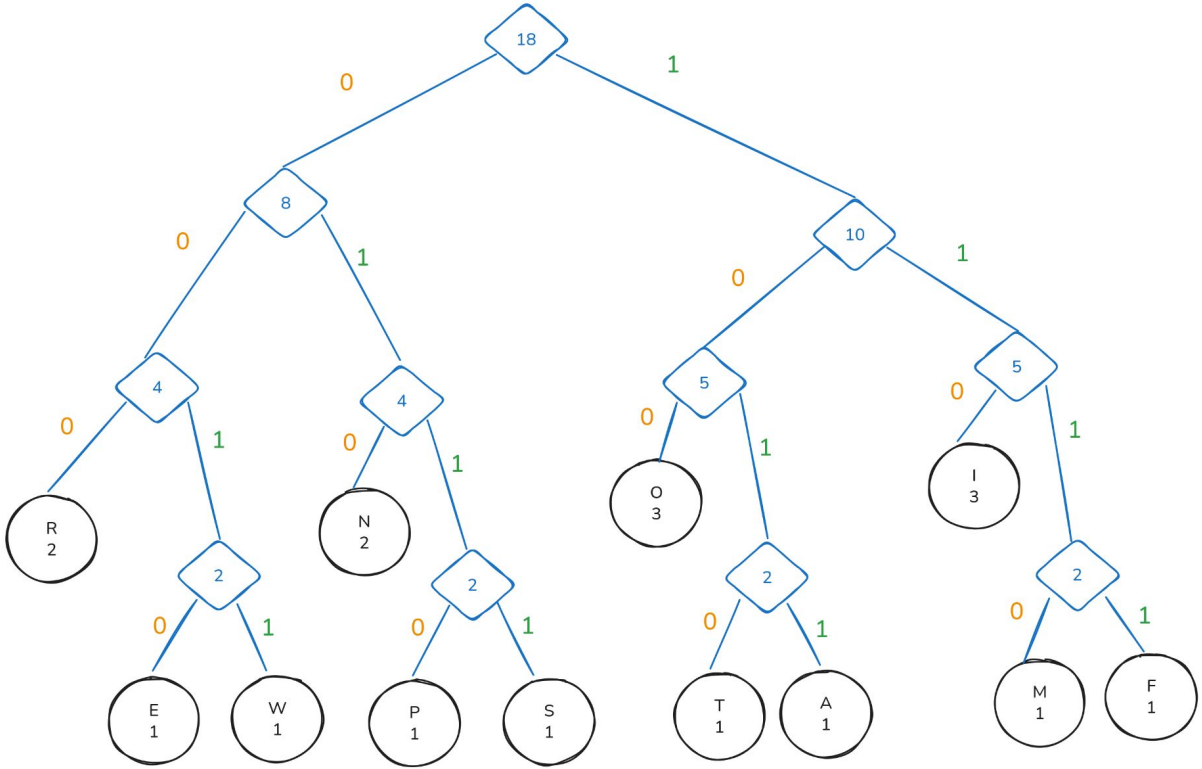
Assignment 1.1

Given string:

INFORMATIONISPOWER

Charcount:

I - 3
O - 3
N - 2
R - 2
F - 1
M - 1
A - 1
T - 1
S - 1
P - 1
W - 1
E - 1



I	110
O	100
N	010
R	000
F	1111
M	1110
A	1011
T	1010
S	0111
P	0110
W	0011
E	0010

Assignment 1.2

Encode the original string “INFORMATIONISPOWER”. Replace each character in the original string "INFORMATIONISPOWER" with its Huffman code. Write out the final bit string representing the compressed original string.

I	N	F	O	R	M	A	T	I	O	N	I	S	P	O	W	E	R
110	*010*	*1111*	*100*	*000*	*1110*	*1011*	*1010*	*110*	*100*	*010*	*110*	*0111*	*0110*	*100*	*0011*	*0010*	*000*

Assignment 1.3

Write the string “INFORMATIONISPOWER” in the form of 8-bit ASCII code.

Char	I	N	F	O	R	M	A	T	I	O	N	I	S	P	O	W	E	R
ASCII	73	78	70	79	82	77	65	84	73	79	78	73	83	80	79	87	69	82
Bin	01001001	01001110	01000110	01001111	01010010	01001101	01000001	01010100	01001001	01001111	01001110	01001001	01010011	01010000	01001111	01010111	01000101	01010010

Resulting ASCII binary string:
010010010100111001000110010011111010100100100110101000001010101000100100101001111010
011100100100101010011010100000100111101010111010001010101010010

Compression ratio:

$$\frac{C(M)\ bits}{M\ bits} = \frac{(3 * 10 + 4*8)}{(8* 18)} = 0.43$$

Assignment 2.1

TOBEORNOTTOBEORTOBEORNOT

Resulting dictionary:

Character	ASCII
B	66
E	69
N	78
O	79
R	82
T	84
…	..
TO	256
OB	257
BE	258
EO	259
OR	260
RN	261
OT	263
TOB	265
BEO	266
ORT	267

Last word	Current word	New entry	Dict-Code
…	T	-	
T	O	TO = 256	84
O	B	OB = 257	79
B	E	BE = 258	66
E	O	EO = 259	69
O	R	OR = 260	79
R	N	RN = 261	82
N	O	NO = 262	78
O	T	OT = 263	79
T	T	TT = 264	84
T	O	-	-
TO	B	TOB = 265	256
B	E	-	-
BE	O	BEO = 266	258
O	R	-	-
OR	T	ORT = 267	260
T	O	-	-
TO	B	-	-
TOB	E	TOBE = 268	265
E	O	-	-
EO	R	EOR = 269	259
R	N	-	-
RN	O	RNO = 270	261
O	T	-	-
OT	NULL	-	263

