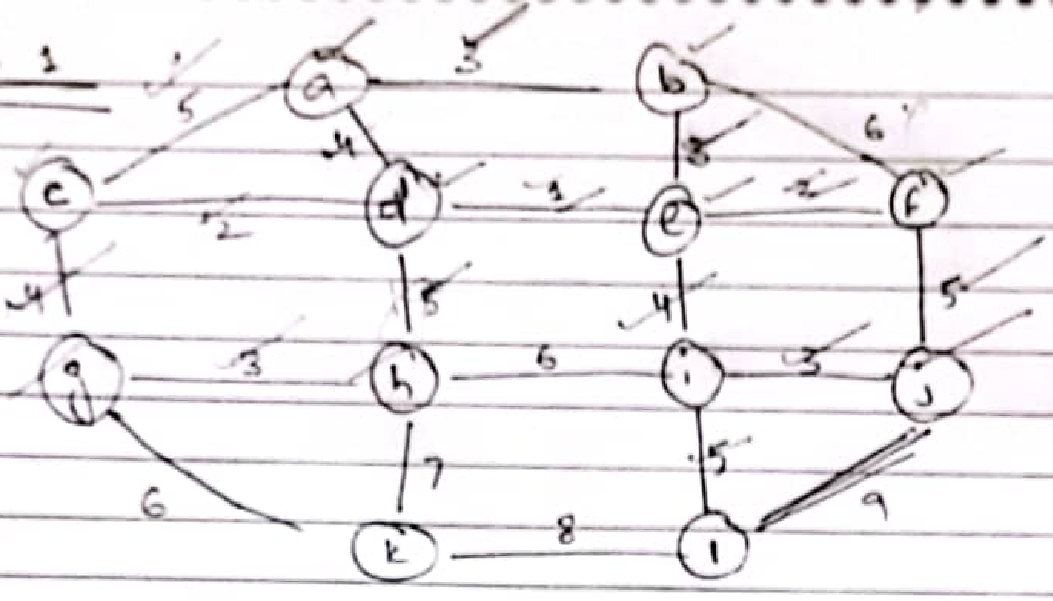
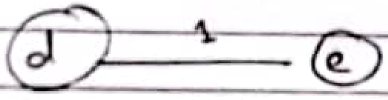
Question 1

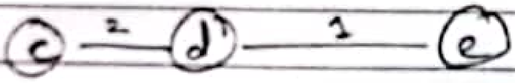
a

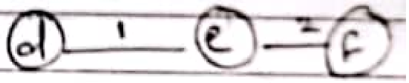


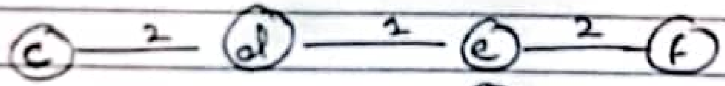
(a) Apply prims:

start with min edge: d – e 1

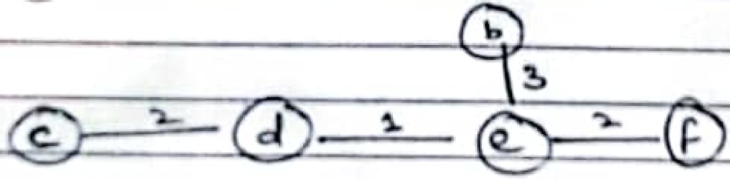
- 

- 

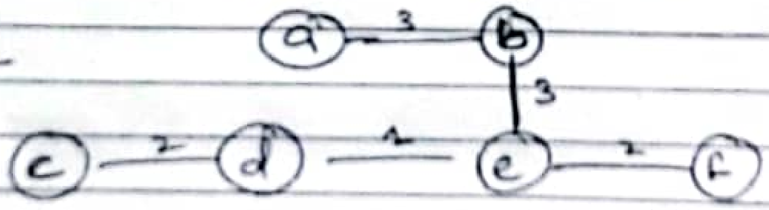
or


- 

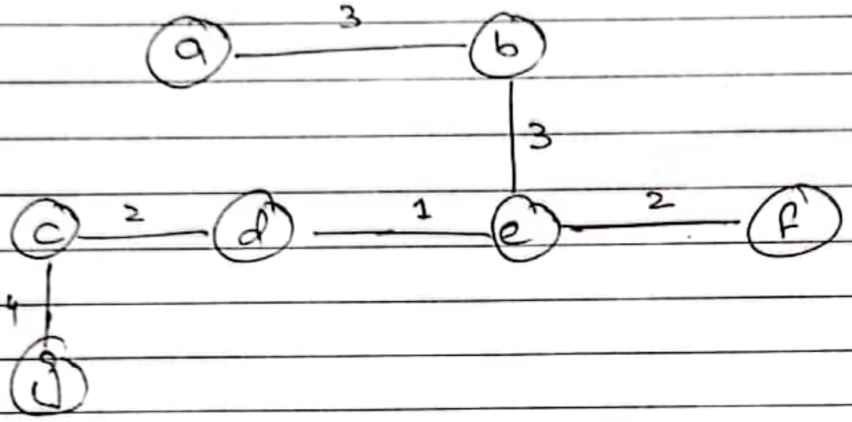
b – e

- 

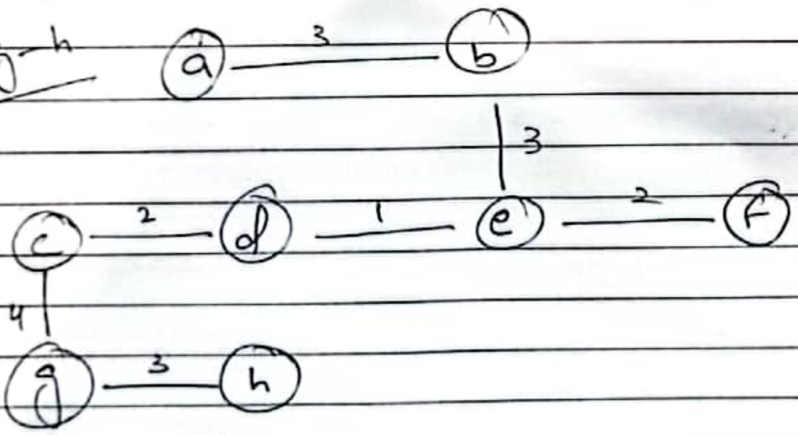
a – b

- 

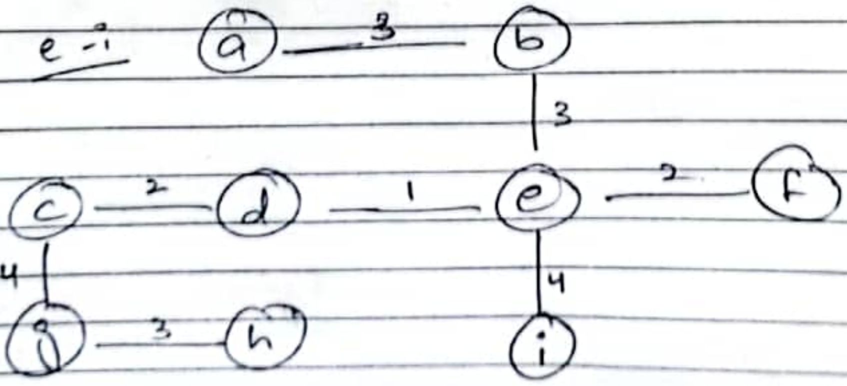
- e - i or c - g



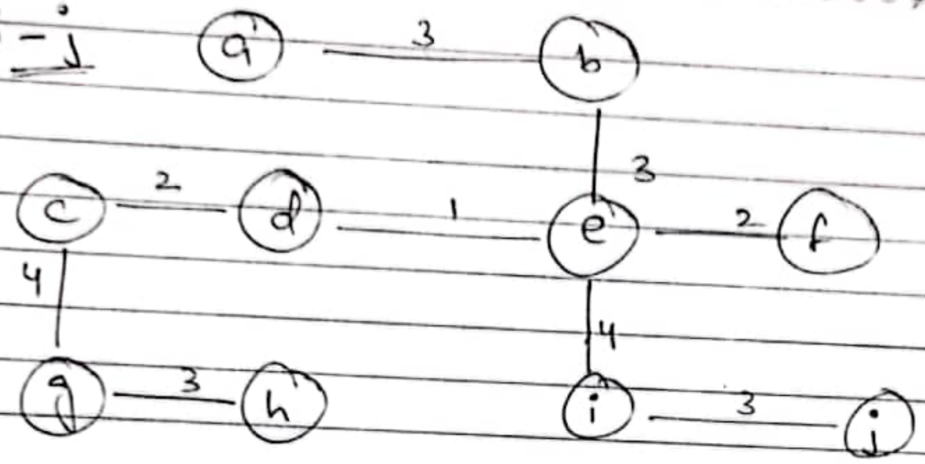
- g-h



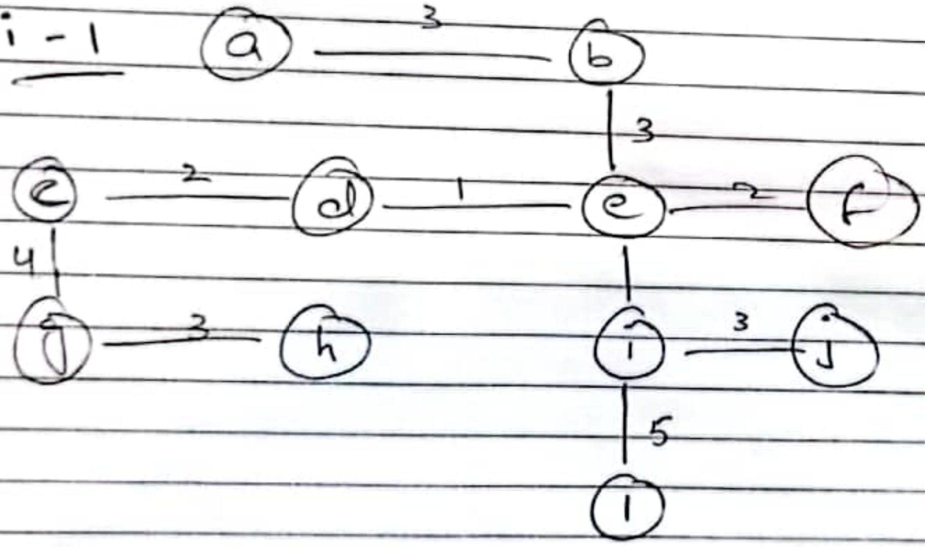
- e-i



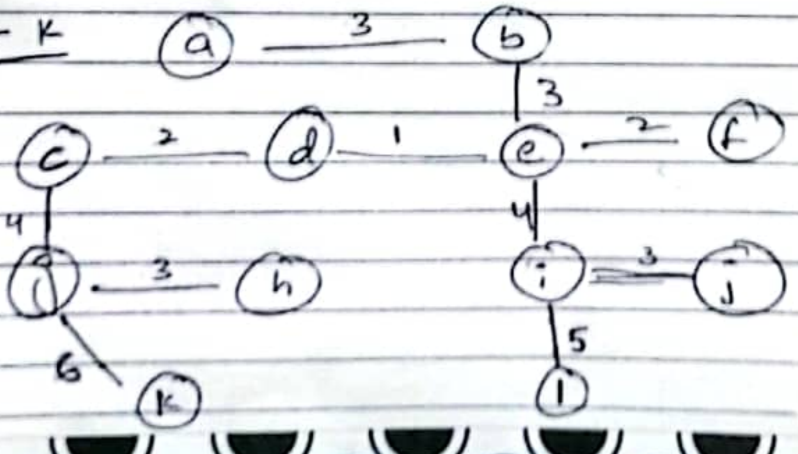
i-j



i-l



g-k



Total cost = 36

1 b)

- Yes we have to check connectivity of graph before applying prims algorithm.
- Prims will only generate spanning tree for connected components containing the start node.

