Fundamentals of Artificial Intelligence

Laboratory

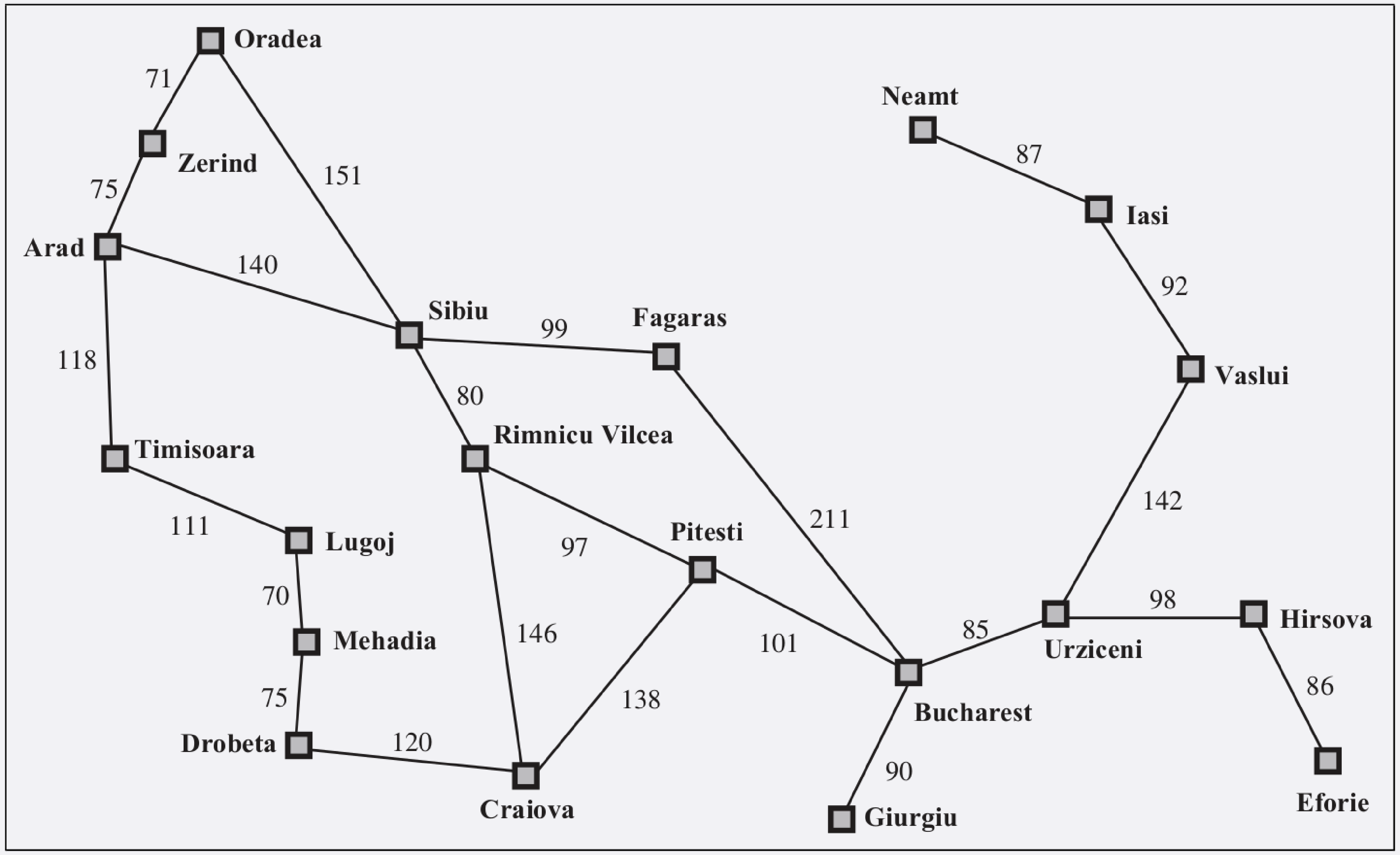
Dr. Mauro Dragoni

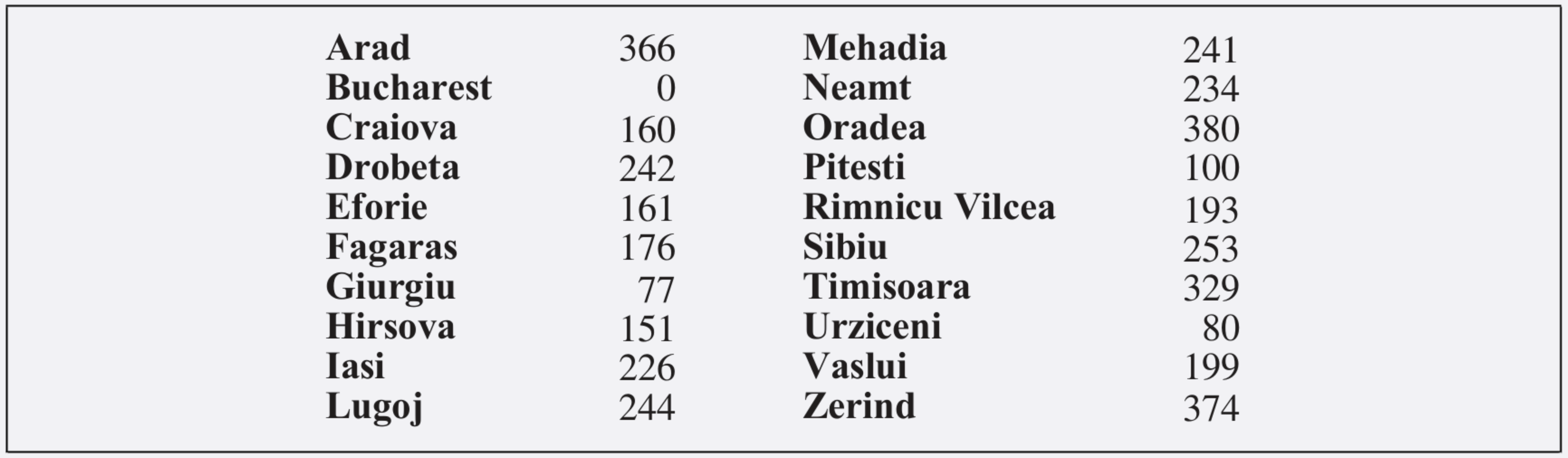
Department of Information Engineering and Computer Science

Academic Year 2022/2023

Exercise 3.11

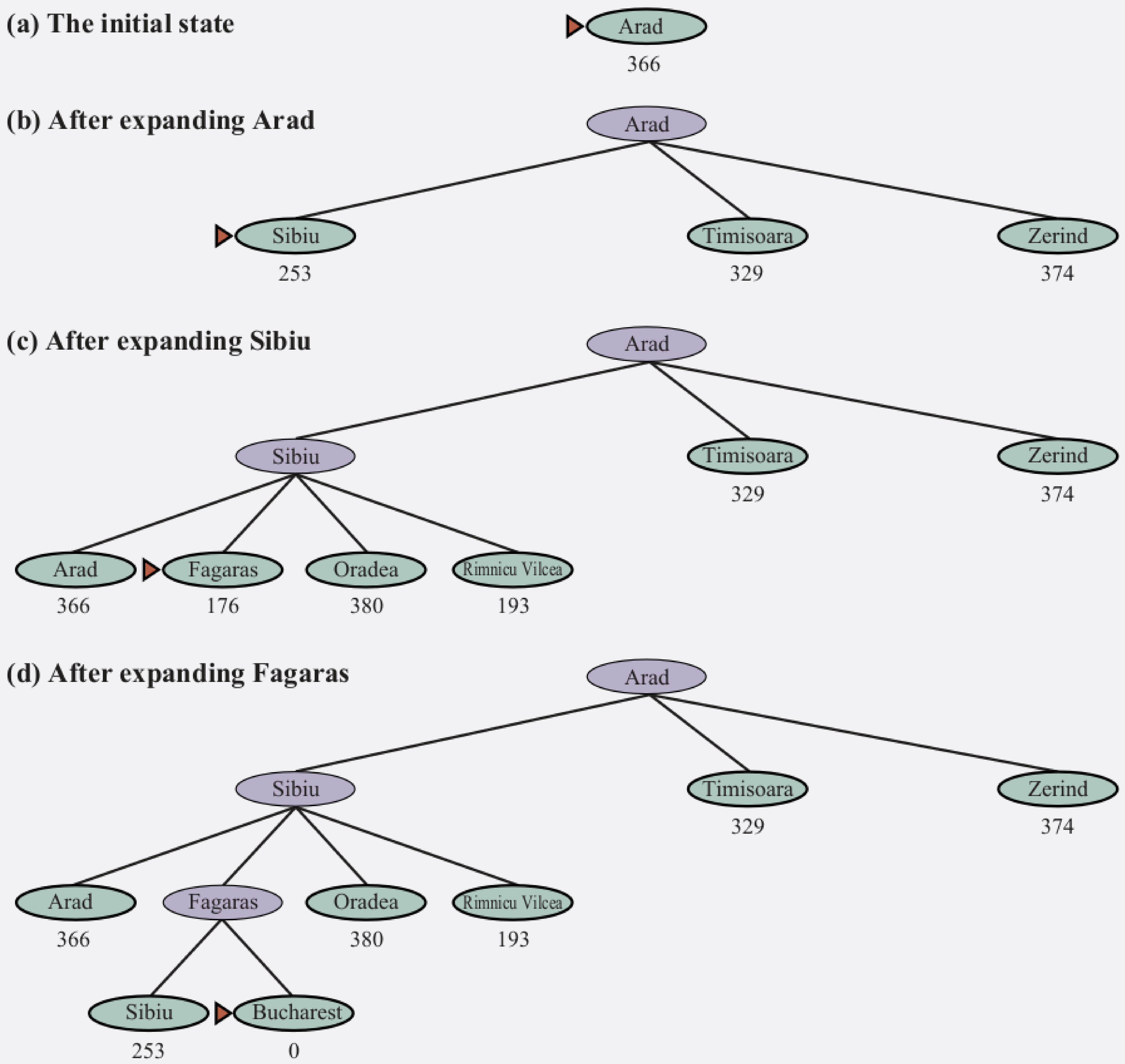
- Apply the **greedy best-first search** strategy for finding the route from Arad to Bucharest.



