1



**Amrita School of Engineeing Amaravati**

**Department of Mathematics**

**Discrete Lab Manual**

**23MAT116**

**Submitted by: Verified By:**

Name: A.Vijay Kumar Dr.Srinivasarao Thota

Roll No: 24002 Assistant Professor of Mathematics

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Title** | **PG.NO** | **SIGNATURE** |
| **1** | Write Mat Lab program to generate a truth table that consists of 3 statements: p,q,r. |  |  |
| **2** | Write recursive program for Fibonacci series in Mat Lab. |  |  |
| **3** | Implement the binary search as a recursive function in Mat Lab. |  |  |
| **4** | Write a Mat Lab program for permutation and combinations. Apply this implementation to the following problem. How many ways are there to select five players from a10-member tennis team to make a trip to a match at another school? |  |  |
| **5** |  |  |  |
| **6** | Create a directed graph using an edge list, and then find the equivalent adjacency matrix representation of the graph. |  |  |
| **7** | Create a graph using an edge list, and then calculate the graph incidence matrix. |  |  |
| **8** | Create a directed graph using an edge list, and then calculate the incidence matrix. |  |  |
| **9** | Create and plot a graph, and then find the degree of each node. |  |  |
| **10** | Create and plot a directed graph. Calculate the shortest path between nodes. |  |  |
| **11** | Create and plot a graph with weight edges. Find the shortest path between nodes, and specify two outputs to also return the length of the path. |  |  |

* **Question:1**

Write MatLab program to generate a truth table that consists of 3 statements: *p*, *q*, *r*.

**Solution:**

% Generate all combinations of p, q, r

[p, q, r] = ndgrid([0 1]);

% Flatten the grids into column vectors

p = p(:);

q = q(:);

r = r(:);

% Combine into a matrix

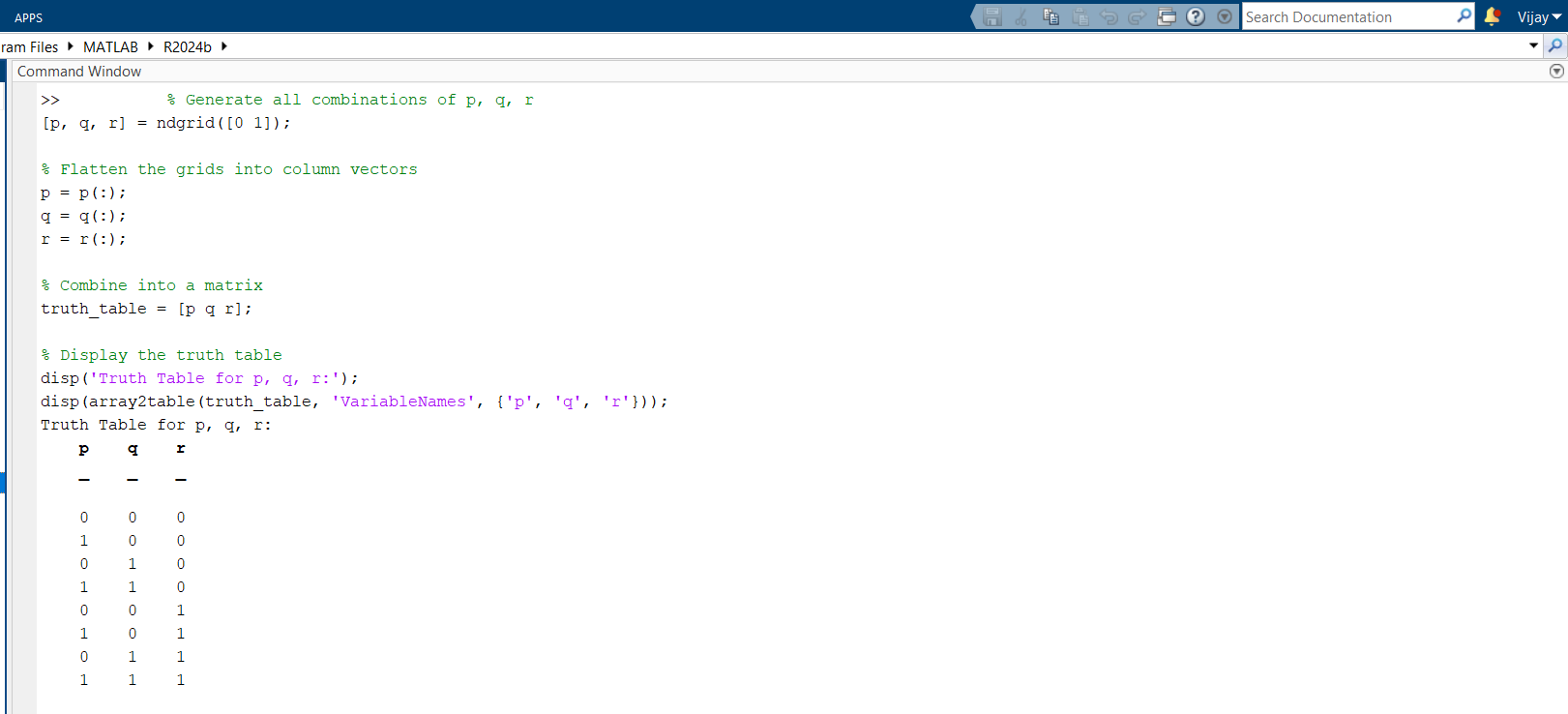
truth\_table = [p q r];

% Display the truth table

disp('Truth Table for p, q, r:');

disp(array2table(truth\_table, 'VariableNames', {'p', 'q', 'r'}));

**Output:**

****

* **Question:2**

Write recursive program for Fibonacci series in MatLab.