Part :- (c)

5/5

No you cannot always construct an MST of Graph (New) by adding one new edge to T.

Reasons :-

Cycle Formation :-

Adding a new edge could form a cycle in the MST, requiring the removal of a higher weight edge to maintain a tree structure.

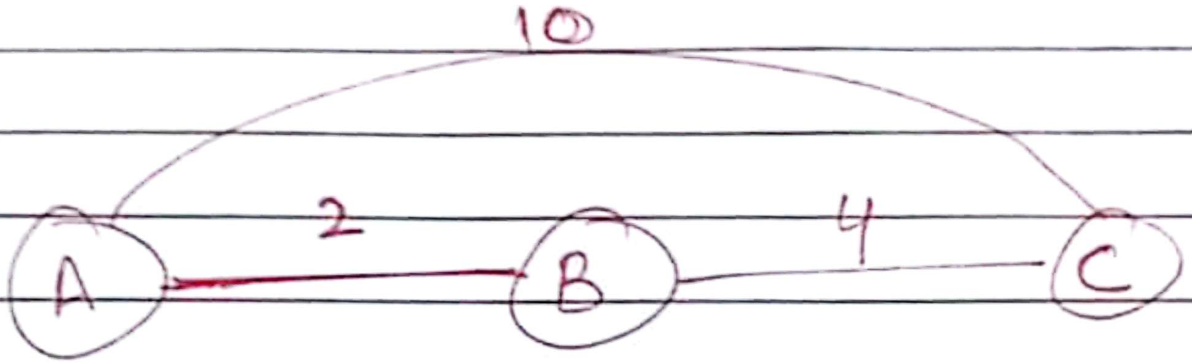
New minimum edge :-

The new edges might provide a better (low-weight) connection requiring changes to the existing MST.

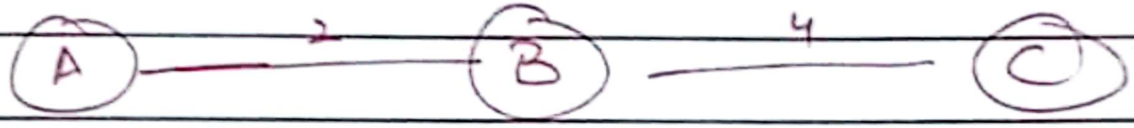
Correct Approach :-

To construct the MST of Graph (New) use Prim's or Kruskal algorithm on the entire graph Graph (New) as these algorithms will correctly handle the new vertex and edges.