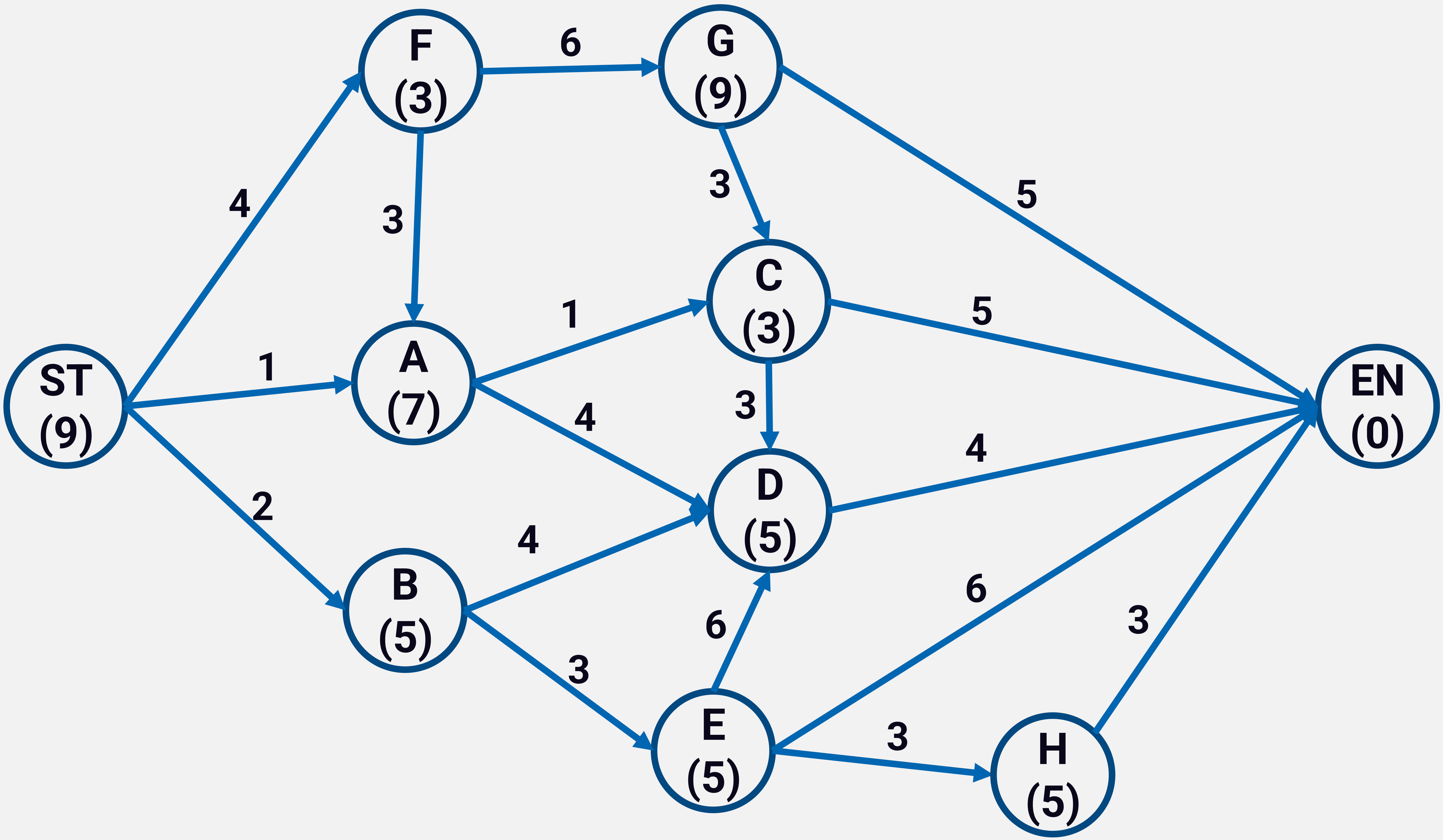


Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Hirsova	151	Urziceni	80
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374

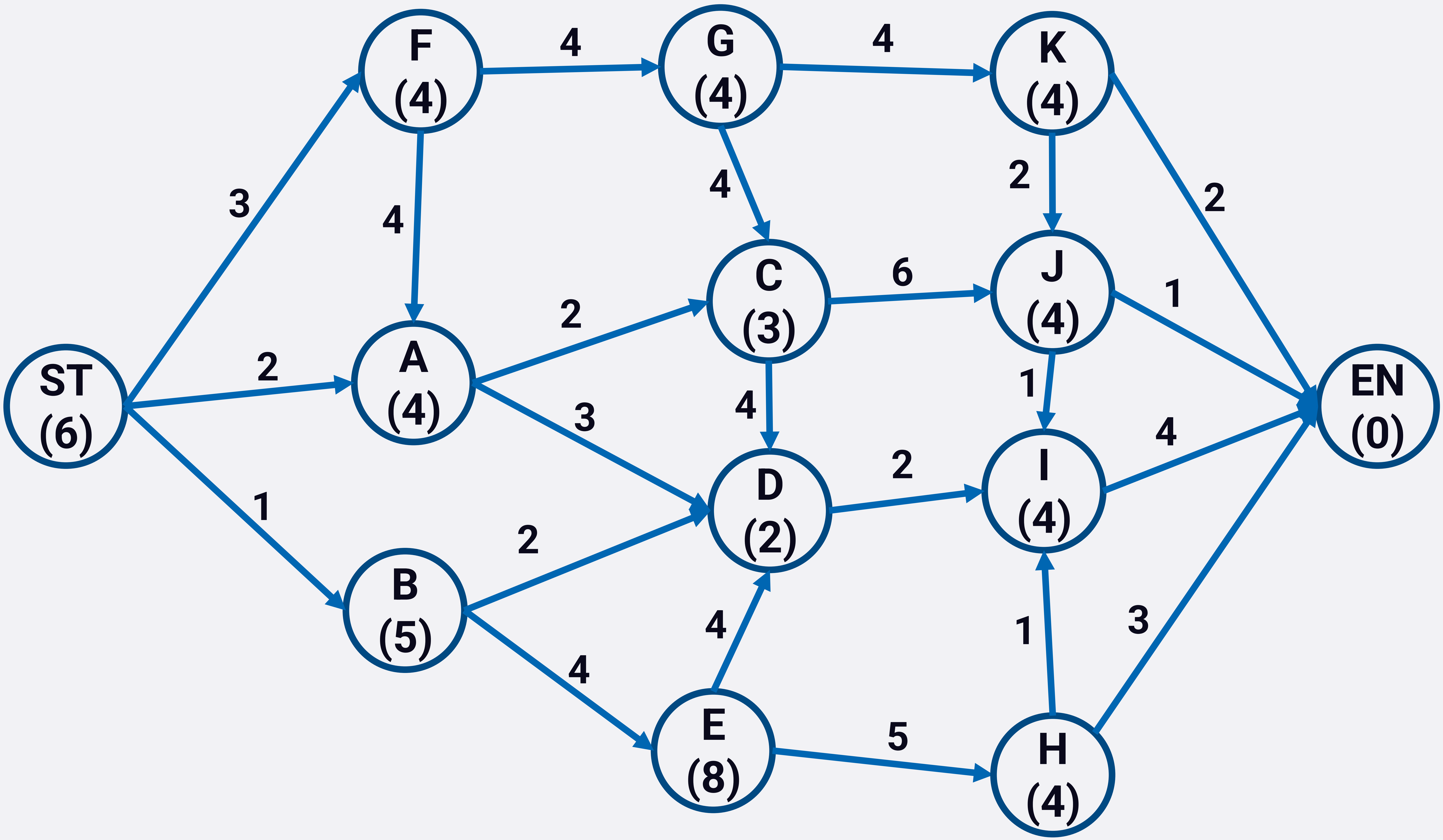
Exercise 3.11 - Solution



Exercise 3.18



Exercise 3.19



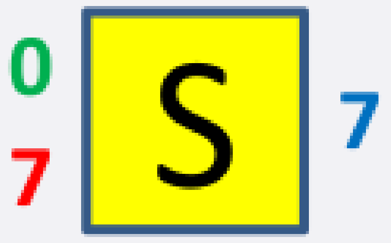
Exercise 3.12

- A* algorithm

```
------------------
WHILE (QUEUE not empty && first path not reach goal) DO
     Remove first path from QUEUE
     Create paths to all children
     Reject paths with loops
     Add paths and sort QUEUE (by f = cost + heuristic)
     IF QUEUE contains paths: P, Q
          AND P ends in node Ni && Q contains node Ni
          AND cost(P) ≥ cost(Q)
     THEN remove P

IF goal reached THEN success ELSE failure
------------------
```

