each>

</table>

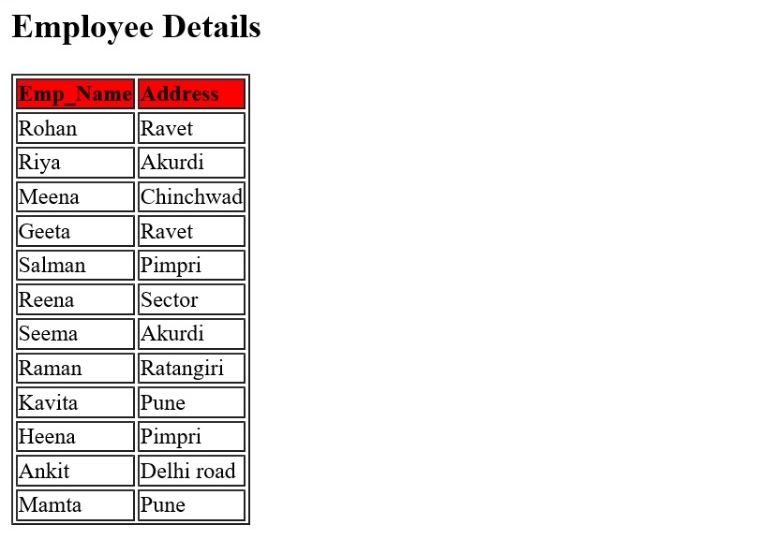
</body>

</html>

</xsl:template>

</xsl:stylesheet>

Output :-



Practical No. 4

Input :-

<!DOCTYPE html>

<html>

<head>

<title>

Calculator Program in JavaScript

</title>

<script>

function insert(num)

{

document.form1.textview.value = document.form1.textview.value + num;

}

function equal()

{

var exp = document.form1.textview.value;

if(exp)

{

document.form1.textview.value = eval(exp)

}

}

function backspace()

{

var exp = document.form1.textview.value;

document.form1.textview.value = exp.substring(0, exp.length - 1); /\* remove the element from total length ? 1 \*/

}

</script>

<style>

.formstyle

{

width: 300px;

height: 330px;

margin: 20px auto;

border: 3px solid skyblue;

border-radius: 5px;

padding: 20px;

text-align: center;

background-color: grey;

}

h1 {

text-align: center;

padding: 23px;

background-color: skyblue;

color: white;

}

input:hover

{

background-color: green;

}

\*{

margin: 0;

padding: 0;

}

.btn{

width: 50px;

height: 50px;

font-size: 25px;

margin: 2px;

cursor: pointer;

background-color: red;

color: white;

}

.textview{

width: 223px;

margin: 5px;

font-size: 25px;

padding: 5px;

background-color: lightgreen;

}

</style>

</head>

<body>

<h1> Calculator Program in JavaScript </h1>

<div class= "formstyle">

<form name = "form1">

<input class= "textview" name = "textview">

</form>

<center>

<table >

<tr>

<td> <input class = "btn" type = "button" value = "C" onclick = "form1.textview.value = ' ' " > </td>

<td> <input class = "btn" type = "button" value = "B" onclick = "backspace()" > </td>

<td> <input class = "btn" type = "button" value = "/" onclick = "insert('/')" > </td>

<td> <input class = "btn" type = "button" value = "x" onclick = "insert('\*')" > </td>

</tr>

<tr>

<td> <input class = "btn" type = "button" value = "7" onclick = "insert(7)" > </td>

<td> <input class = "btn" type = "button" value = "8" onclick = "insert(8)" > </td>

<td> <input class = "btn" type = "button" value = "9" onclick = "insert(9)" > </td>

<td> <input class = "btn" type = "button" value = "-" onclick = "insert('-')" > </td>

</tr>

<tr>

<td> <input class = "btn" type = "button" value = "4" onclick = "insert(4)" > </td>

<td> <input class = "btn" type = "button" value = "5" onclick = "insert(5)" > </td>

<td> <input class = "btn" type = "button" value = "6" onclick = "insert(6)" > </td>

<td> <input class = "btn" type = "button" value = "+" onclick = "insert('+')" > </td>

</tr>

<tr>

<td> <input class = "btn" type = "button" value = "1" onclick = "insert(1)" > </td>

<td> <input class = "btn" type = "button" value = "2" onclick = "insert(2)" > </td>

<td> <input class = "btn" type = "button" value = "3" onclick = "insert(3)" > </td>

<td rowspan = 5> <input class = "btn" style = "height: 110px" type = "button" value = "=" onclick = "equal()"> </td>

</tr>

<tr>

<td colspan = 2> <input class = "btn" style = "width: 106px" type = "button" value = "0" onclick = "insert(0)" > </td>

<td> <input class = "btn" type = "button" value = "." onclick = "insert('.')"> </td>