Result:

Original string

T	O	B	E	O	R	N	O	T	T	O	B	E	O	R	T	O	B	E	O	R	N	O	T
84	79	66	69	79	82	78	79	84	84	79	66	69	79	82	84	79	66	69	79	82	78	79	84

Encoded

T	O	B	E	O	R	N	O	T	TO	BE	OR	TOB	EO	RN	OT
84	79	66	69	79	82	78	79	84	256	258	260	265	259	261	263

Compression ratio: $\frac{C(M)\ bits}{M\ bits} = \frac{16 * 12}{24 * 12} = 0.67$

Assignment 2.2 Decoding

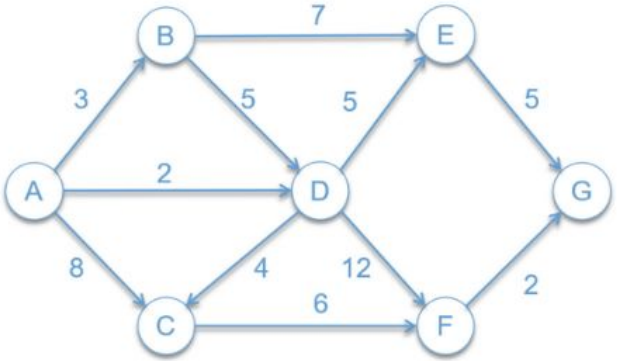
Given sequence: 83 69 81 85 69 78 67 69 256 258 260 265

Current code	Output	New entry
83	S	-
69	E	SE = 256
81	Q	EQ = 257
85	U	QU = 258
69	E	UE = 259
78	N	EN = 260
67	C	NC = 261
69	E	CE = 262
256	SE	ES = 263
258	QU	SEQ = 264
260	EN	QUE = 265
265	QUE	ENQ = 266

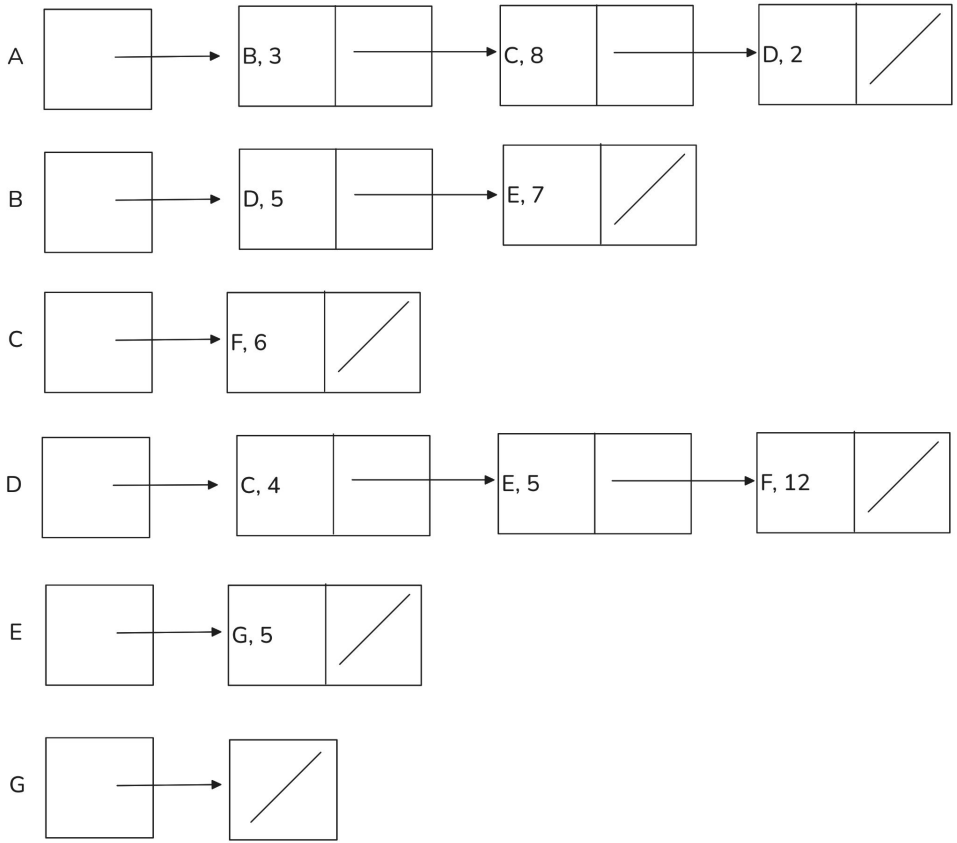
Dictionary	
Character	ASCII
67	C
69	E
78	N
81	Q
83	S
85	U
…	…
256	SE
257	EQ
258	QU
259	UE
260	EN
261	NC
262	CE
263	ES
264	SEQ
265	QUE
266	ENQ