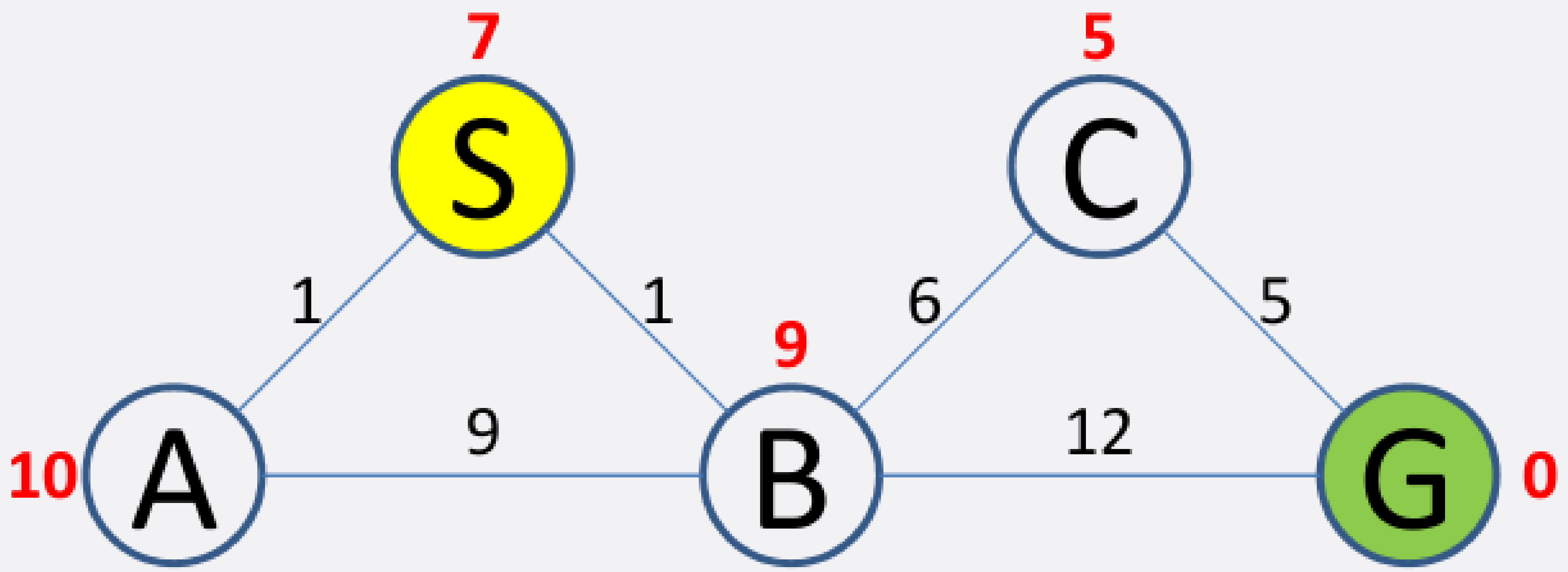
Exercise 3.12



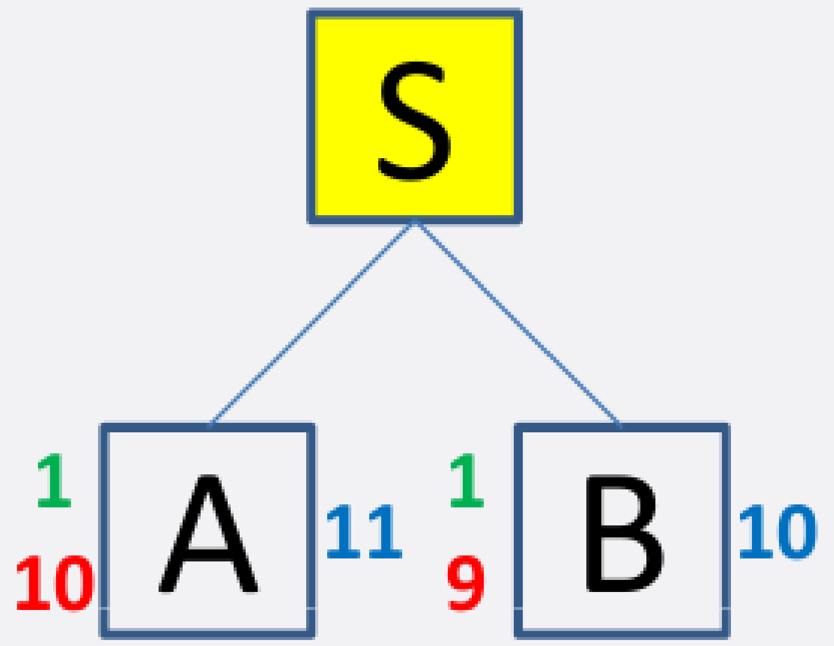
f = accumulated path cost + heuristic

QUEUE = path containing root

QUEUE = <S>



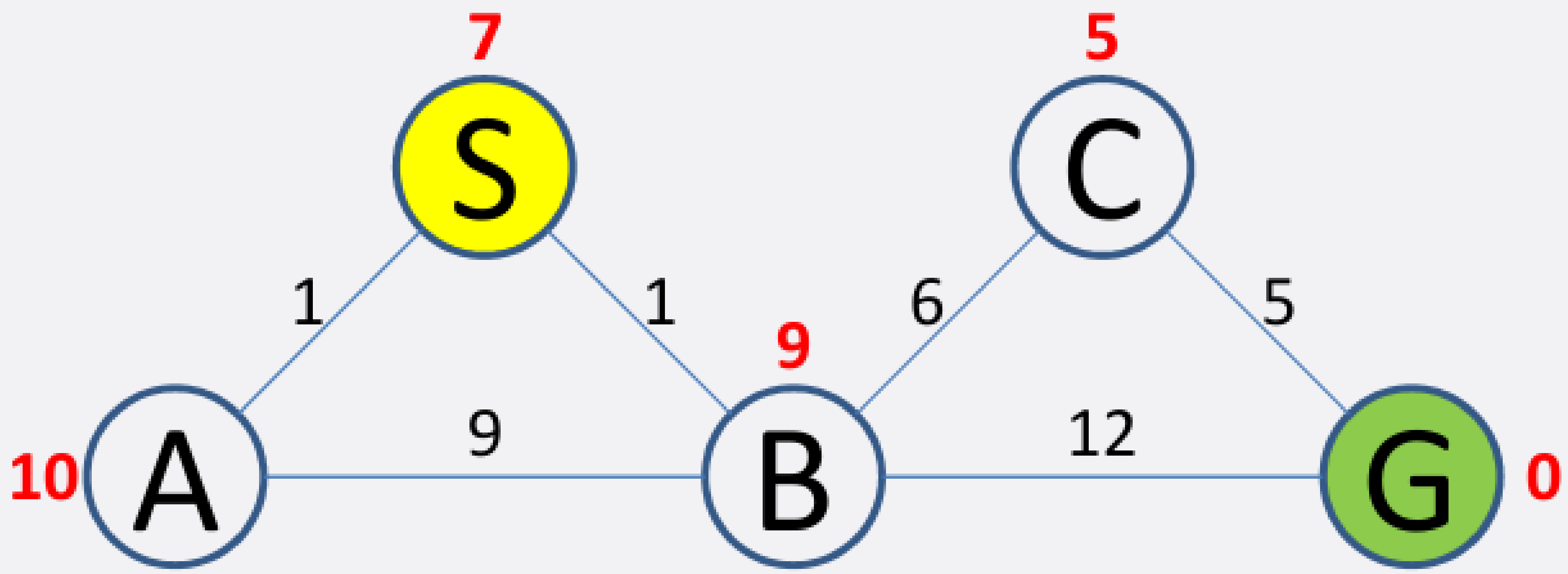
Exercise 3.12



f = accumulated path cost + heuristic

Remove first path, Create paths to all children, Reject loops and Add paths. SORT QUEUE by f