</tr>

</table>

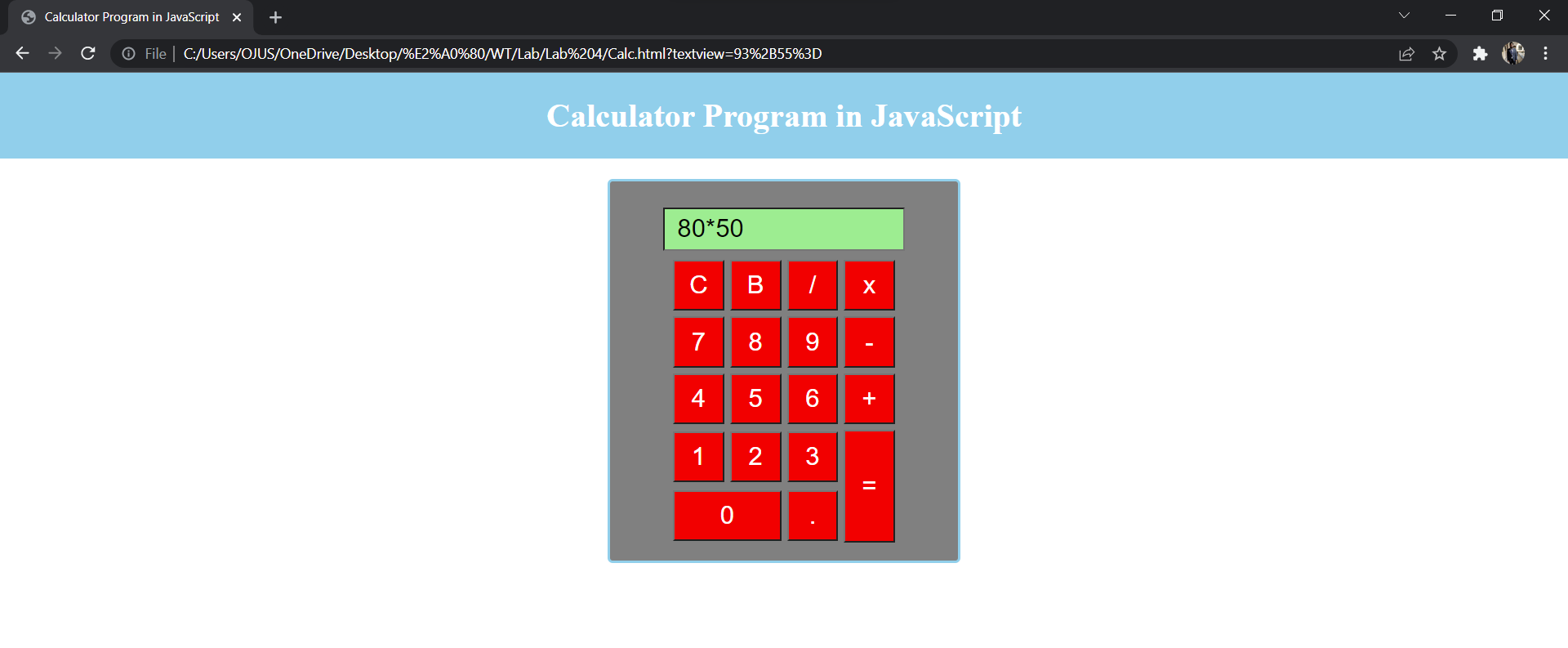
</center>

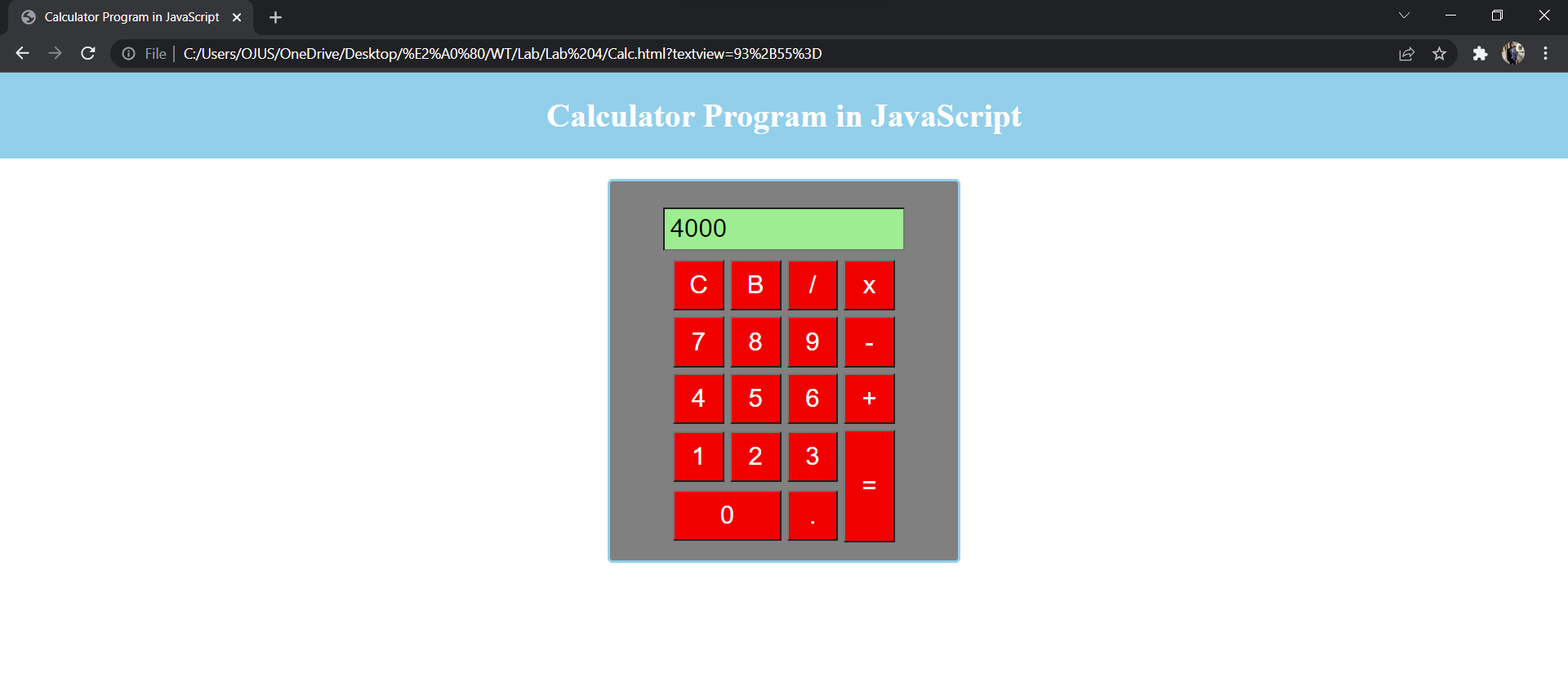
</div>

</body>