**Solution:**

function result = fibonacci(n)

% Recursive function to compute the nth Fibonacci number

if n == 0

result = 0;

elseif n == 1

result = 1;

else

result = fibonacci(n - 1) + fibonacci(n - 2);

end

end

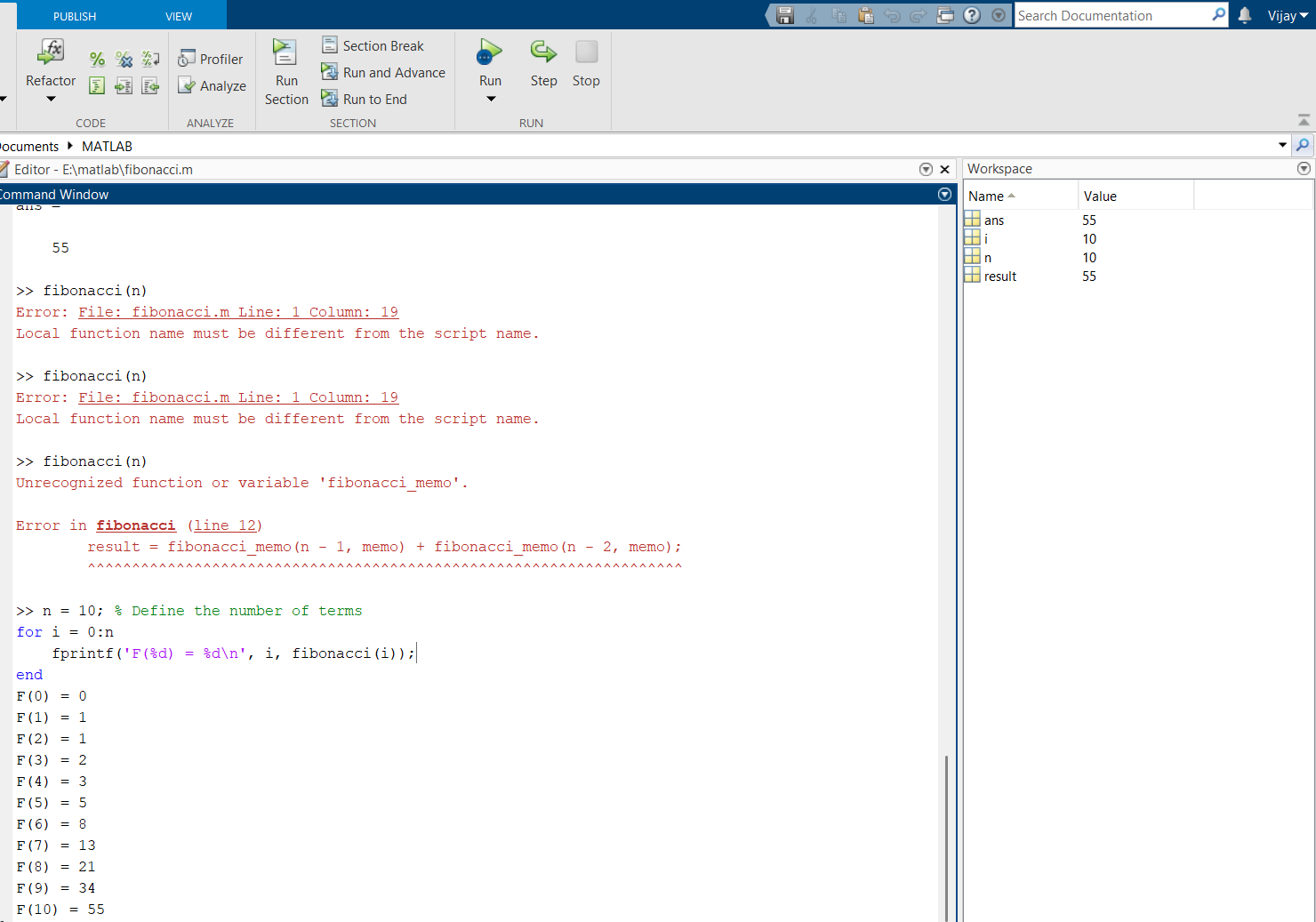
n = 10; % Define the number of terms

for i = 0:n

fprintf('F(%d) = %d\n', i, fibonacci(i));

end

**Output:**

****

* **Question:3**

Implement the binary search as a recursive function in Ma Lab.