QUEUE = <SB,SA>

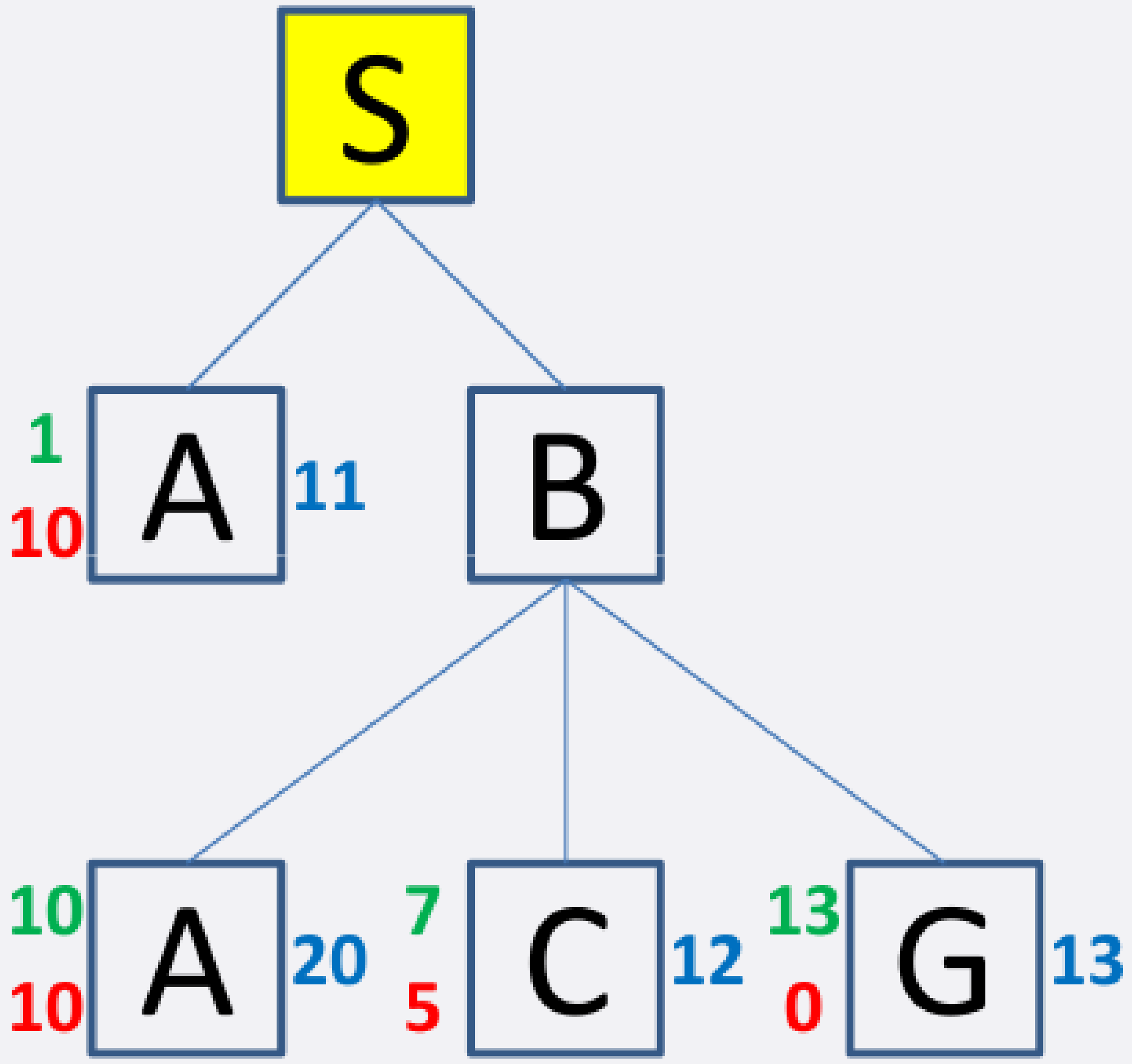


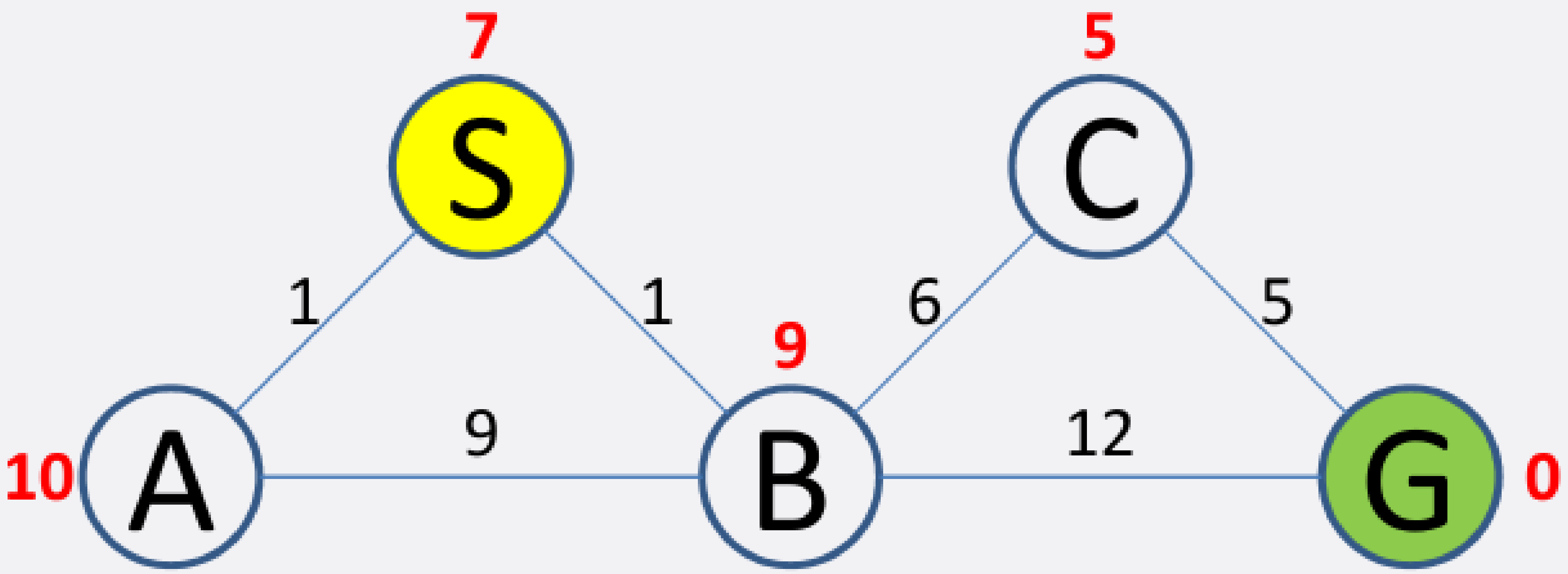
Exercise 3.12

f = accumulated path cost + heuristic

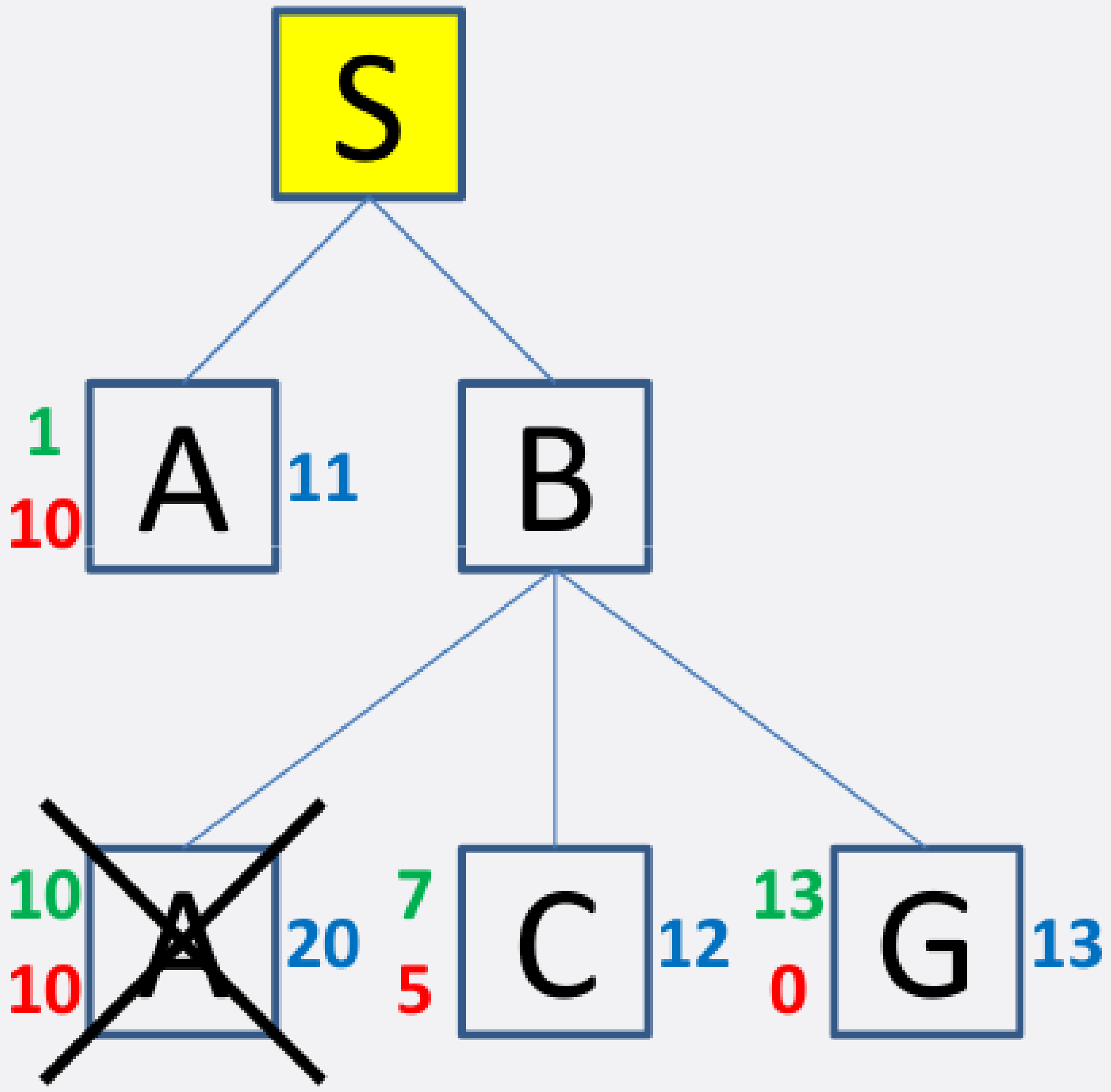
Remove first path, Create paths to all children, Reject loops and Add paths. SORT QUEUE by f

QUEUE = <SA,SBC,SBG,SBA>





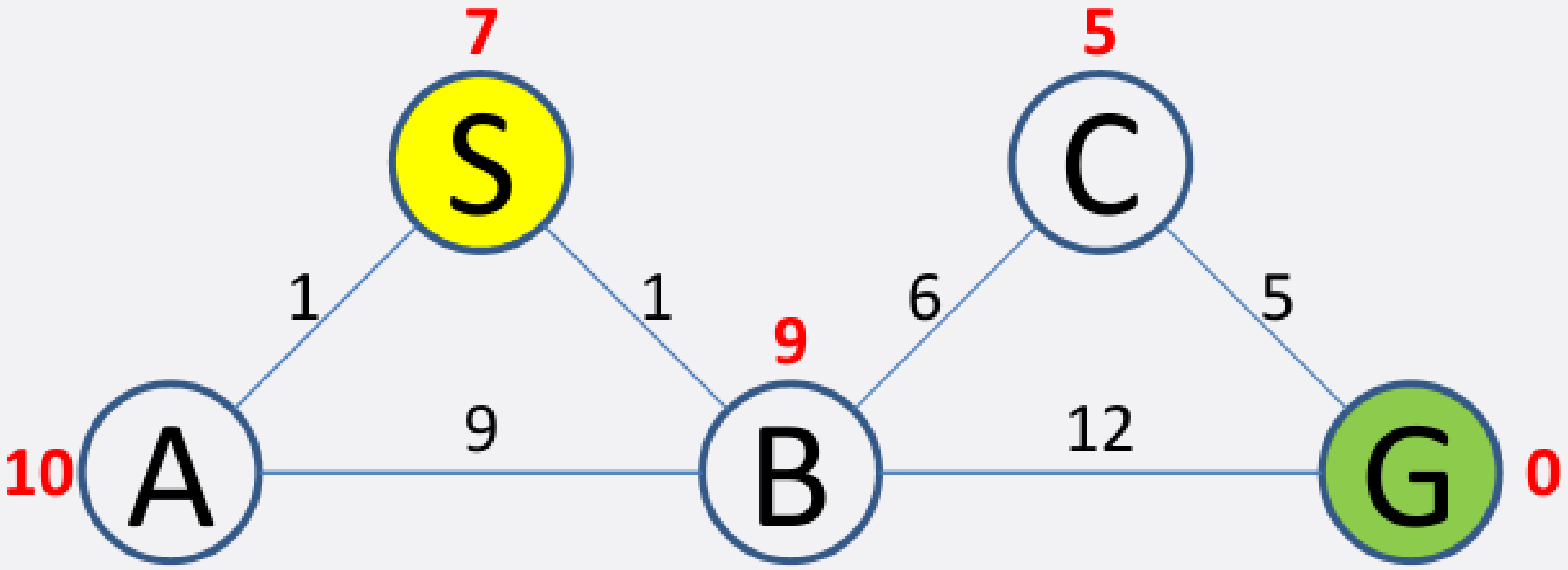
Exercise 3.12



f = accumulated path cost + heuristic

```
IF QUEUE contains paths: P, Q
   AND P ends in node Ni && Q contains node Ni
   AND cost(P) ≥ cost(Q)
THEN remove P
```