Output: S E Q U E N C E SE QU EN QUE

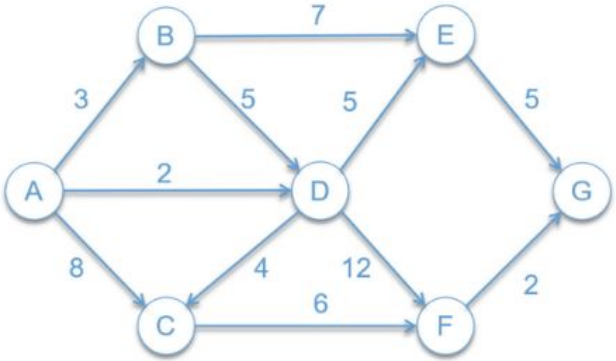
Assignment 3.1 Representation - Adjacency list



Adjacency list



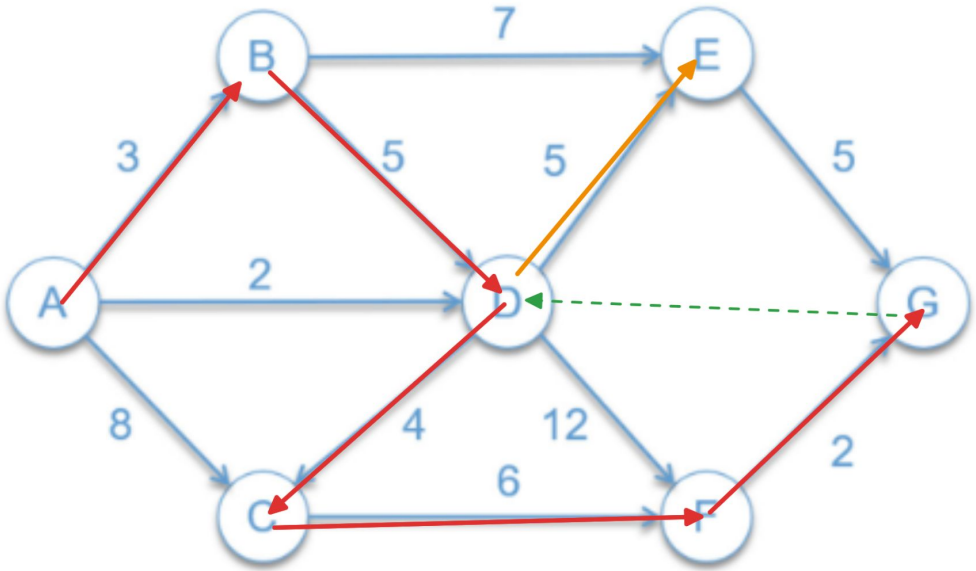
Assignment 3.1 Representation - Adjacency matrix



	A	B	C	D	E	F	G
A	0	3	8	2	0	0	0
B	0	0	0	5	7	0	0
C	0	0	0	0	0	6	0
D	0	0	4	0	5	12	0
E	0	0	0	0	0	0	5
F	0	0	0	0	0	0	2
G	0	0	0	0	0	0	0