**Solution:**

function idx = binarySearch(arr, key, low, high)

if low > high

idx = -1; % Not found

else

mid = floor((low + high)/2);

if arr(mid) == key

idx = mid;

elseif arr(mid) > key

idx = binarySearch(arr, key, low, mid - 1);

else

idx = binarySearch(arr, key, mid + 1, high);

end

end

end

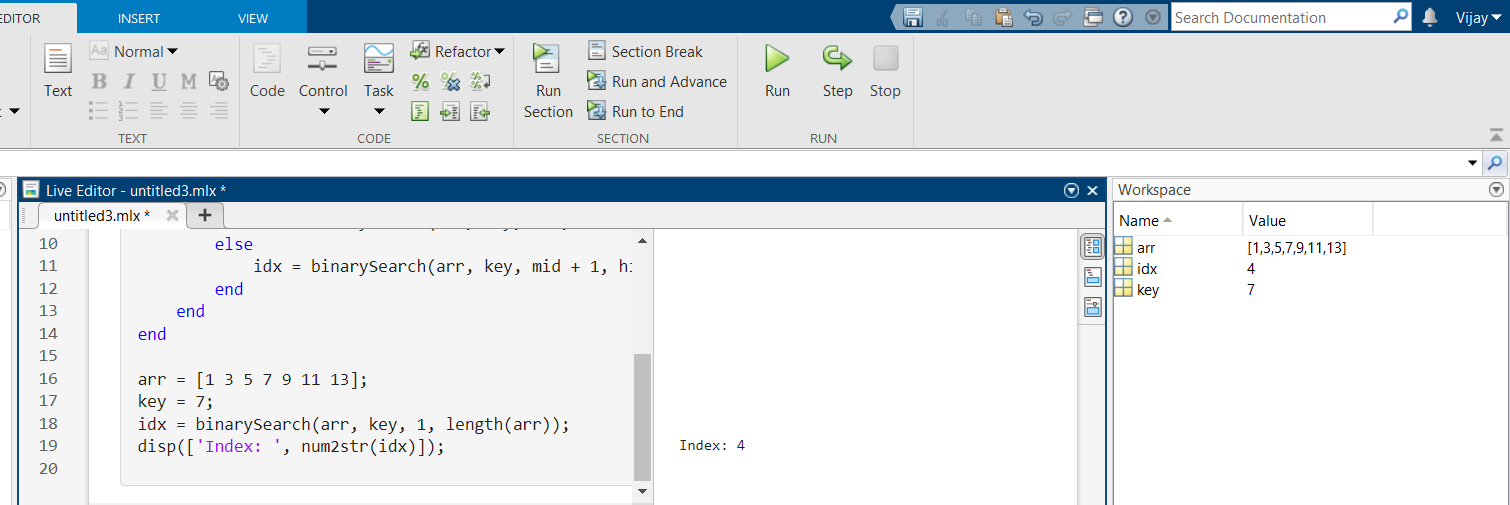
Ex: arr = [1 3 5 7 9 11 13];

key = 7;

idx = binarySearch(arr, key, 1, length(arr));

disp(['Index: ', num2str(idx)]);

**Output:**

****

* **Question:4**

Write a MatLab program for permutation and combinations. Apply this implement-

-tation to the following problem.

How many ways are there to select five players from a 10-member tennis team to

make a trip to a match at another school?

**Solution:**

n = 10;

r = 5;

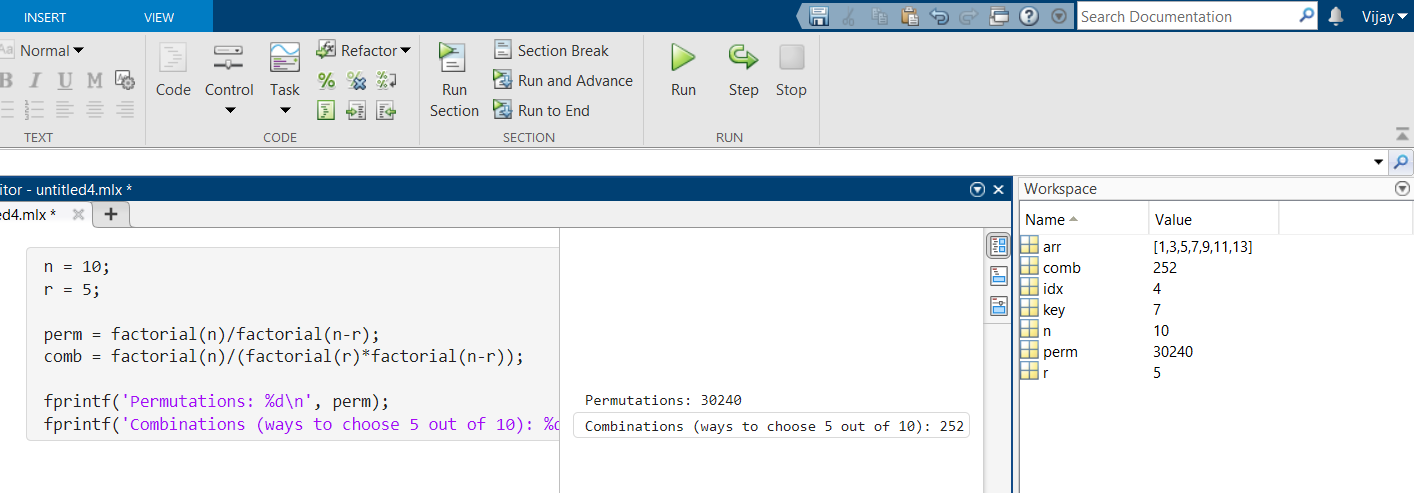
perm = factorial(n)/factorial(n-r);