QUEUE = <SA,SBC,SBG,**SBA**>

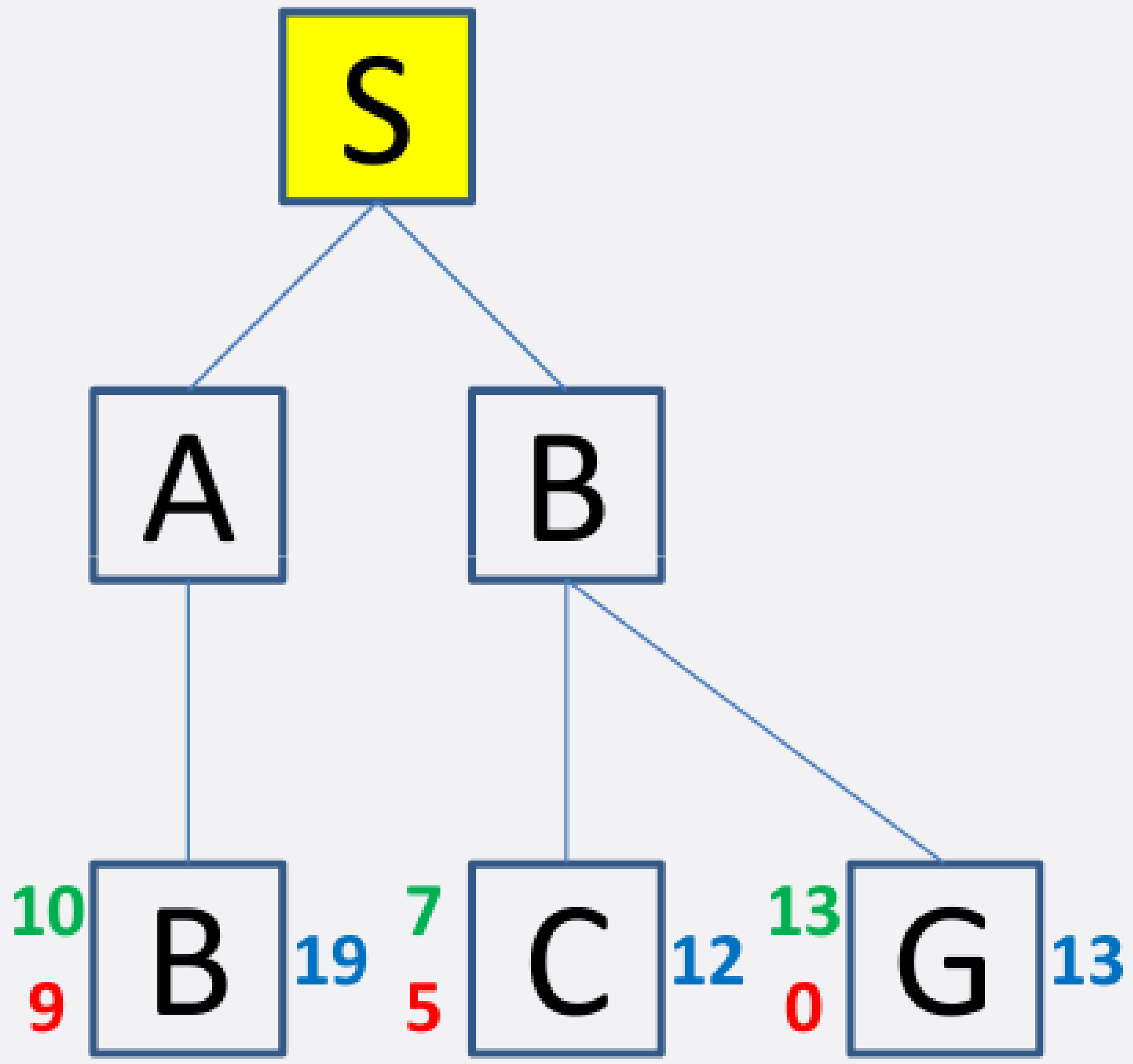


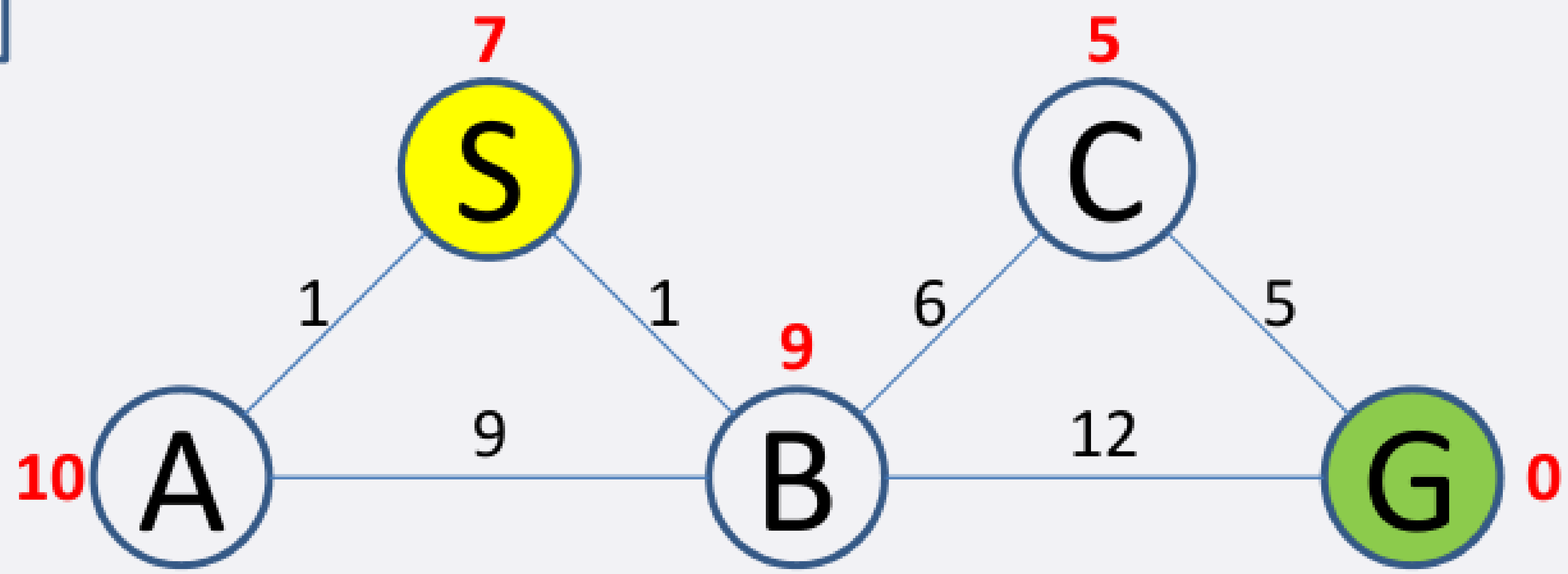
Exercise 3.12

f = accumulated path cost + heuristic

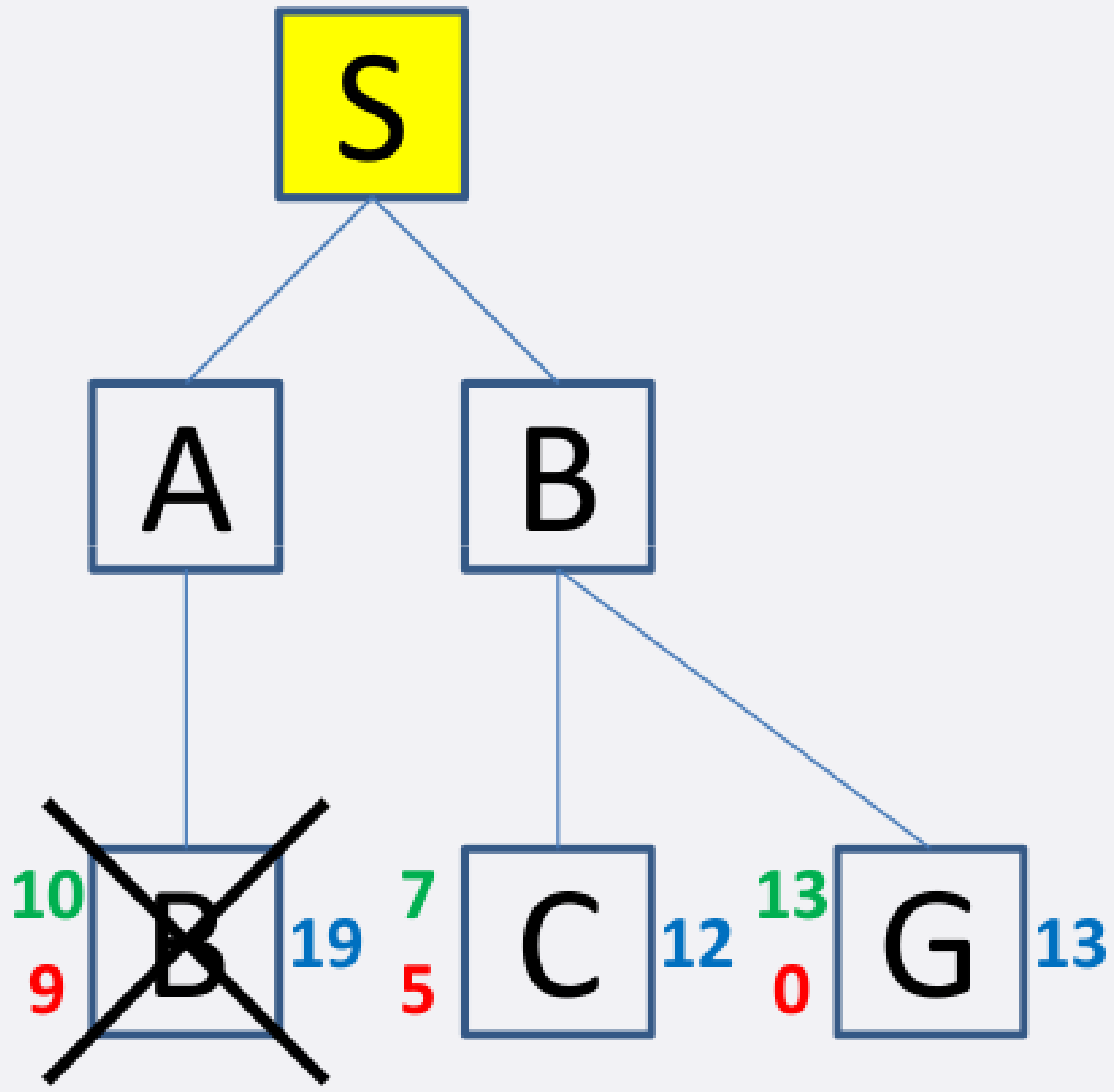
Remove first path, Create paths to all children, Reject loops and Add paths. SORT QUEUE by f

QUEUE = <SBC,SBG,SAB>





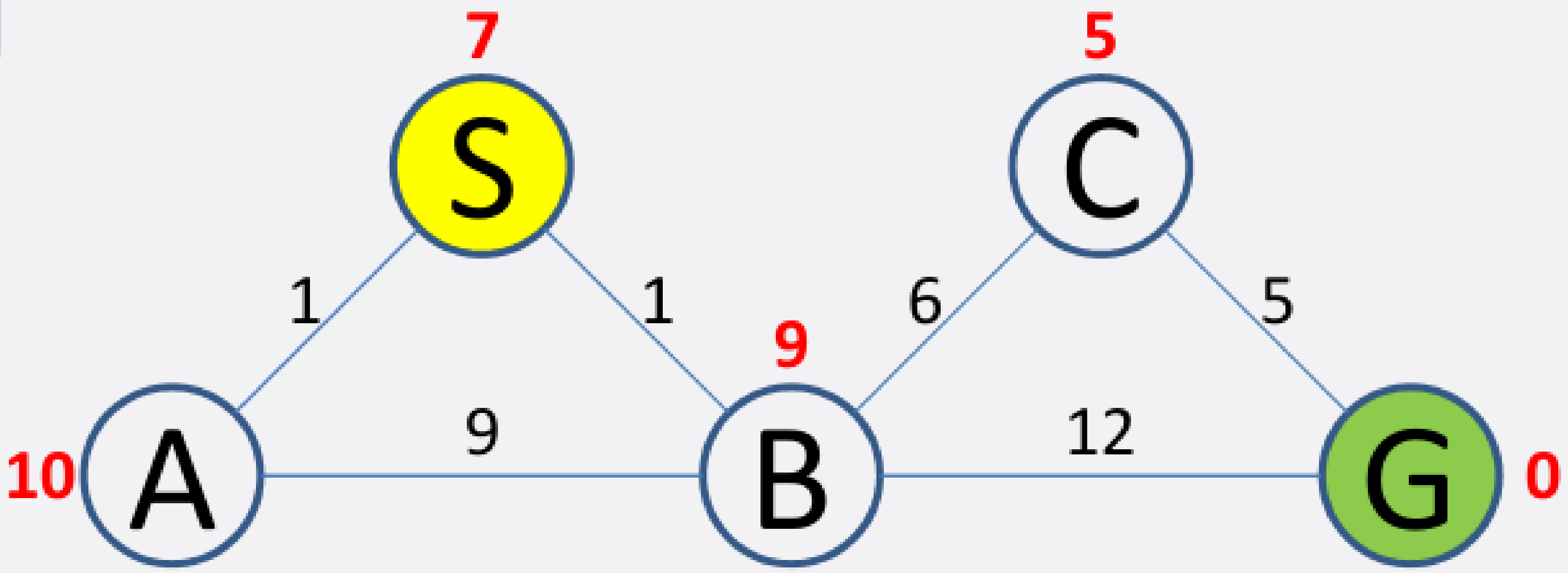
Exercise 3.12



f = accumulated path cost + heuristic

```
IF QUEUE contains paths: P, Q
   AND P ends in node Ni && Q contains node Ni
   AND cost(P) ≥ cost(Q)
THEN remove P
```