Assignment 3.2 dfs

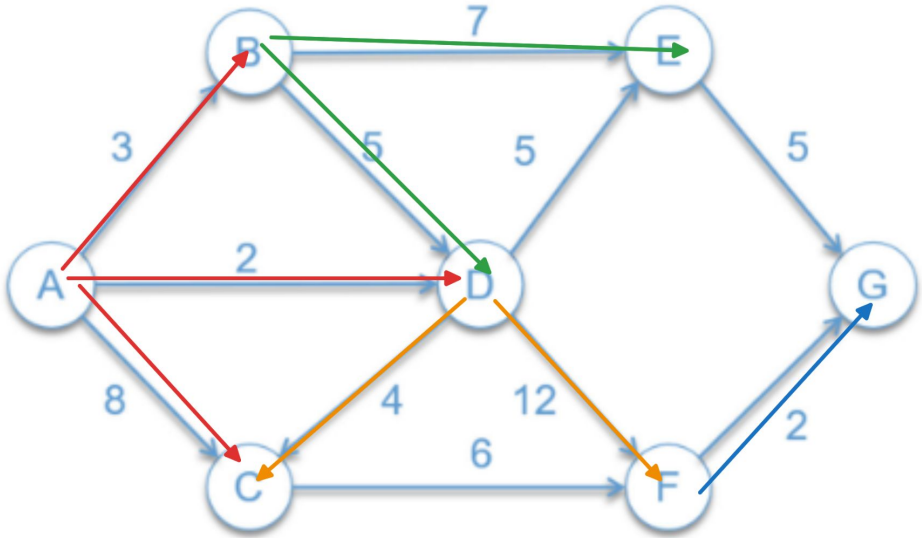
DFS



A - B - D - C - F - G - E

Assignment 3.2 bfs

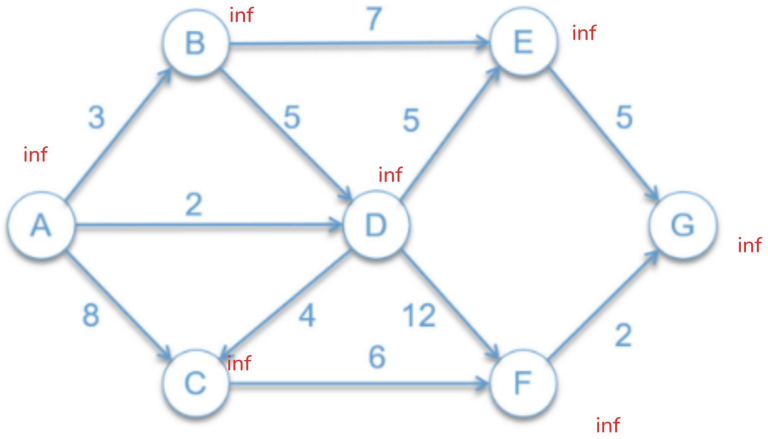
BFS



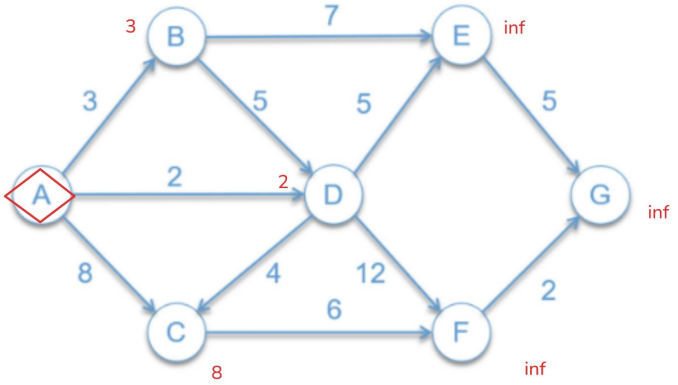
A B D C E D F G

Assignment 3.3 Shortest path (Dijkstra's algorithm)

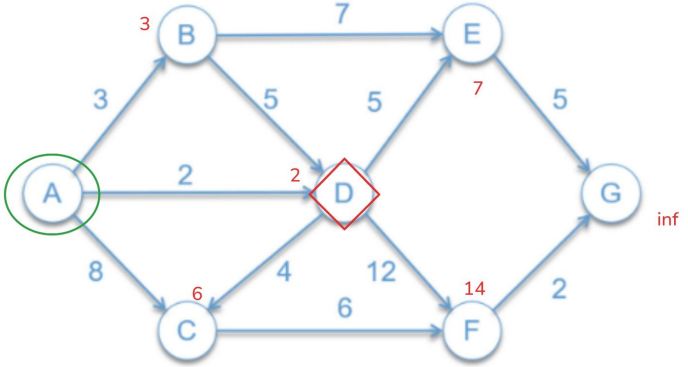
0 Unvisited {A, B, D, C, E, F, G}



1 Unvisited {A, B, D, C, E, F, G}



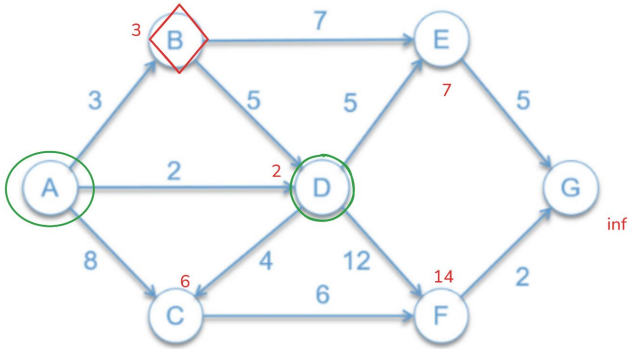
2 Unvisited {B, D, C, E, F, G}



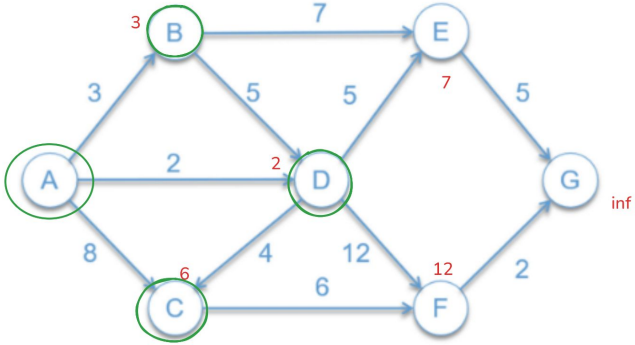
Assignment 3.3 Shortest path (Dijkstra's algorithm)