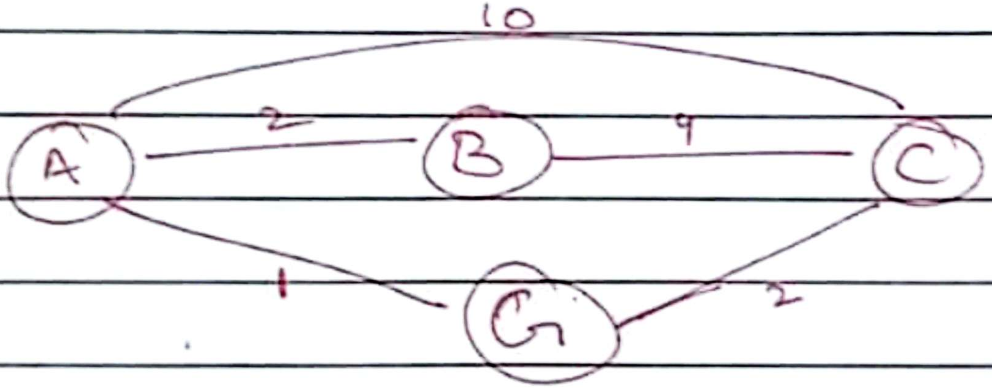
Previous graph



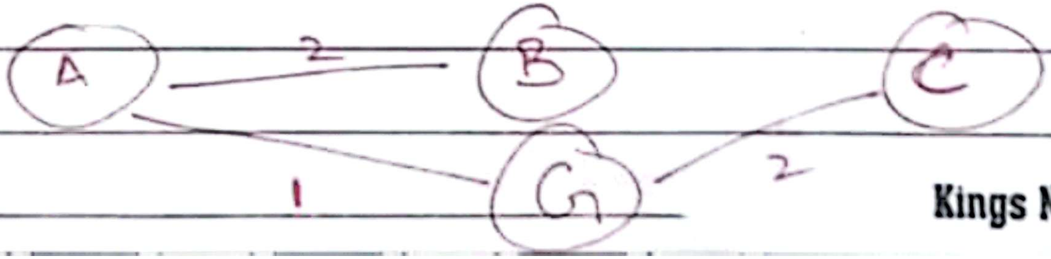
Tree



New graph

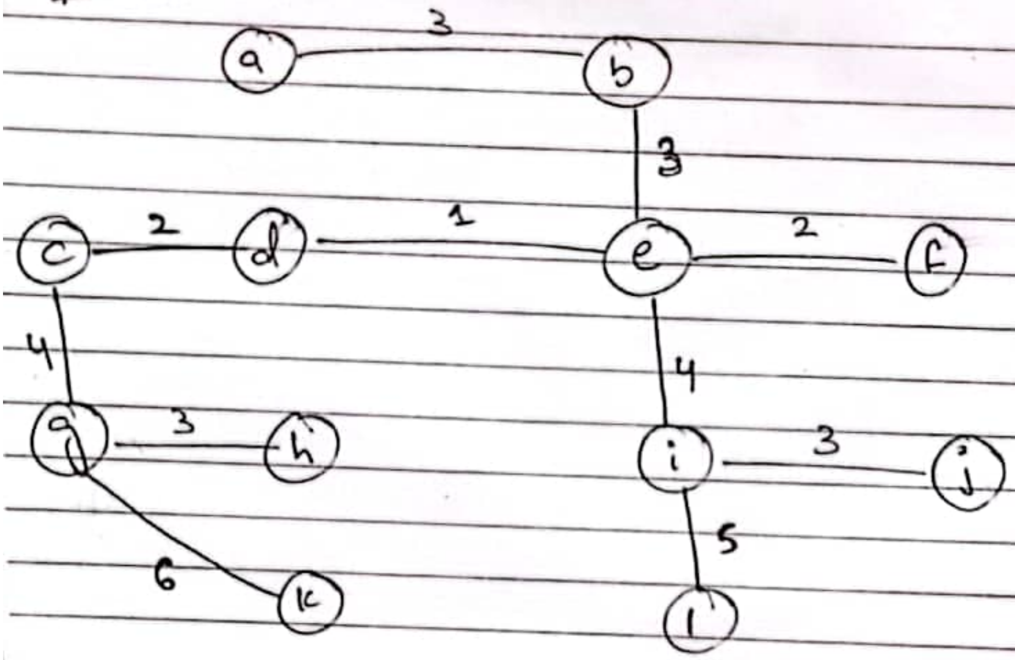


New Tree



ate: ....

Question 2



1) d - e : 1
2) e - f : 2
3) c - d : 2
4) a — b : 3
5) g - h : 3
6) i - j : 3
7) b - e : 3
8) c - g : 4
9) e - i : 4
10) a - d : 4 ⟶ form a cycle (don't include).
a - c : 5 ⟶ ″
f - j : 5 ⟶ ″
d - h : 5 ⟶ ″

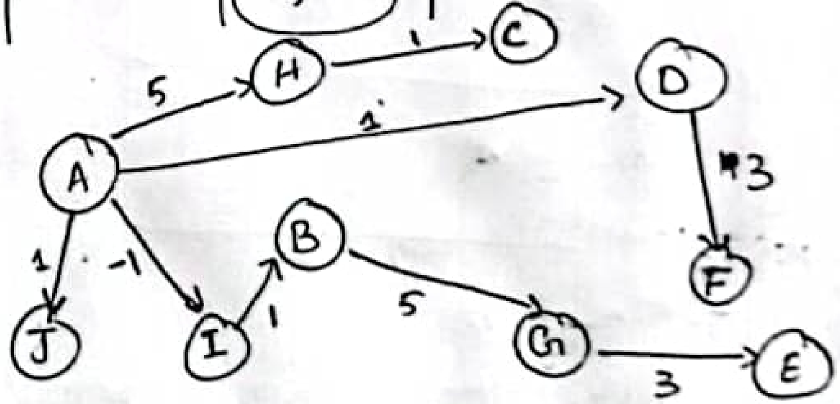
10) i - l : 5
11) b - f : 6 → cycle
12) h - i : 6 → cycle
g - k : 6