QUEUE = < SBC,SBG,**SAB**>

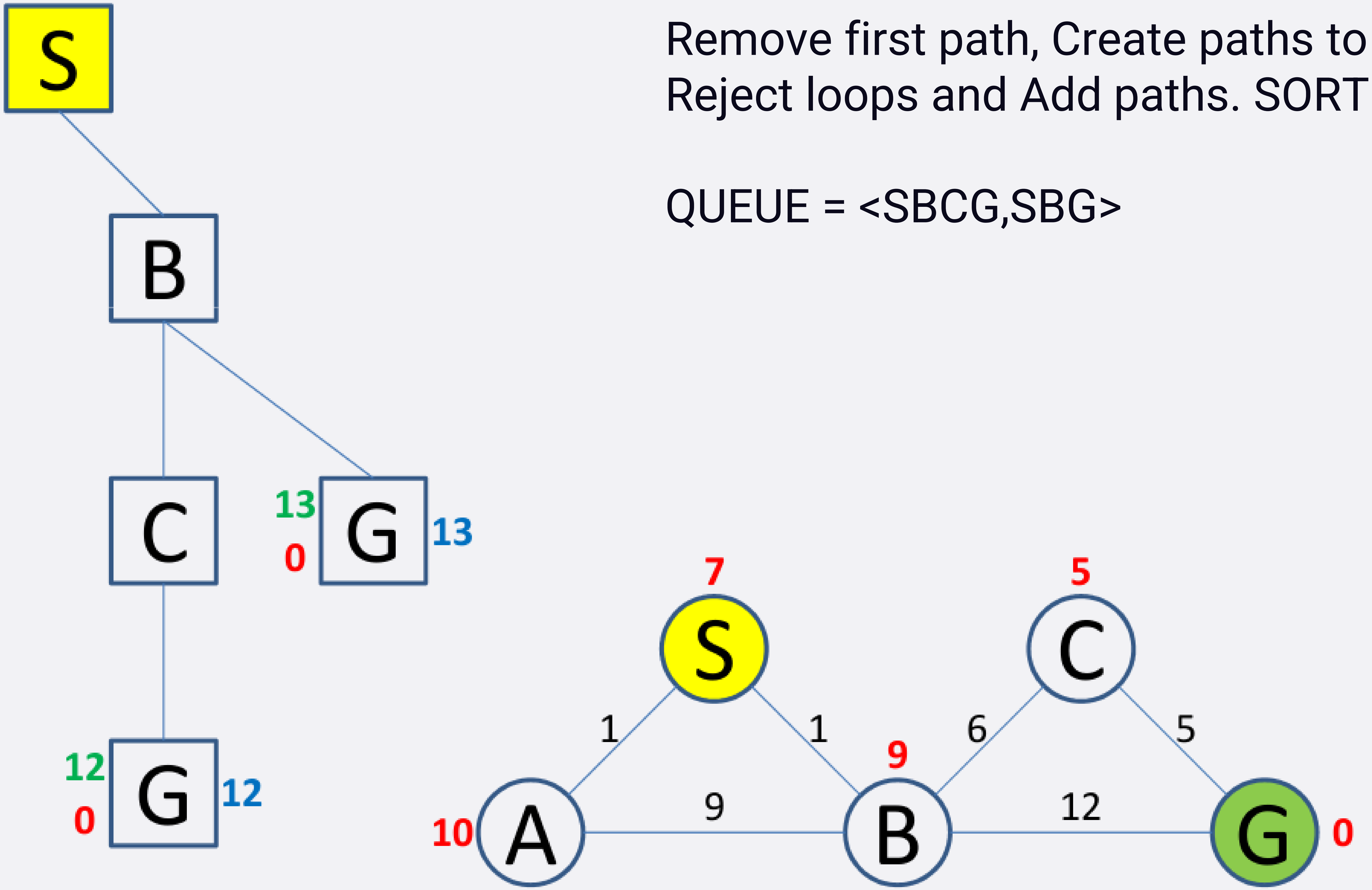


Exercise 3.12

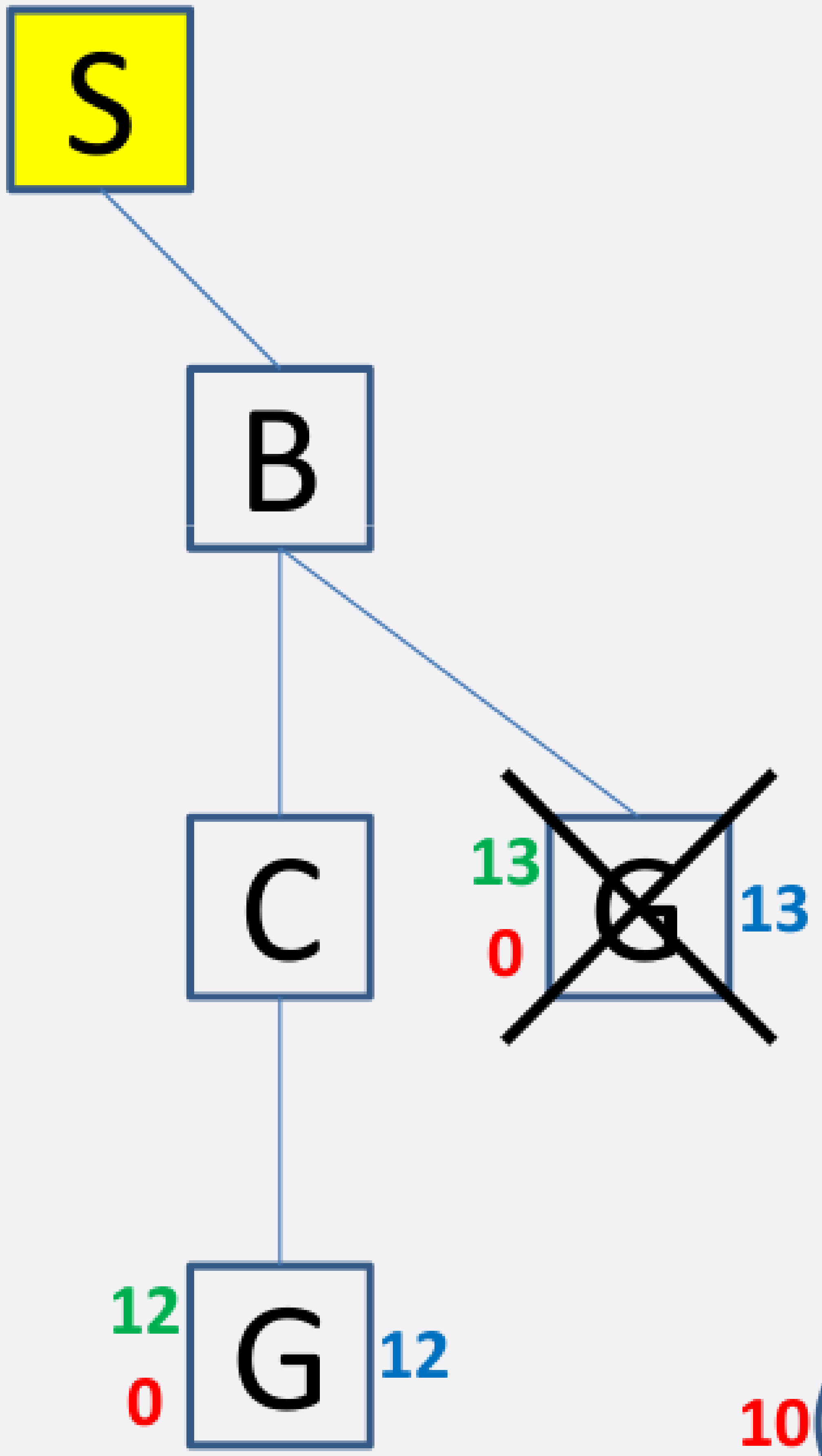
f = accumulated path cost + heuristic

Remove first path, Create paths to all children, Reject loops and Add paths. SORT QUEUE by f

QUEUE = <SBCG,SBG>



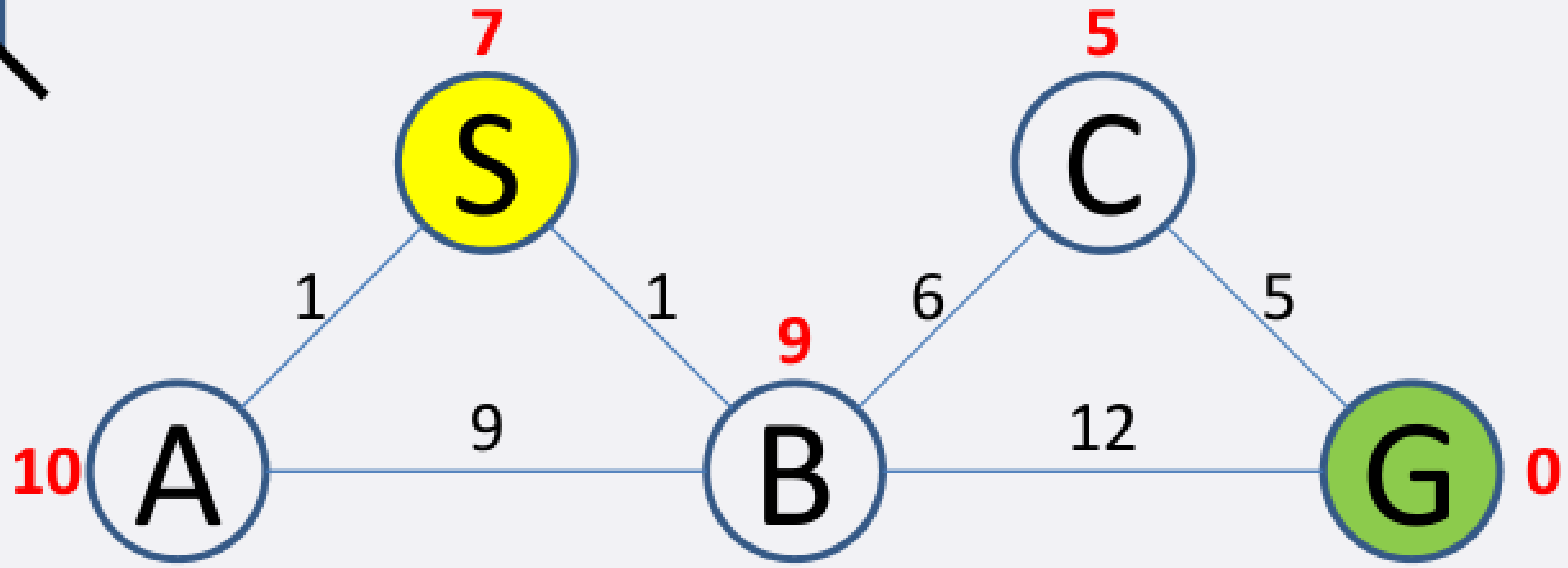
Exercise 3.12



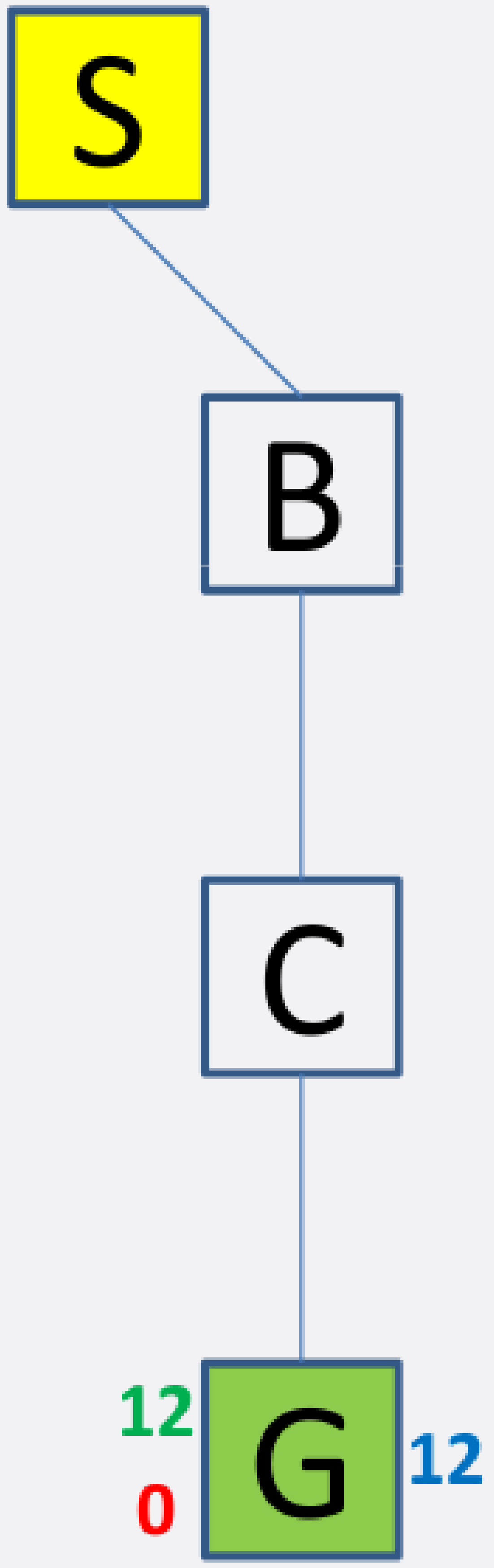
f = accumulated path cost + heuristic

```
IF QUEUE contains paths: P, Q
   AND P ends in node Ni && Q contains node Ni
   AND cost(P) ≥ cost(Q)
THEN remove P
```