3 Unvisited {B, C, E, F, G}

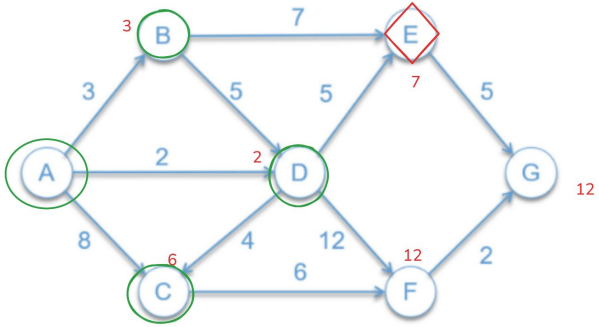
Nothing to update, proceed to next



4 Unvisited {E, F, G}

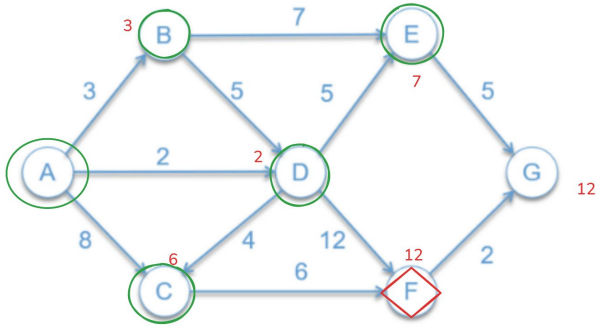


5 Unvisited {E, F, G}



6 Unvisited {F, G}

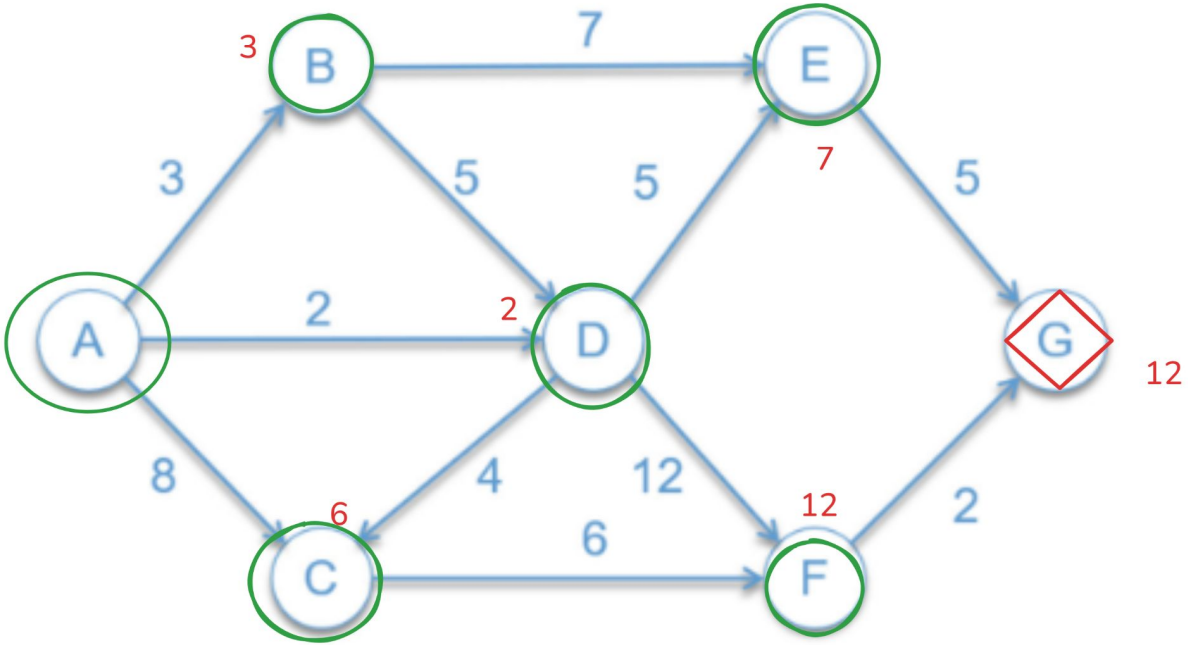
Distance from F to G = 14.
Nothing to update.



Assignment 3.3 Shortest path (Dijkstra's algorithm) **RESULT**

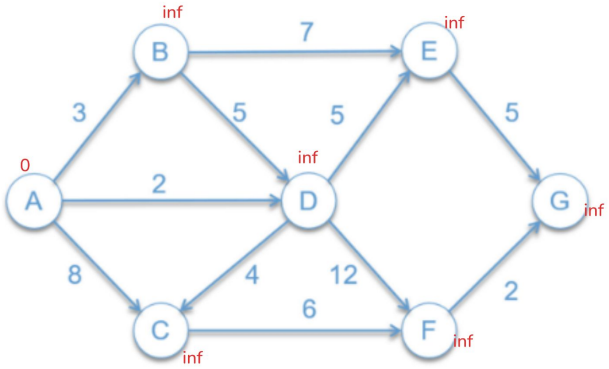
7 Unvisited {G}

Stop at G, shortest path = 12

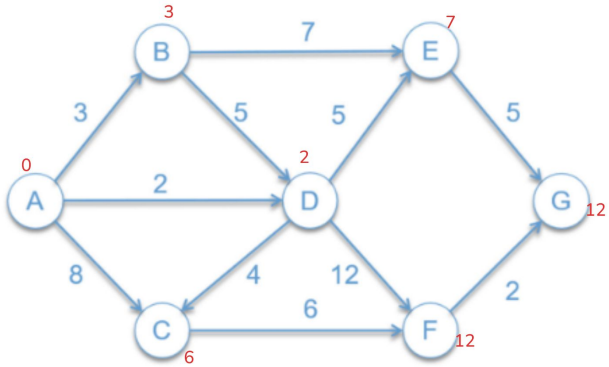


Assignment 3.3 Shortest path (Bellman-Ford algorithm)