Total cost : 36

Question 3

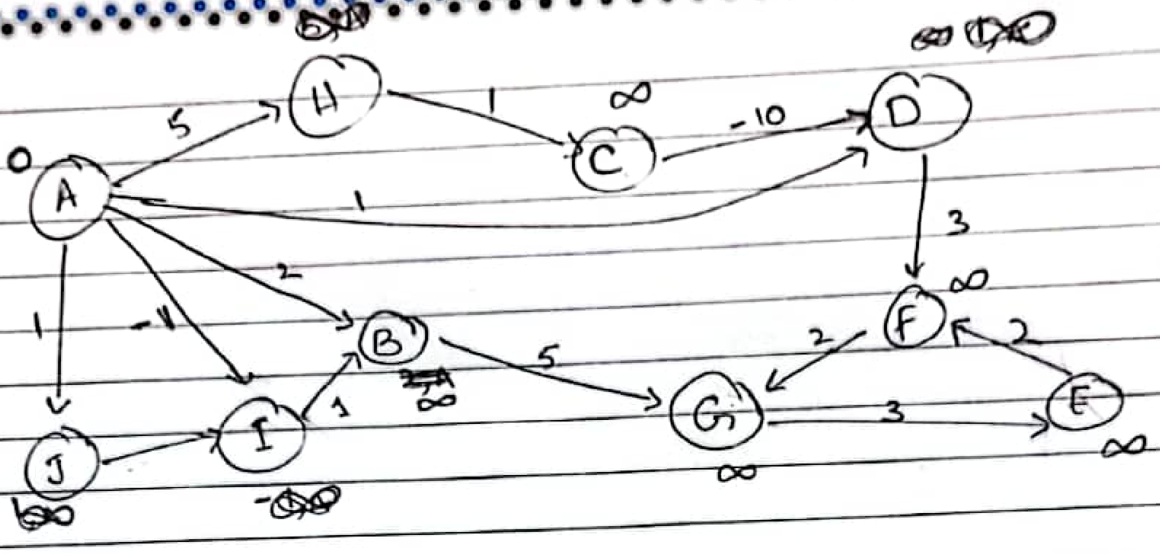
Dijkastra:

		B	C	D	E	F	G	H	I	J
step	N	d(B), p(B)	d(c), p(c)	d(D), p(D)	d(E), p(E)	d(F), p(F)	d(G), p(G)	d(H), p(H)	d(I), p(I)	d(J), p(J
0	A	~~2, A~~ ∞	∞	1, A ~~∞~~	∞	∞	∞	5, A	-1, A	1, A
1	AI	0, I	∞	1, A	∞	∞	∞	5, A		1, A
2	AIB		∞	1, A	∞	∞	5, B	5, A		1, A
3	AIBJ		∞	1, A	∞	∞	5, B	5, A		
4	AIBJD		∞	&	∞	4, D	5, B	5, A		
~~5~~	~~AIBJDG~~		∞		~~8, G~~	~~6,~~				
5	AIBJDF		∞		∞		5, B	5, A		
6	AIBJDFG		∞		8, G			5, A		
7	AIBJDFGH		6, H		8, G					
8	AIBJDFGHC				8, G					
9	AIBJDFGHCE									

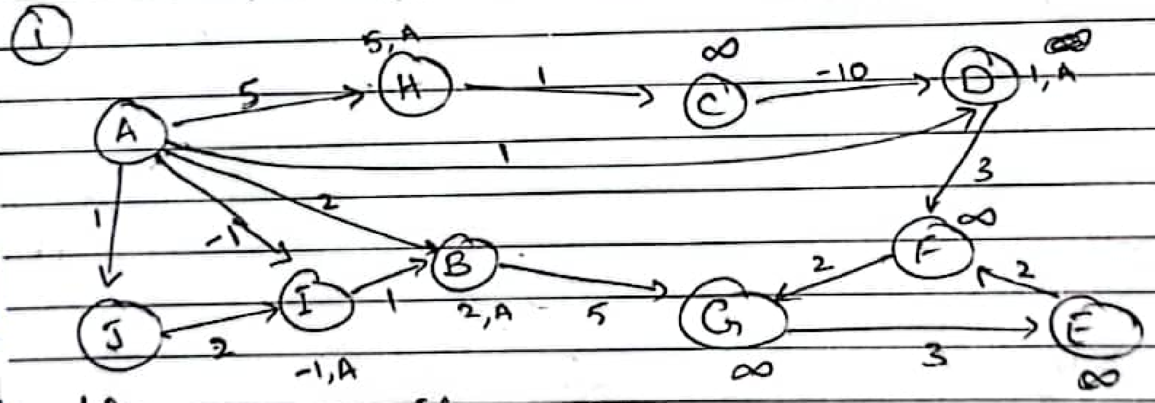


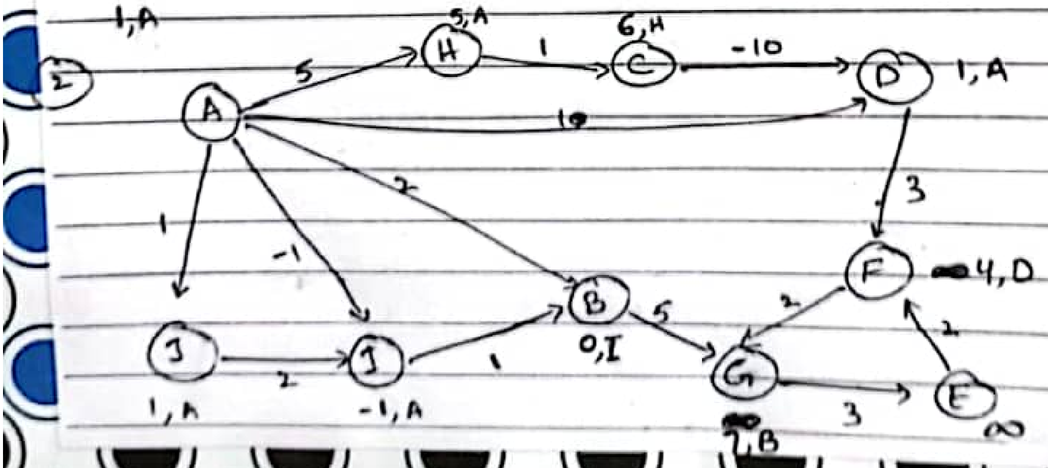
Date: ____________

Bellman ford.