QUEUE = < SBCG,**SBG**>



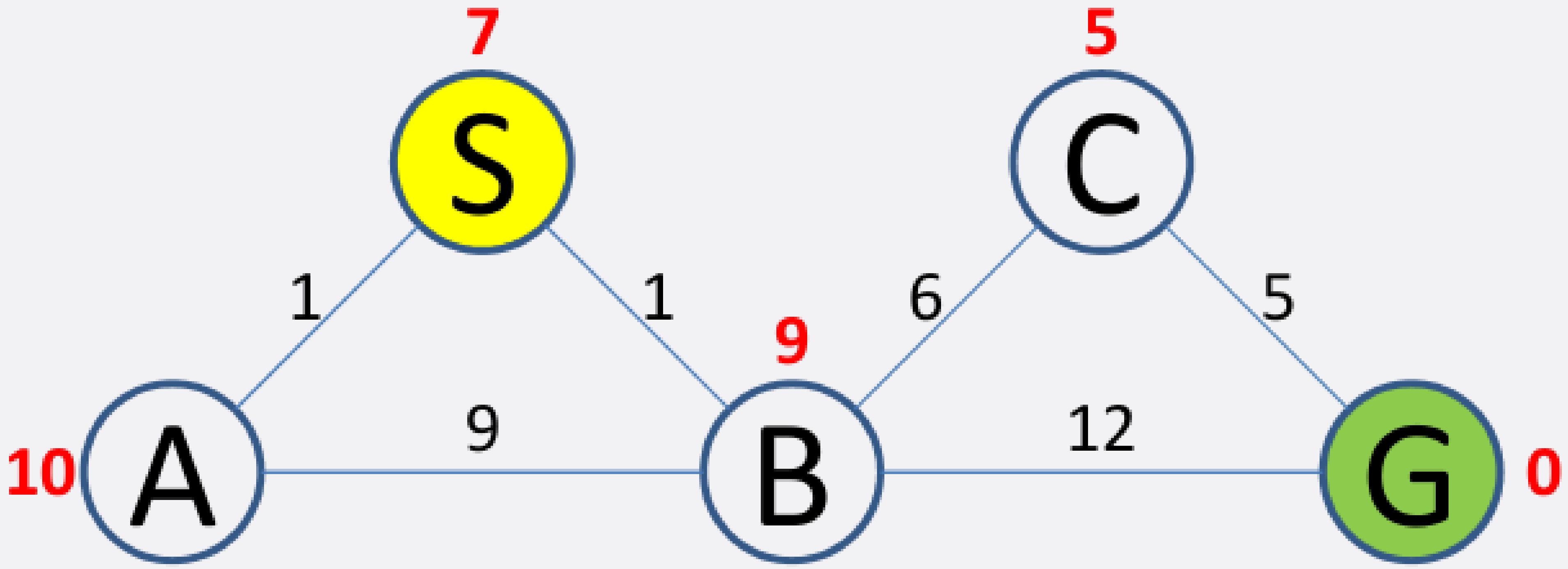
Exercise 3.12



f = accumulated path cost + heuristic

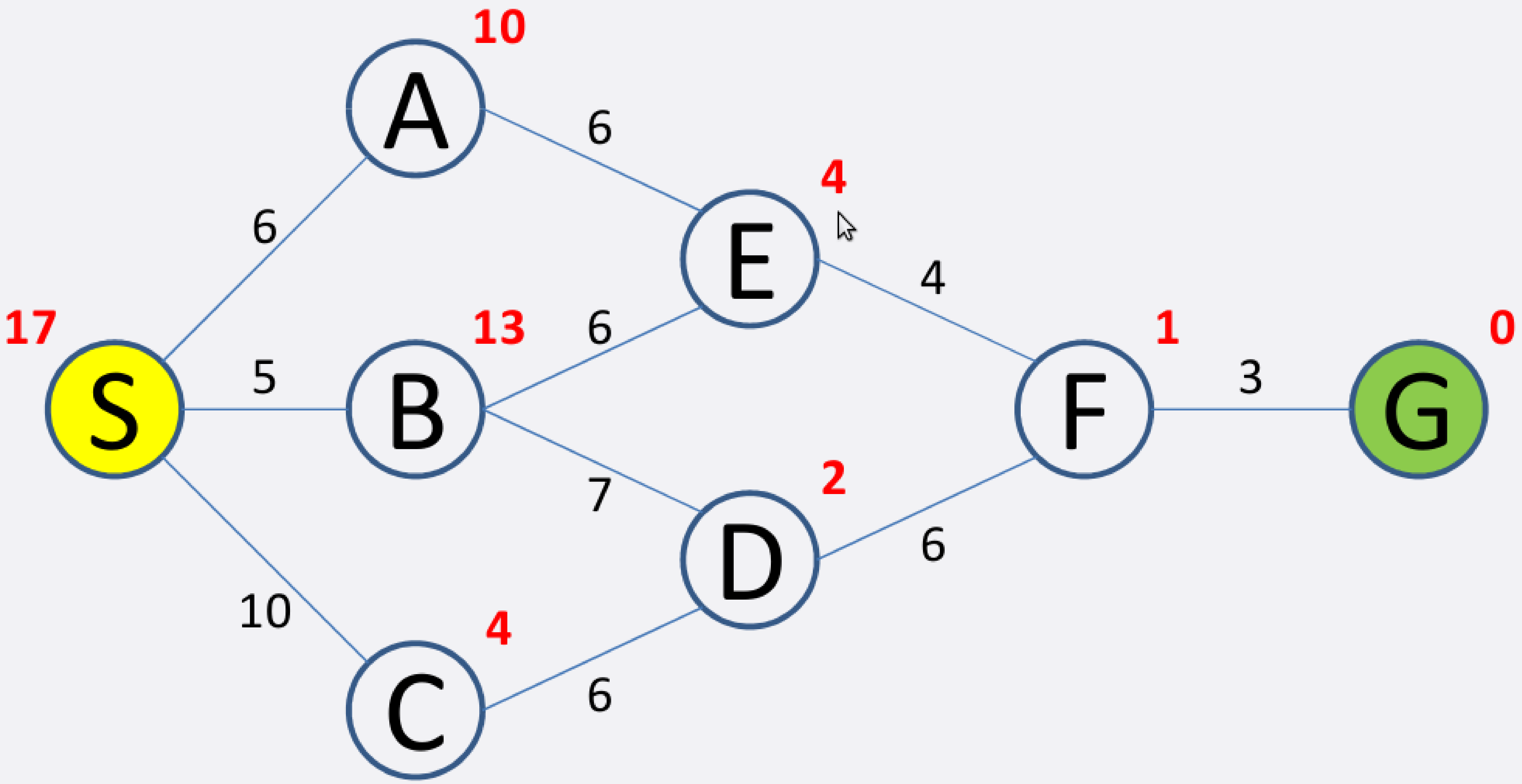
SUCCESS

QUEUE = < SBCG>



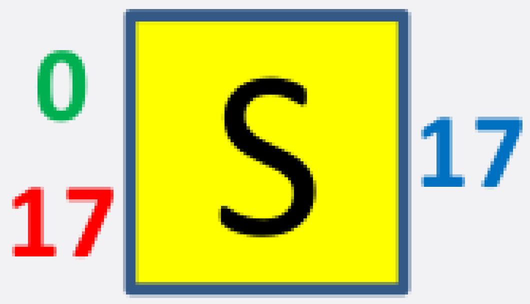
Exercise 3.13

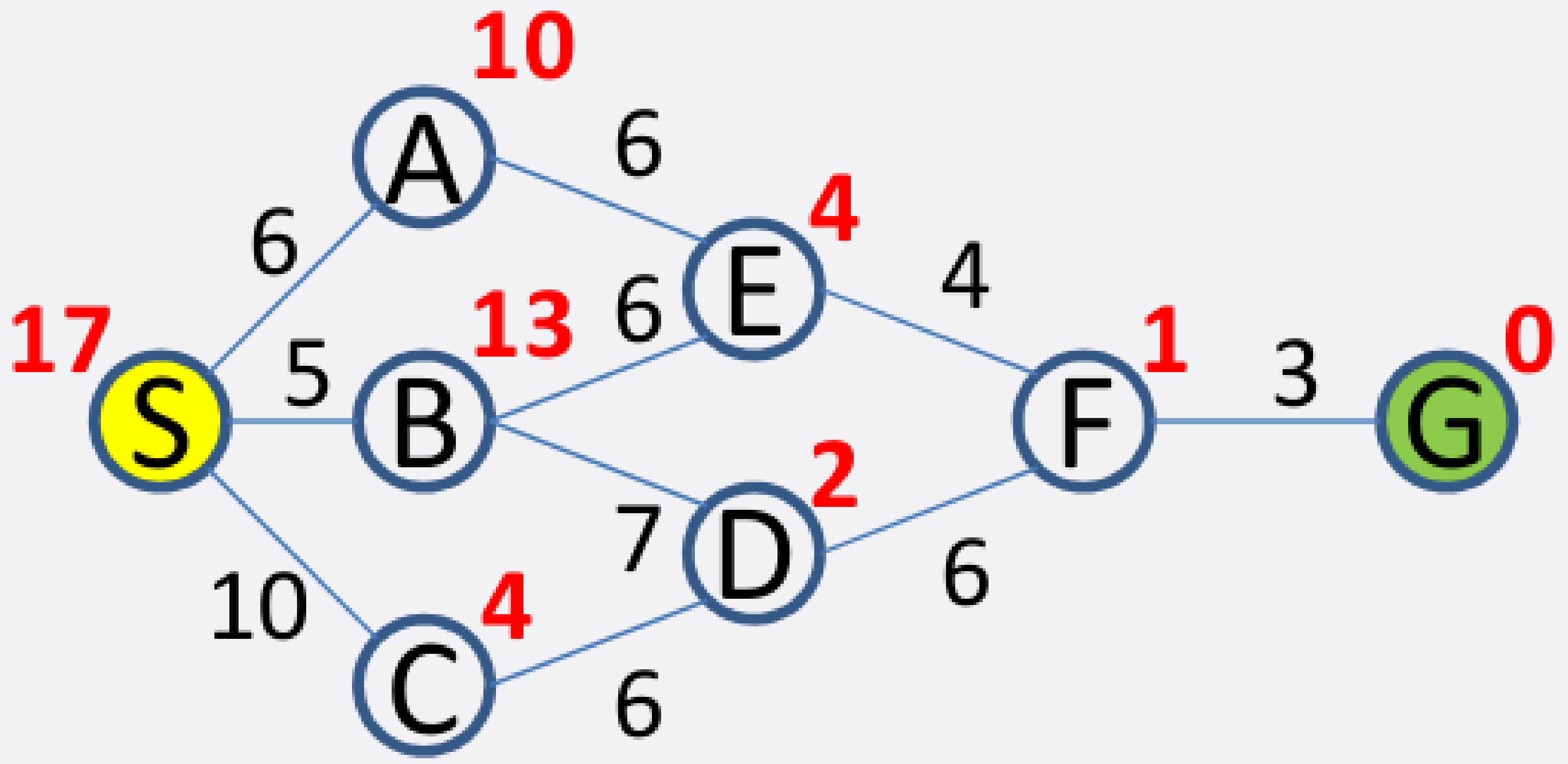
- Perform the A* Algorithm on the following figure. Explicitly write down the queue at each step.