comb = factorial(n)/(factorial(r)\*factorial(n-r));

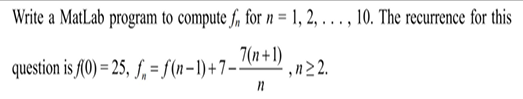
fprintf('Permutations: %d\n', perm);

fprintf('Combinations (ways to choose 5 out of 10): %d\n', comb);

**Output:**

****

* **Question:5**



**Solution:**

f = zeros(1, 11);

f(1) = 25;

f(2) = 7;

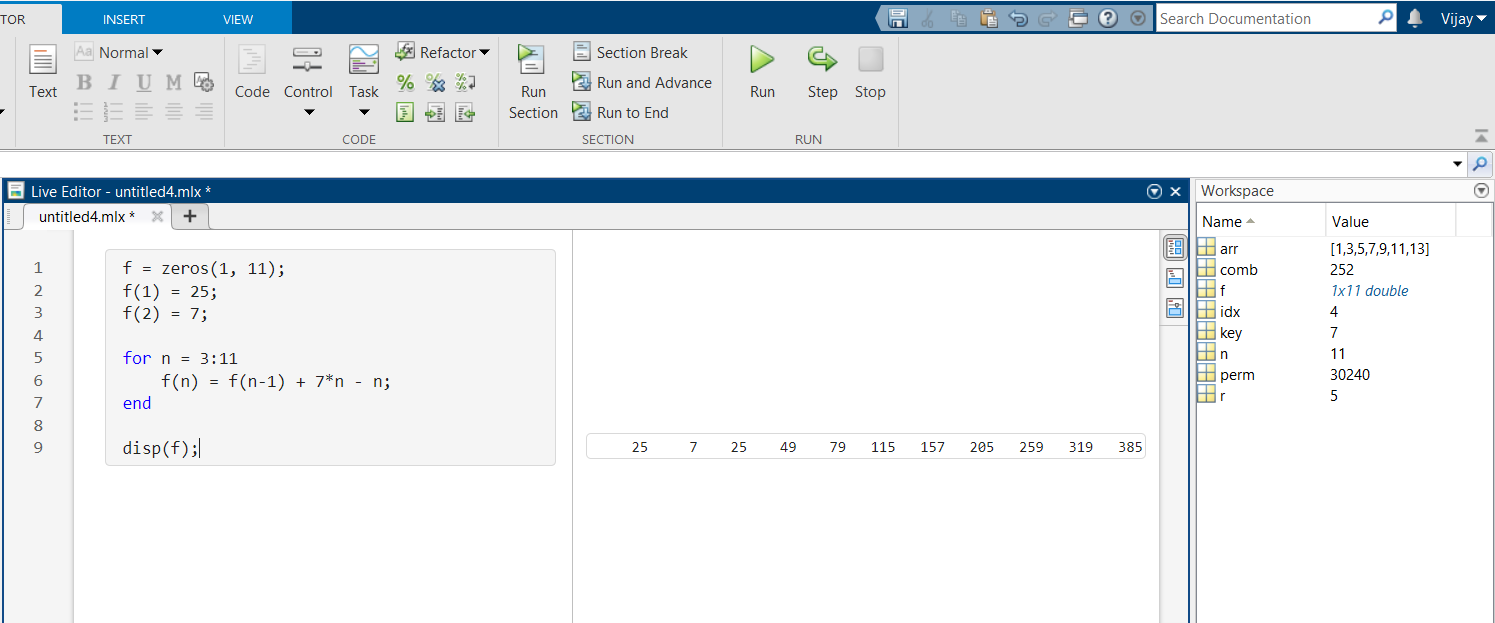
for n = 3:11

f(n) = f(n-1) + 7\*n - n;

end

disp(f);

**Output:**

****

* **Question:6**

Create a directed graph using an edge list, and then find the equivalent

adjacency matrix representation of the graph.