</html>

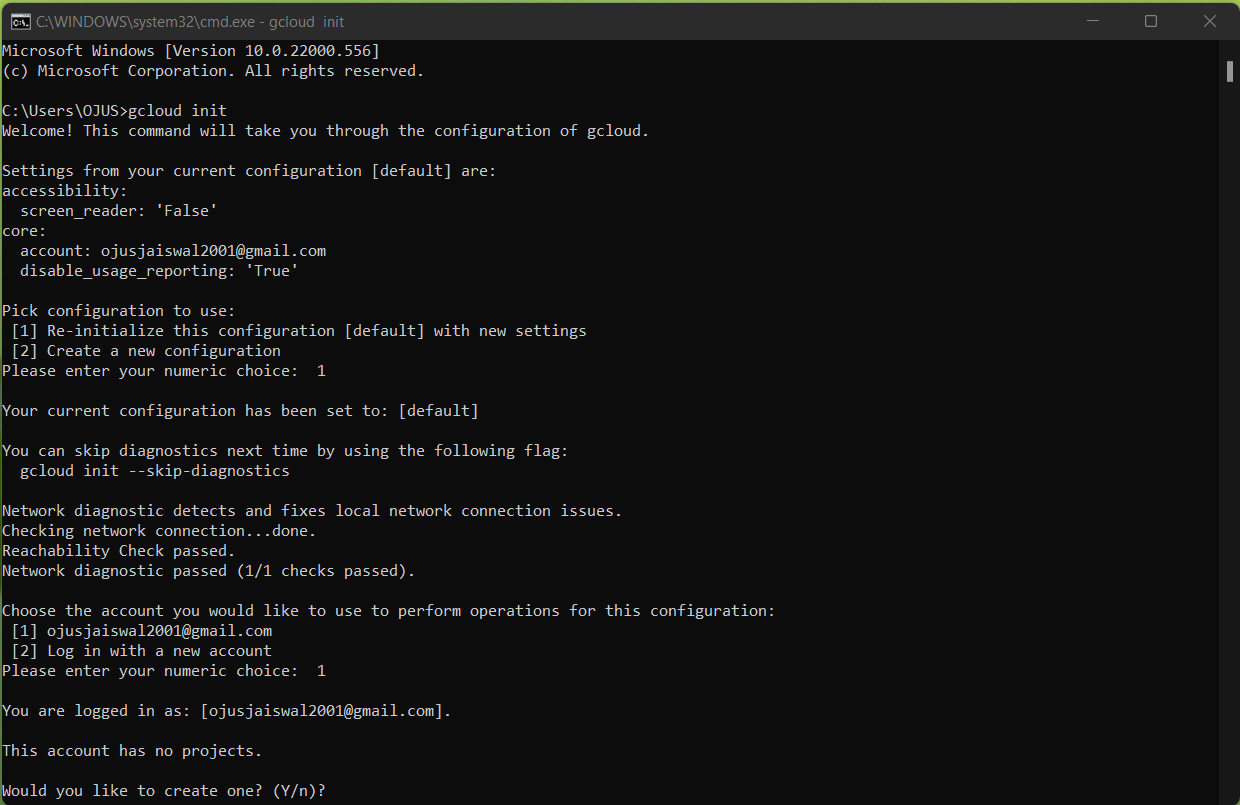
Output :-





Practical No. 2

Output :-



Practical No. 3

Output :-