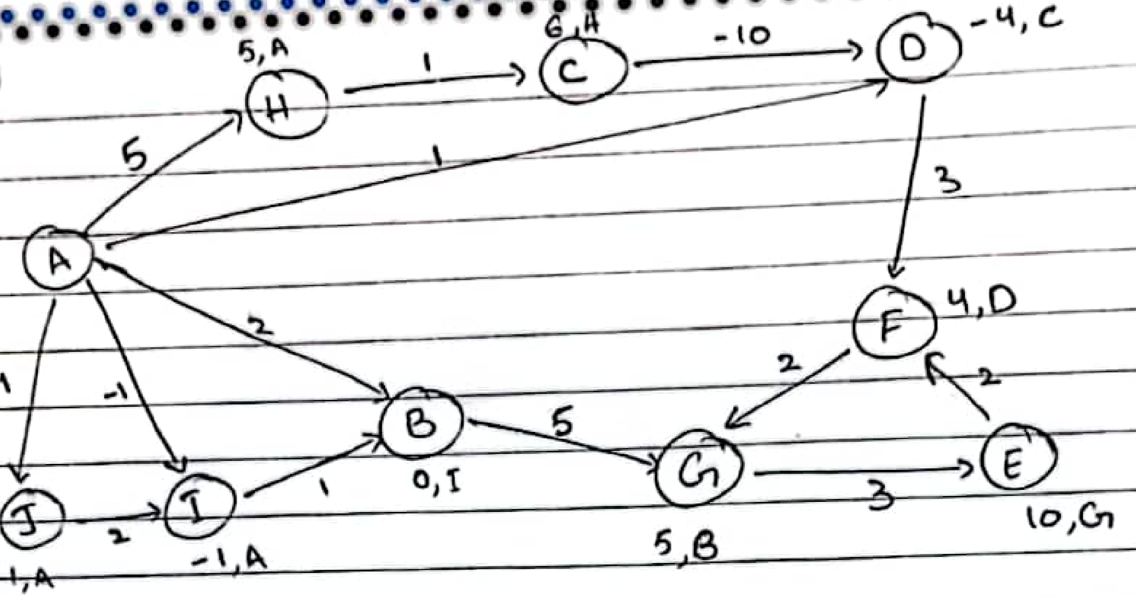
Edge List : (A,H), (A,J), (A,I), (A,B), (A,D), (J,I), (I,B), (B,G), (G,E), (E,F), (F,G), (H,C), (C,D), (D,F)





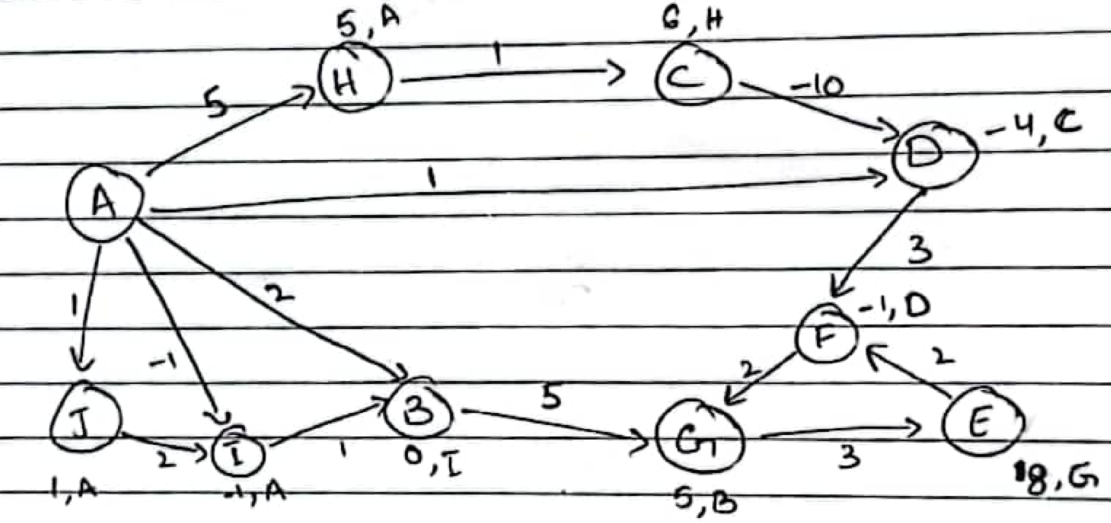
ate:

3



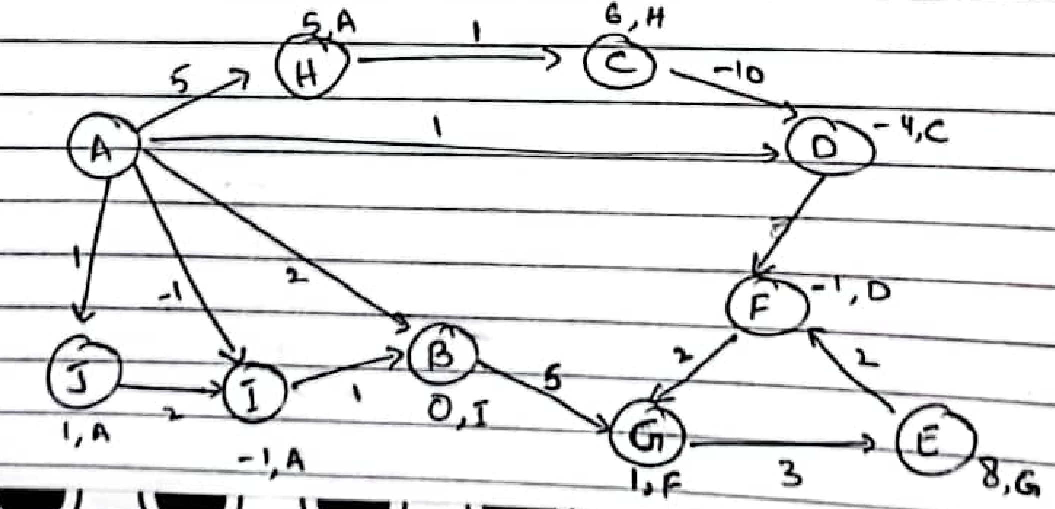
5,A
H
1
6,H
C
-10
D
-4,C
5
1
3
A
F
4,D
2
2
2
1
-1
B
5
G
E
1
0,I
3
J
I
10,G
2
5,B
1,A
-1,A

4



5,A
6,H
H
1
C
-10
5
D
-4,C
A
1
3
F
-1,D
2
1
-1
2
5
2
J
I
B
G
E
2
1
0,I
3
1,A
-1,A
5,B
18,G

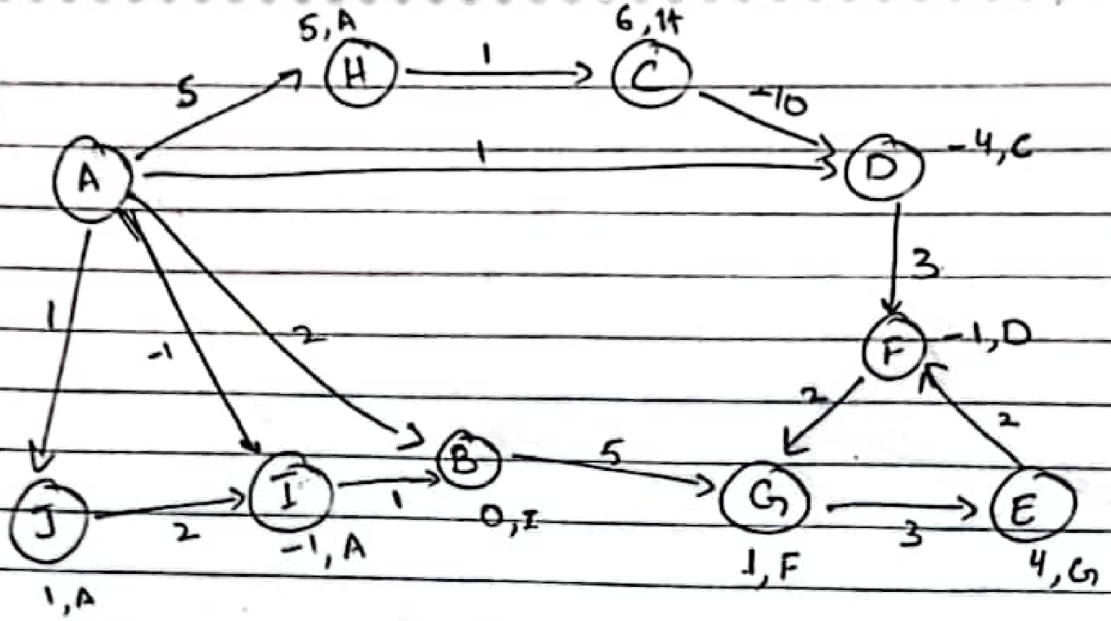
5



5,A
6,H
5
H
1
C
-10
A
1
D
-4,C
1
F
-1,D
-1
2
2
2
J
I
B
5
1
G
E
1,A
2
0,I
-1,A
1,F
3
8,G

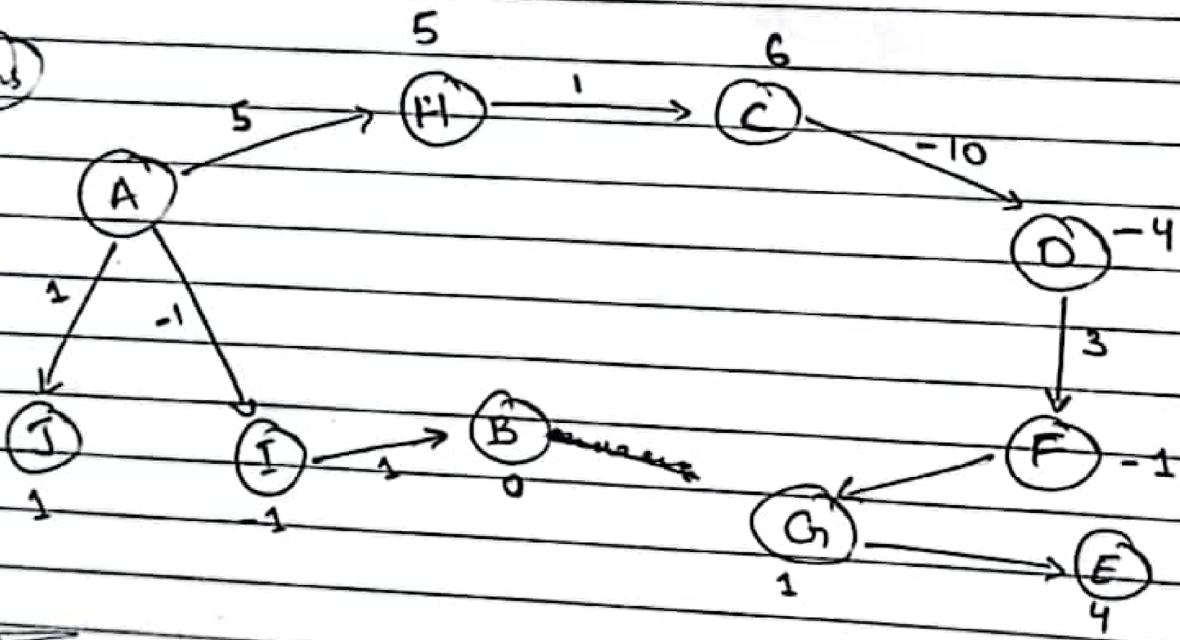
Date:

6



7 same as above

Ans

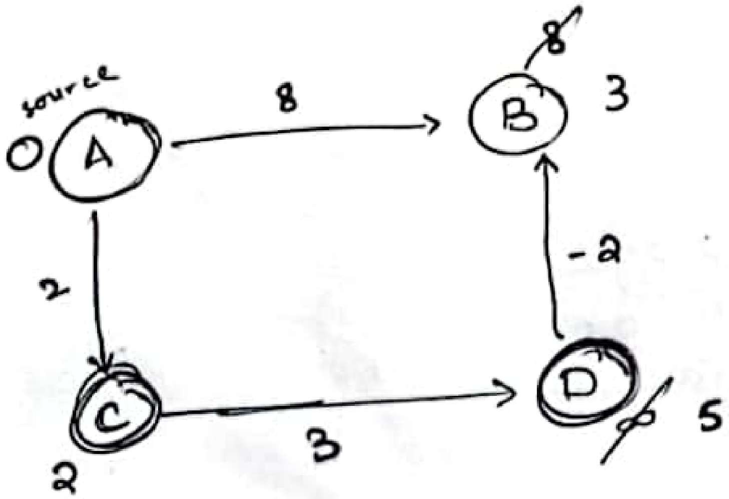


~~B) Part:~~

Q3

Part B:

- Dijkastra may or may not give correct ans for negative weight graphs.
- Example where dijkastra gives correct ans:



- Example when dijkastra gives wrong ans: graph in Question 3

Question 4) 10/10

```
pseudocode function longest commsubstring (A,B)
    n - length (A)
    m = length (B)
  dp = array of size (n+1) * (m+1) initialize to 0
    max length - 0
    end index = 0
for i from 1 to n:-
   for j from 1 to m:-
      if A [i-1] == B[j-1]
        dp [i][j] = dp [i-1][j-1] + 1
     if  dp [i][j] > maxlength.
        maxlength = dp [i][j]
          end index = i
  else:-
        dp [i][j] = 0
```

```
longest substring = A [endIndex - maxlength . endindex]
return (longest substring , maxlength)
```

Dryrun:-

i/j		b	c	d	e	g	g	g	7	e	f	g	g
		0	0	0	0	0	0	0					
a	0				0	0	0	0					
b	0				0	0	0	0					
c	0				0	0	0	0					
d	0	0	0	0	0	0	0	0	1	0	0	0	0
e	0	0	0	0	1	0	0	0	0	2	0	0	0
7	0	0	0	0	0	3	0	0	1	0	3	0	0
g	0	0	0	0	0	0	1	2	0	0	0	1	0
h	0	0	0	0	0	0	0	0	0	0	0	9	

Dryrun:-

i/j		b	c	d	e	g	g	f	e	f	g	g
	0	0	0	0	0	0	0	0	0	0	0	0
a	0	0	0	0	0	0	0	0	0	0	0	0
b	0	1	0	0	0	0	0	0	0	0	0	0
c	0	0	2	0	0	0	0	0	0	0	0	0
d	0	0	0	3	0	6	0	0	0	0	0	0
e	0	0	0	0	4	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0
e	0	0	0	0	1	0	0	6	2	0	0	0
7	0	0	0	0	0	0	0	1	6	3	0	0
g	0	0	0	0	0	1	1	0	0	0	4	
h	0	0	0	0	0	0	0	0	0	0		0

Finally BingO!

max length = 4 (substring "bcde")

endIndex = 5 in A

longestcommstring = A [5-4 : 5]

= "bcde"

Q5:) 10/10

```
function countingSums(int n)
{
        int dp[n] = {0}
        dp[0] = 1
        for i 1 ⟶ n
        {
                for j i ⟶ n
                { dp[j] += dp[j-i]
                }
        }
        return dp[n] - 1
}
```

for n = 50 this function returns: 204225

for n = 10 dp array would change as follows:

Iterations	Array										
1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	2	2	3	3	4	11	5	5	6
3	1	1	2	3	4	5	7	8	10	12	14
4	1	1	2	3	5	6	9	11	15	18	23
5	1	1	2	3	5	7	10	13	18	23	30
6	1	1	2	3	5	7	11	14	20	26	35
7	1	1	2	3	5	7	11	15	21	28	38
8	1	1	2	3	5	7	11	15	22	29	40
9	1	1	2	3	5	7	11	15	22	30	41
10	1	1	2	3	5	7	11	15	22	30	42

result = 41