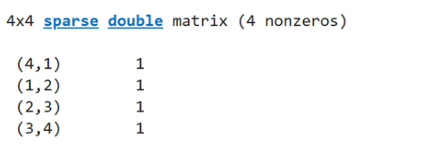
**Solution:**

edgeList = [1 2; 2 3; 3 4; 4 1];

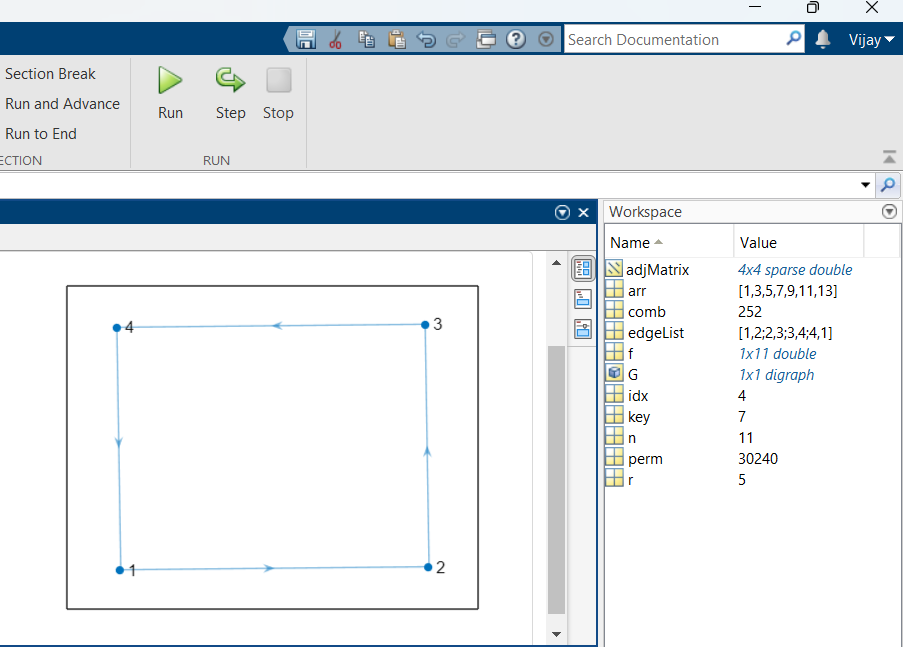
G = digraph(edgeList(:,1), edgeList(:,2));

adjMatrix = adjacency(G)

plot(G)

****

**Output:**

****

* **Question:7**

Create a graph using an edge list, and then calculate the graph incidence matrix.

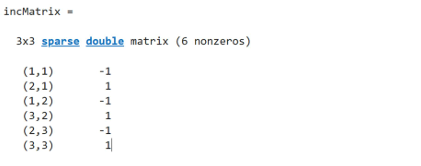
**Solution:**

edgeList = [1 2; 2 3; 3 1];

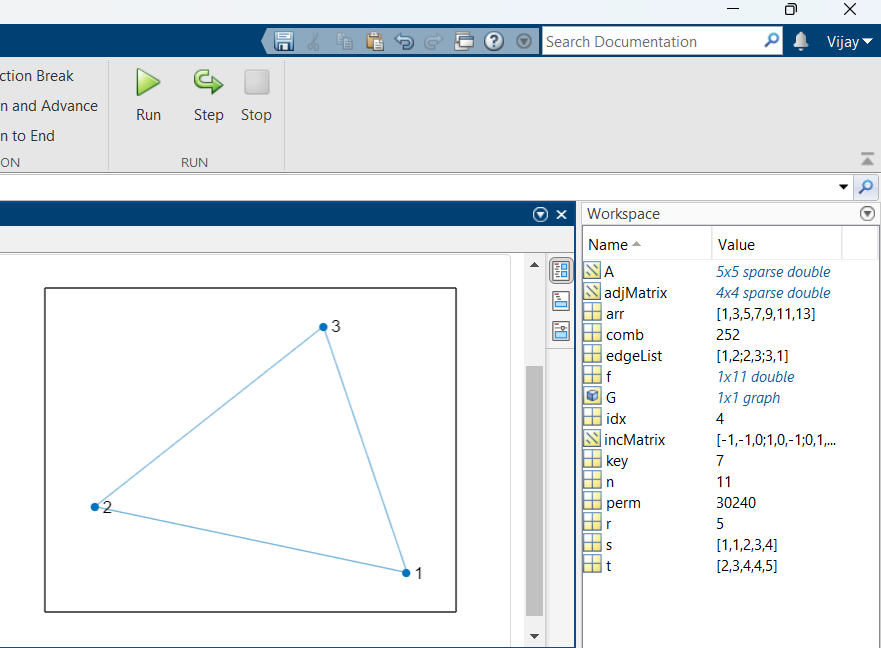
G = graph(edgeList(:,1), edgeList(:,2));

incMatrix = incidence(G)

plot(G)



**Output:**

****

* **Question:8**

Create a directed graph using an edge list, and then calculate the incidence matrix.