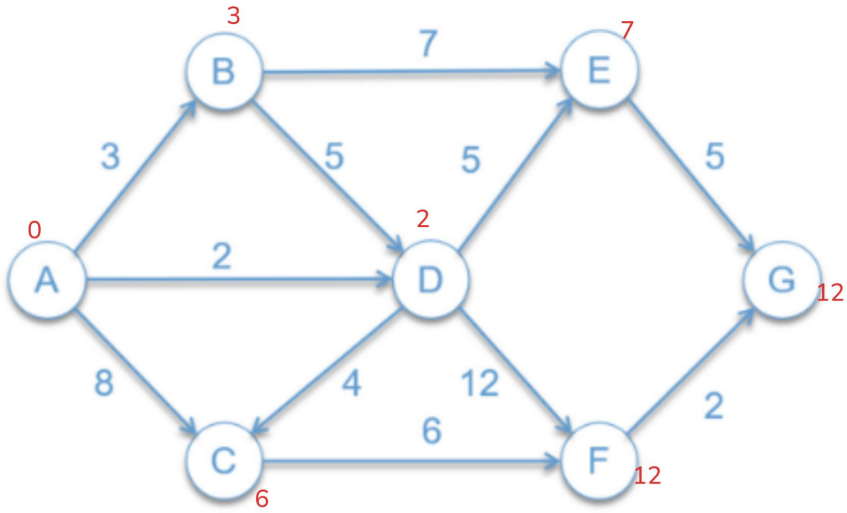
1



2



3

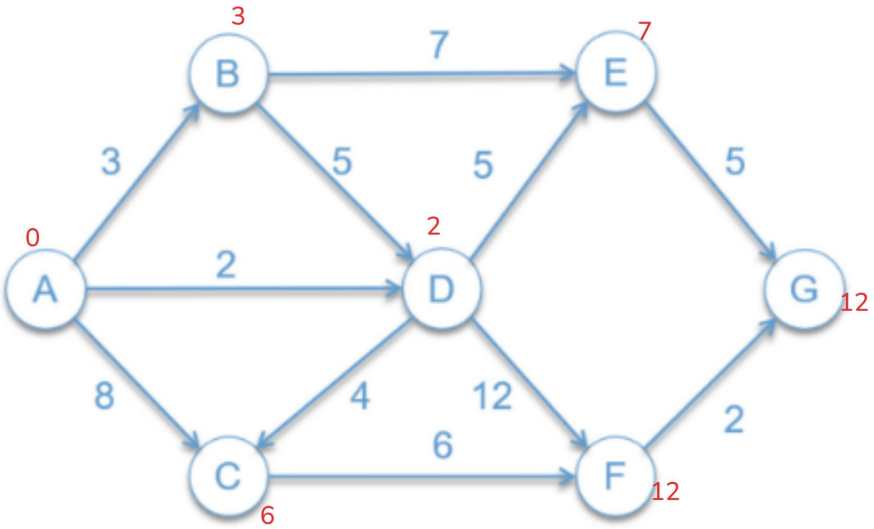


Traversal order:
AB - AD - AC - BE - BD - DE - DF - DC - CF - EG - FG

Iterations: 7 - 1 = 6

Assignment 3.3 Shortest path (Bellman-Ford algorithm)

3



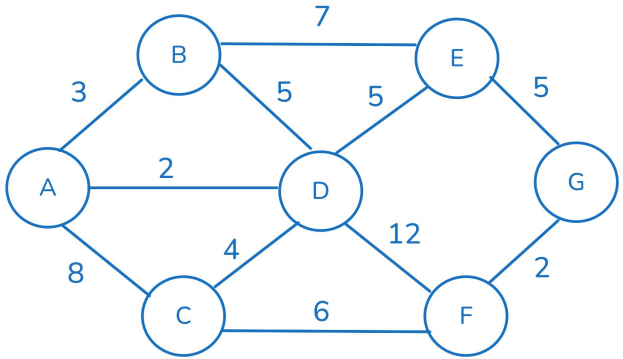
Traversal order:
AB - AD - AC - BE - BD - DE - DF - DC - CF - EG - FG

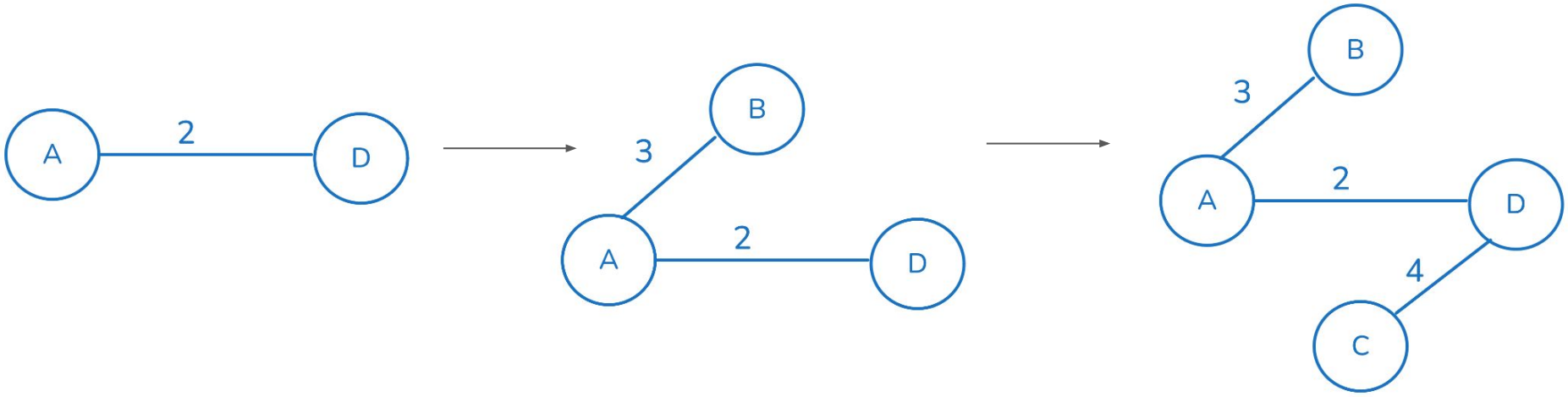
Iterations: 7 - 1 = 6

Resulting table

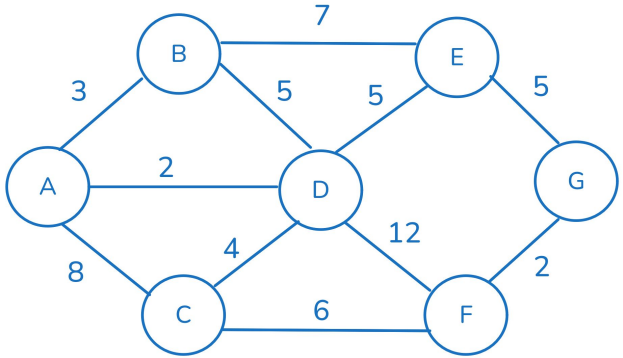
i	A	B	C	D	E	F	G
1	0	inf	inf	inf	inf	inf	inf
2	0	3	6	2	7	12	12
3	0	3	6	2	7	12	12
4	Algorithm stopped						