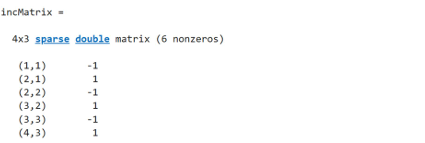
**Solution:**

edgeList = [1 2; 2 3; 3 4];

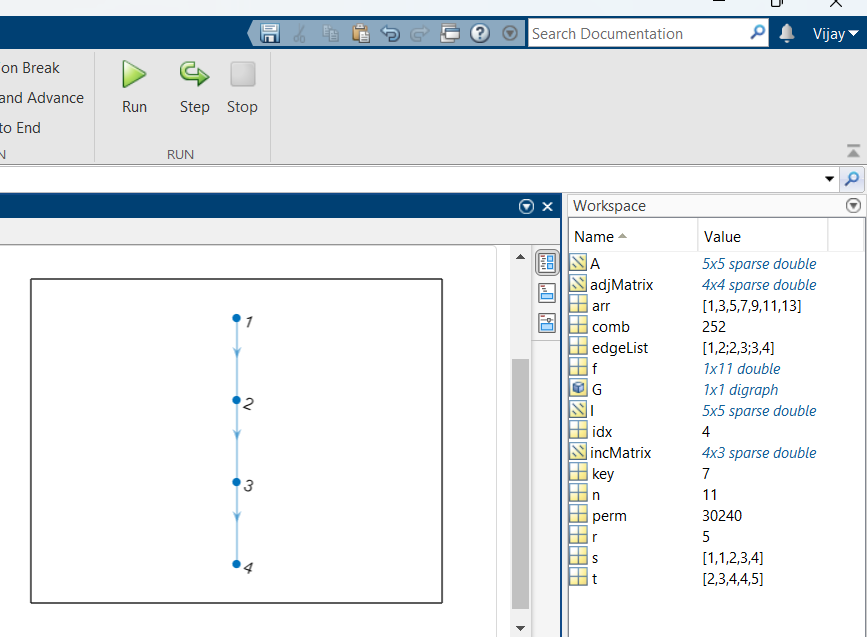
G = digraph(edgeList(:,1), edgeList(:,2));

incMatrix = incidence(G)

plot(G)

****

**Output:**



* **Question:9**

Create and plot a graph, and then find the degree of each node.

**Solution:**

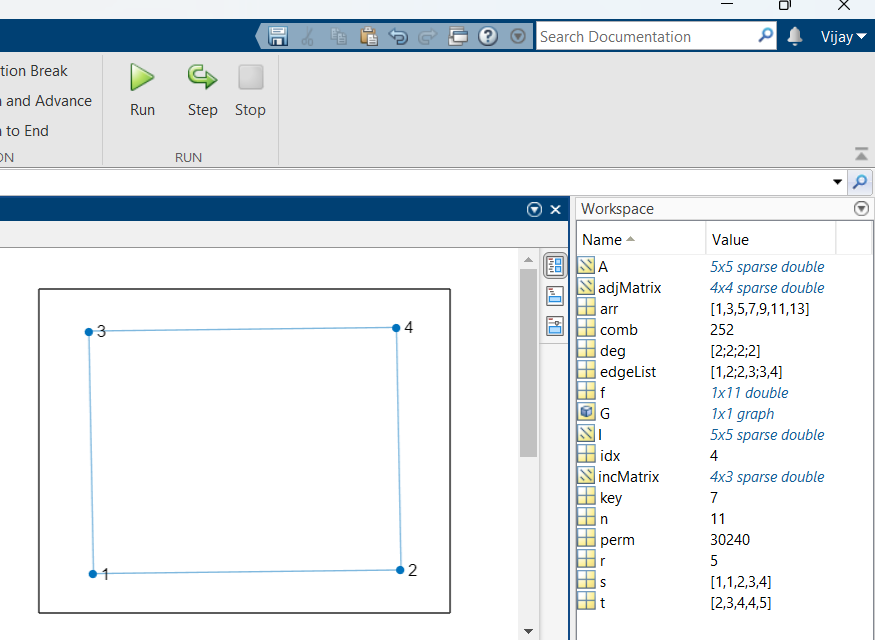
G = graph([1 1 2 3], [2 3 4 4]);

plot(G)

deg = degree(G)

****

**Output:**



* **Question:10**

Create and plot a directed graph. Calculate the shortest path between nodes.