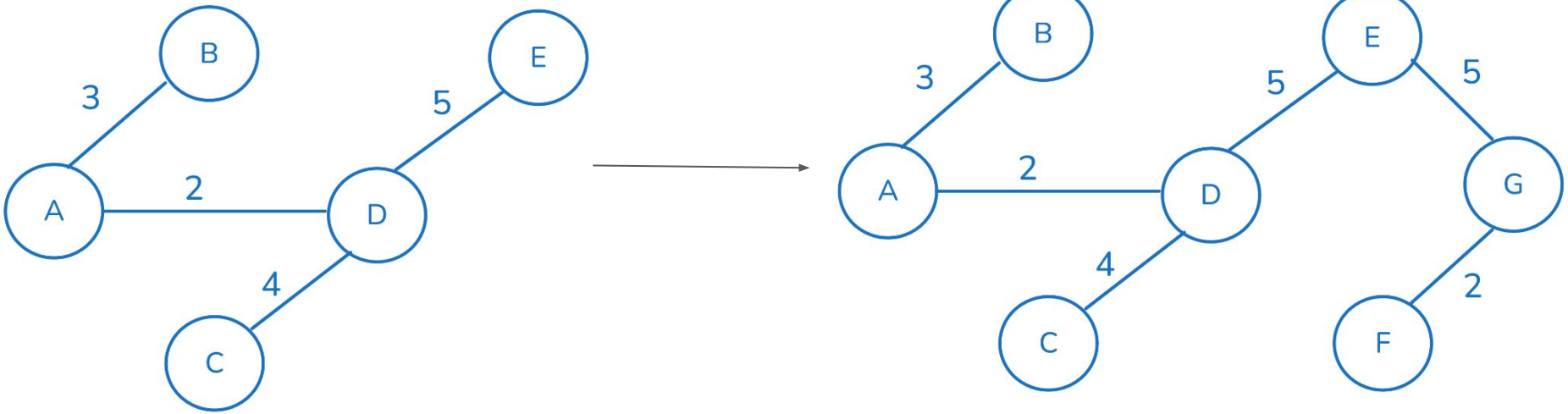
Assignment 3.4 Minimum spanning tree (Kruskal's algorithm)



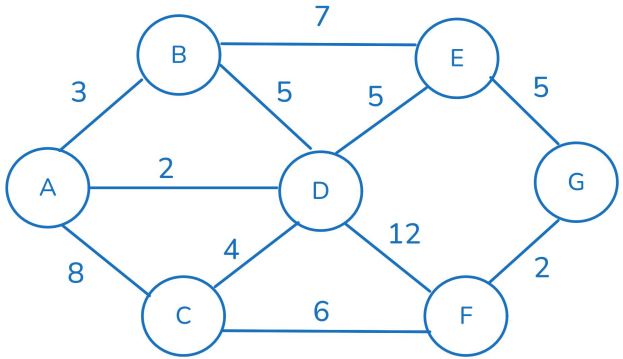


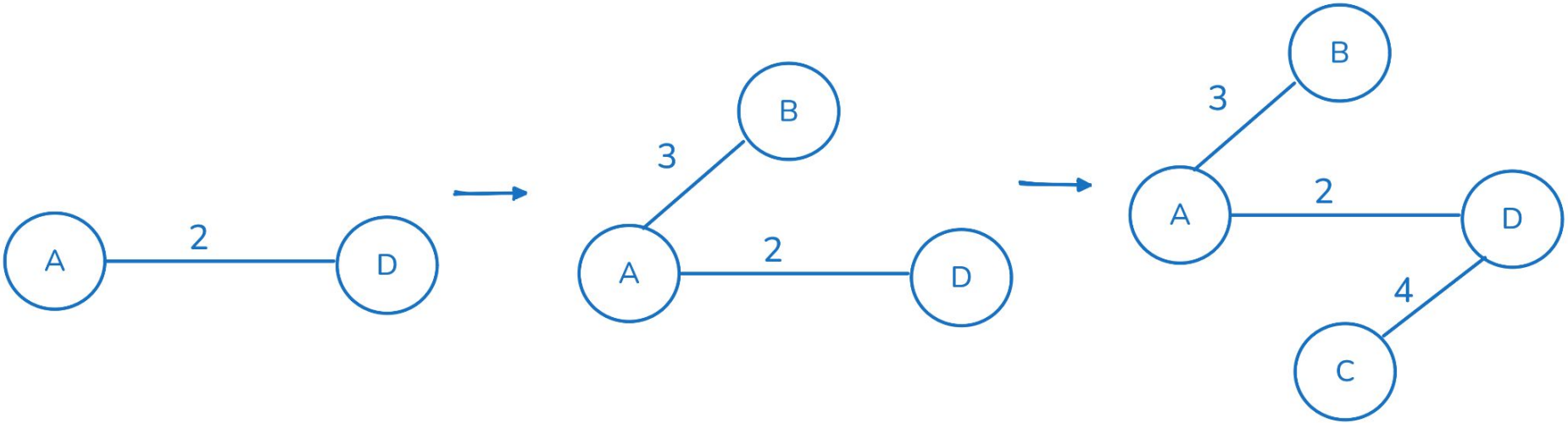
Assignment 3.4 Minimum spanning tree (Kruskal's algorithm)