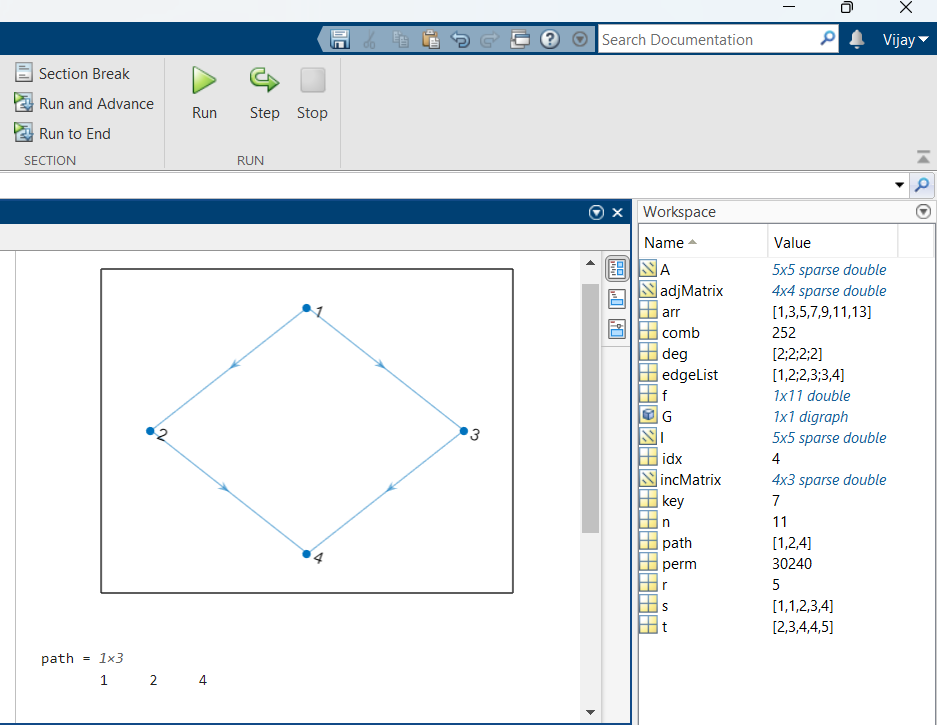
**Solution:**

G = digraph([1 1 2 3], [2 3 4 4]);

plot(G)

path = shortestpath(G, 1, 4)

**Output:**

****

* **Question:11**

Create and plot a graph with weighted edges. Find the shortest path between nodes,

and specify two outputs to also return the length of the path.

**Solution:**

s = [1 1 2 3];

t = [2 3 4 4];

weights = [2 3 1 5];

G = digraph(s, t, weights);

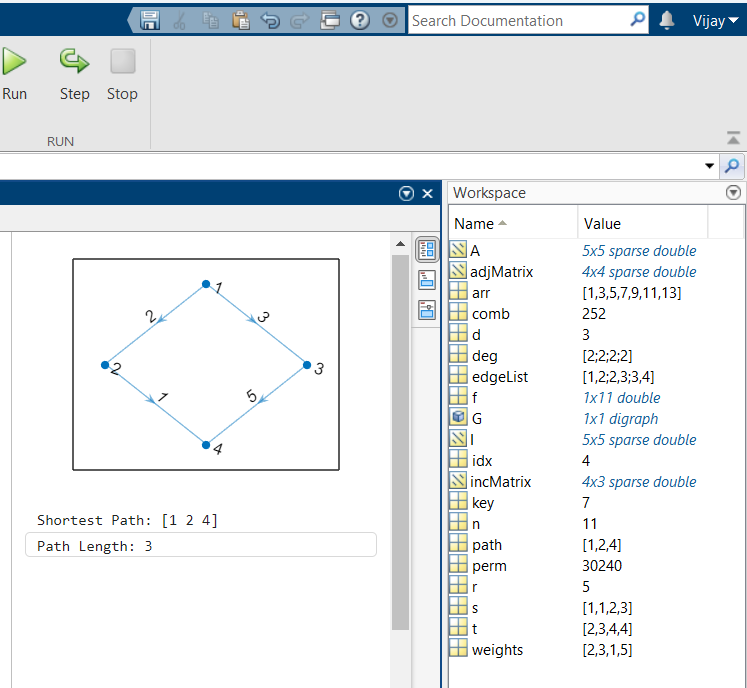
plot(G, 'EdgeLabel', G.Edges.Weight)

[path, d] = shortestpath(G, 1, 4);

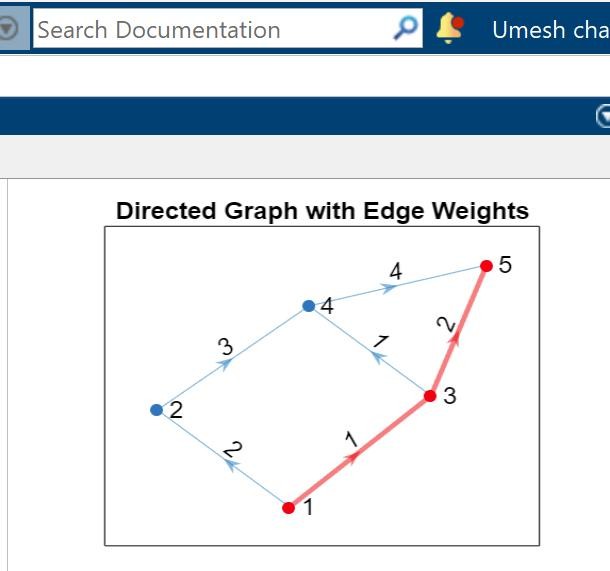
fprintf('Shortest Path: %s\n', mat2str(path));