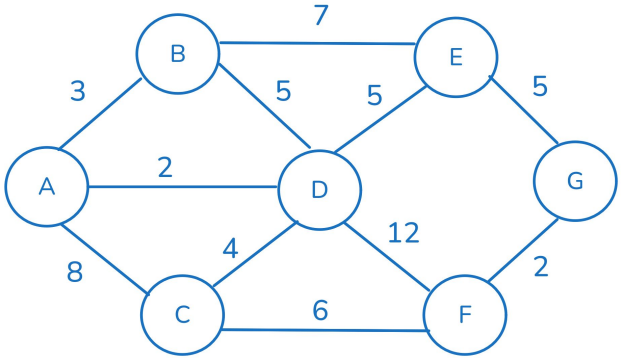


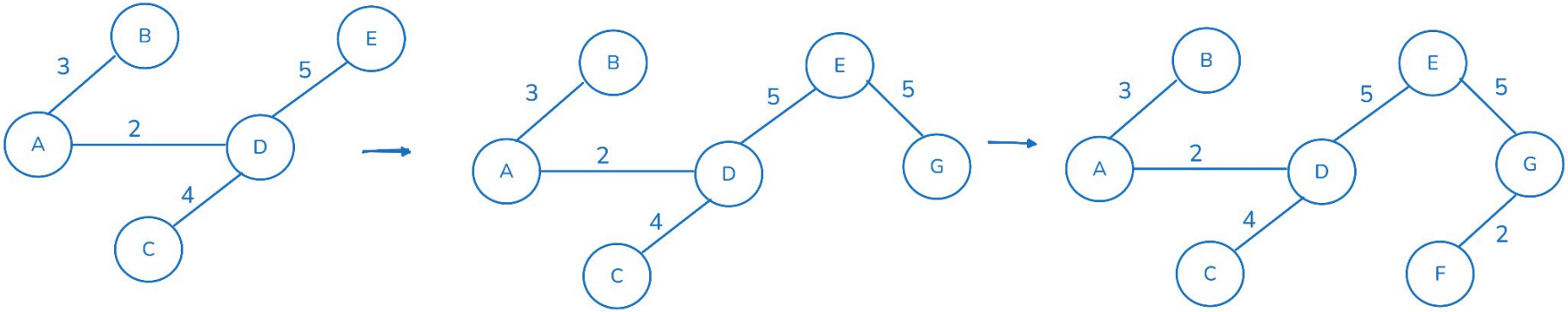
Assignment 3.4 Minimum spanning tree (Prim's algorithm)



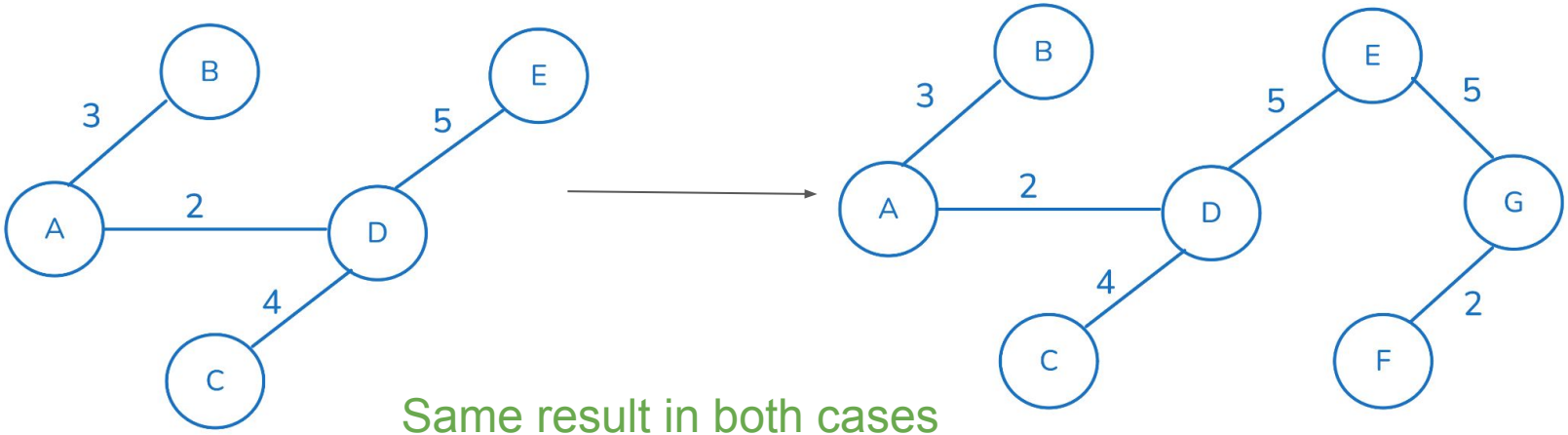


Assignment 3.4 Minimum spanning tree (Prim's algorithm)





Assignment 3.4 Minimum spanning tree (Comparing the results)



Same result in both cases