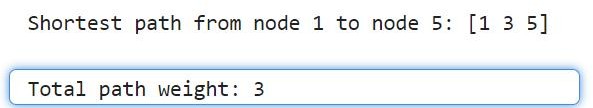
fprintf('Path Length: %d\n', d);

**Output:**



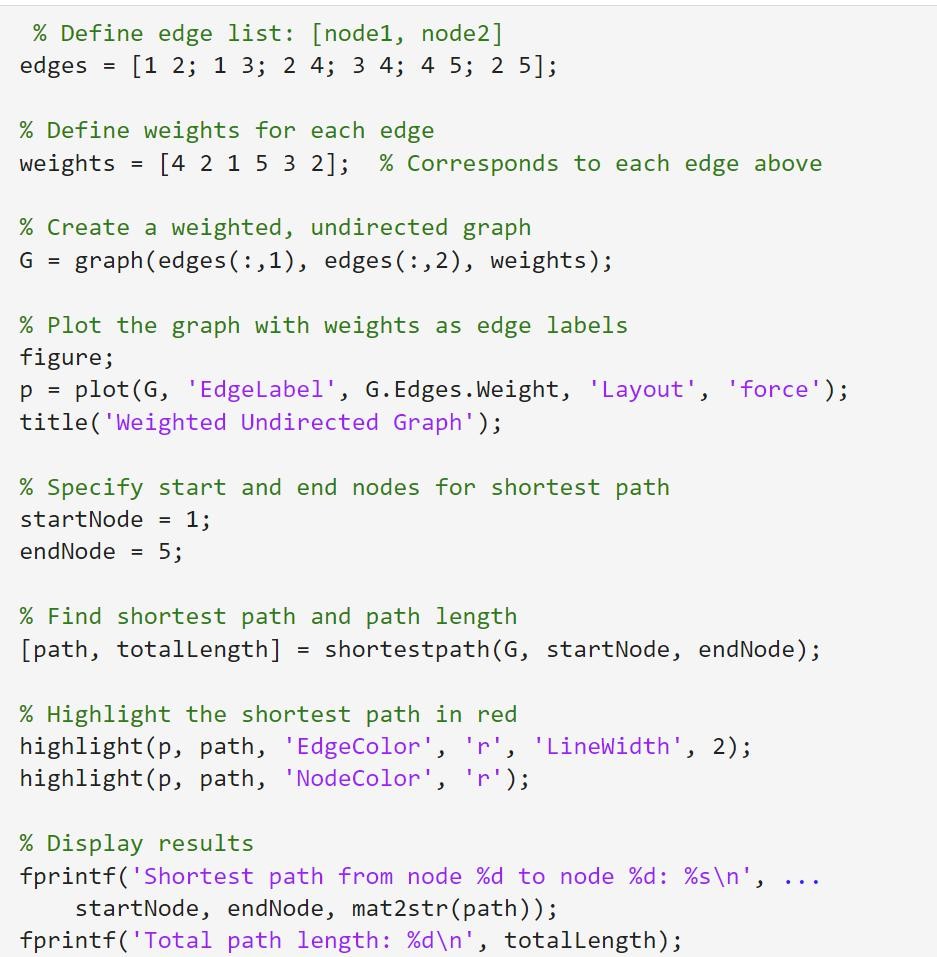
**OUTPUT:-**



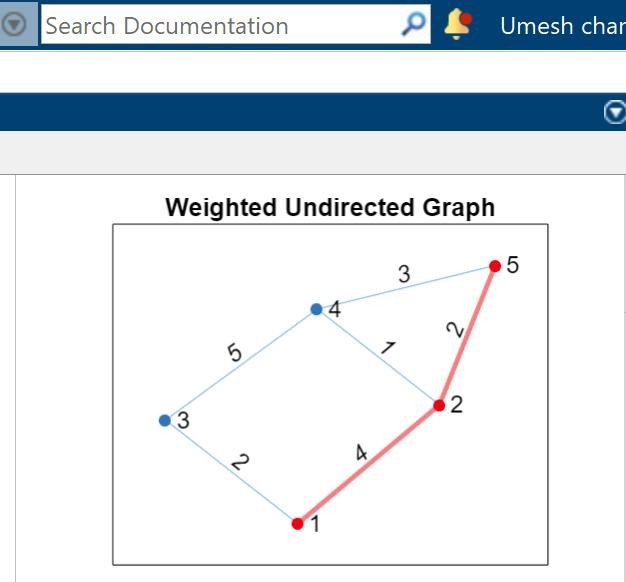
****

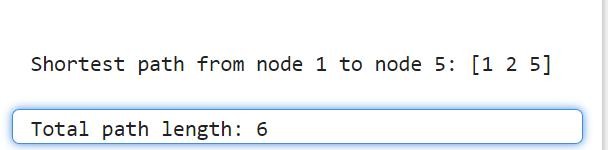
**QUESTION 11:-** Create and plot a graph with weighted edges. Find the shortest path between nodes, and specify two outputs to also return the length of the path.

**SOLUTION:-**



**OUTPUT:-**



****