Exercise 3.13

- Step 1



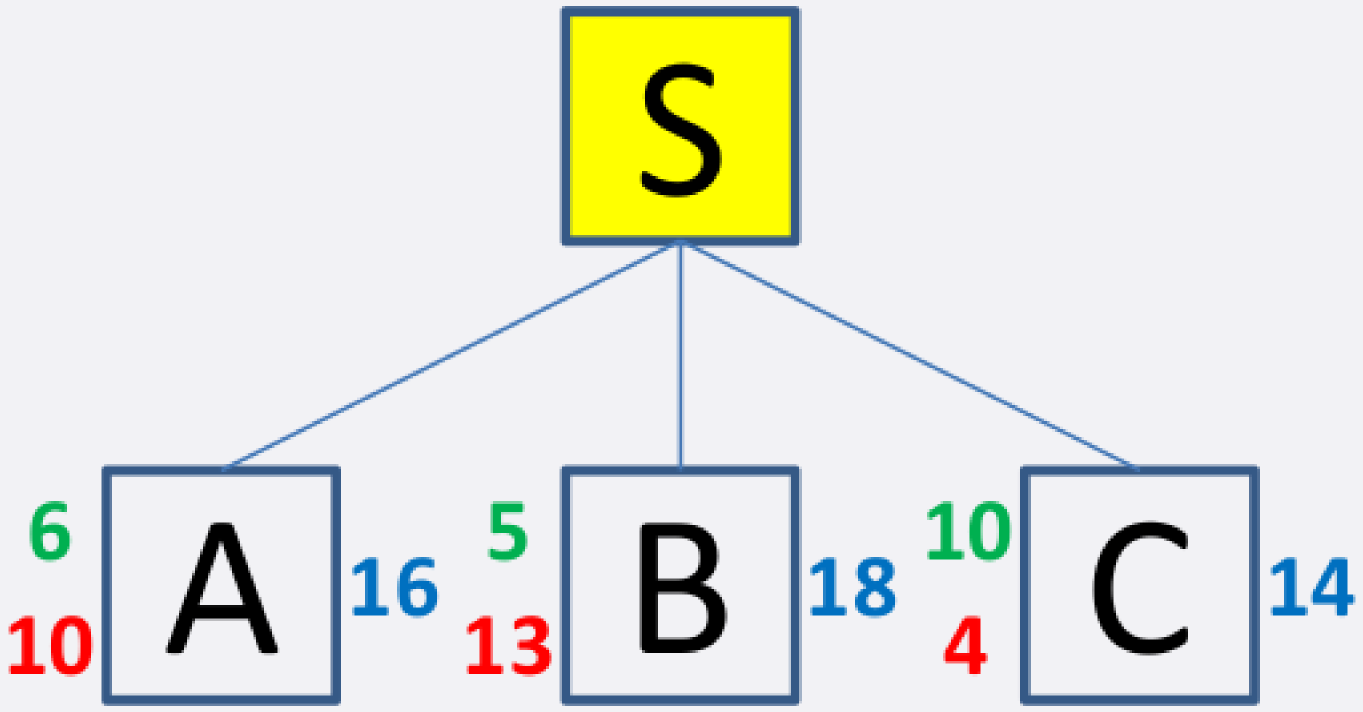


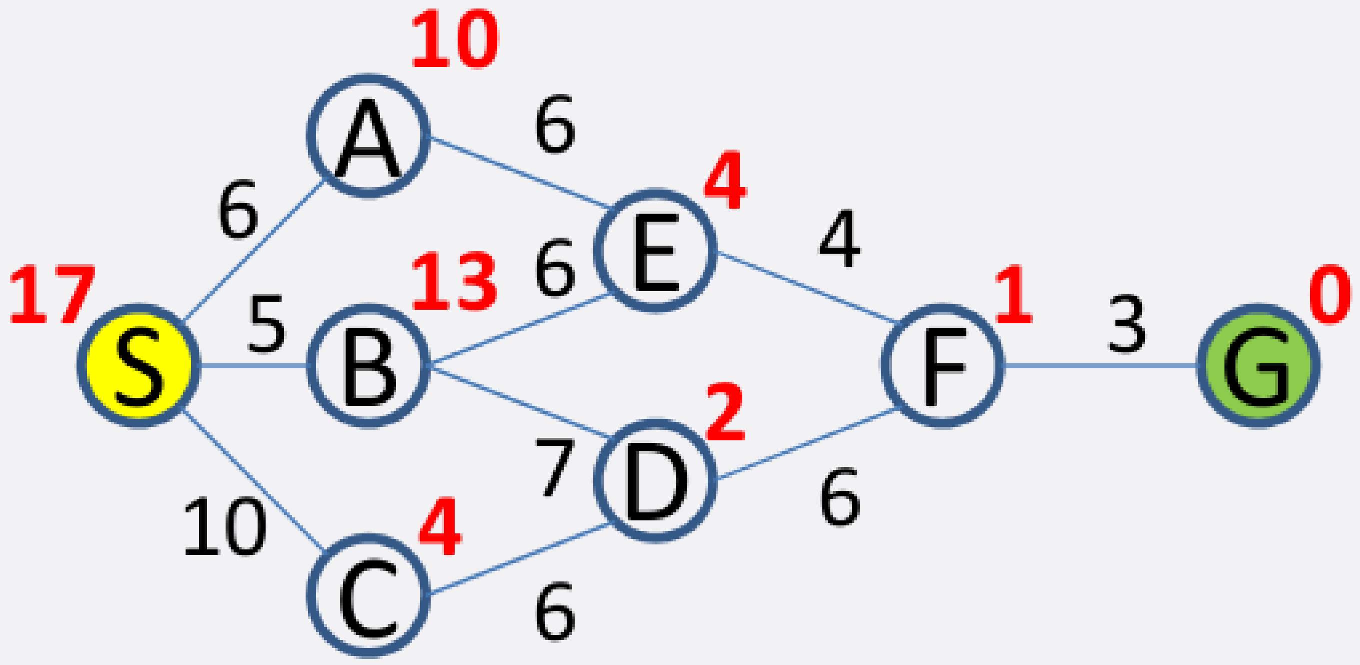
QUEUE:

S

Exercise 3.13

- Step 2





<u>QUEUE:</u>

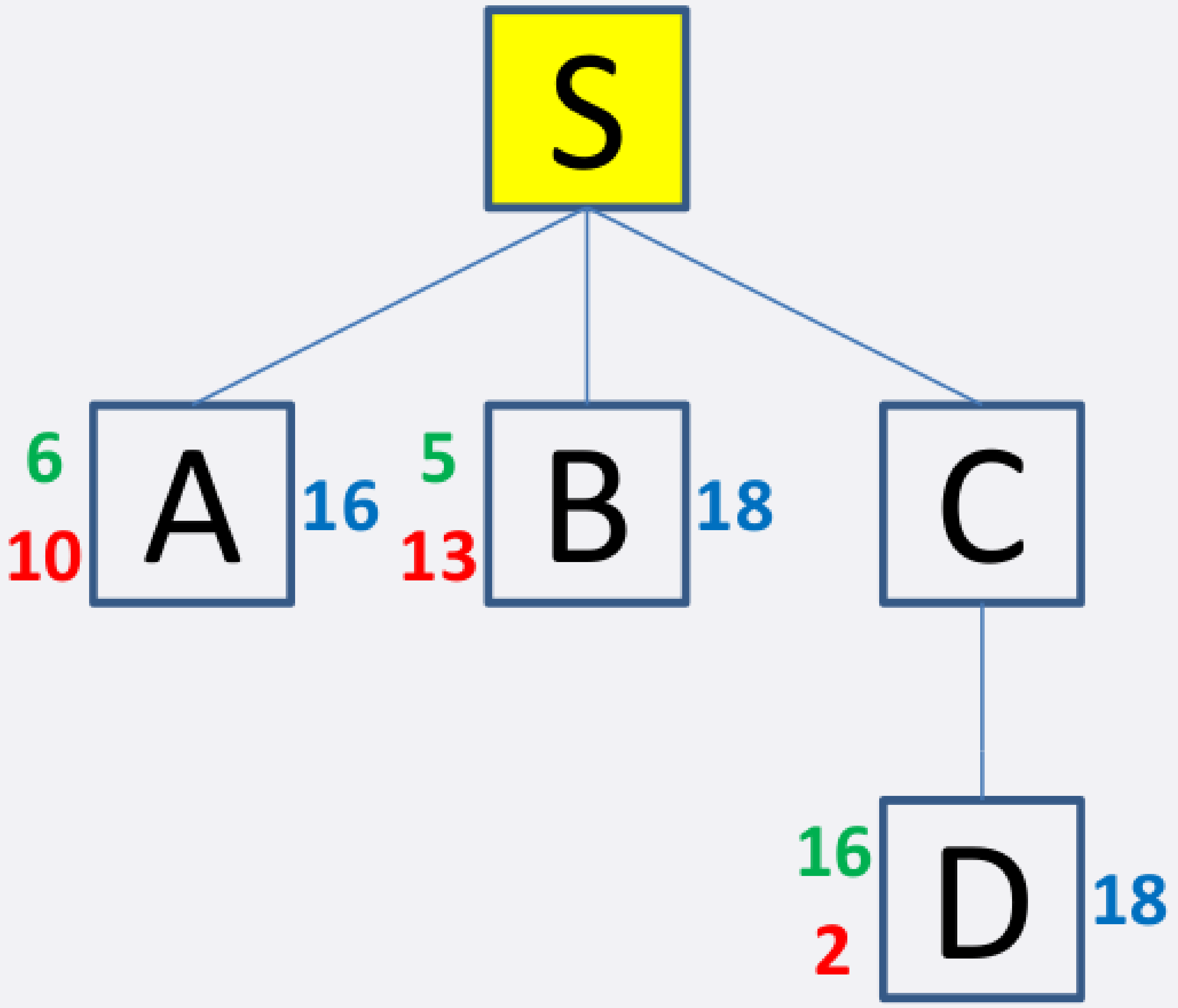
SC

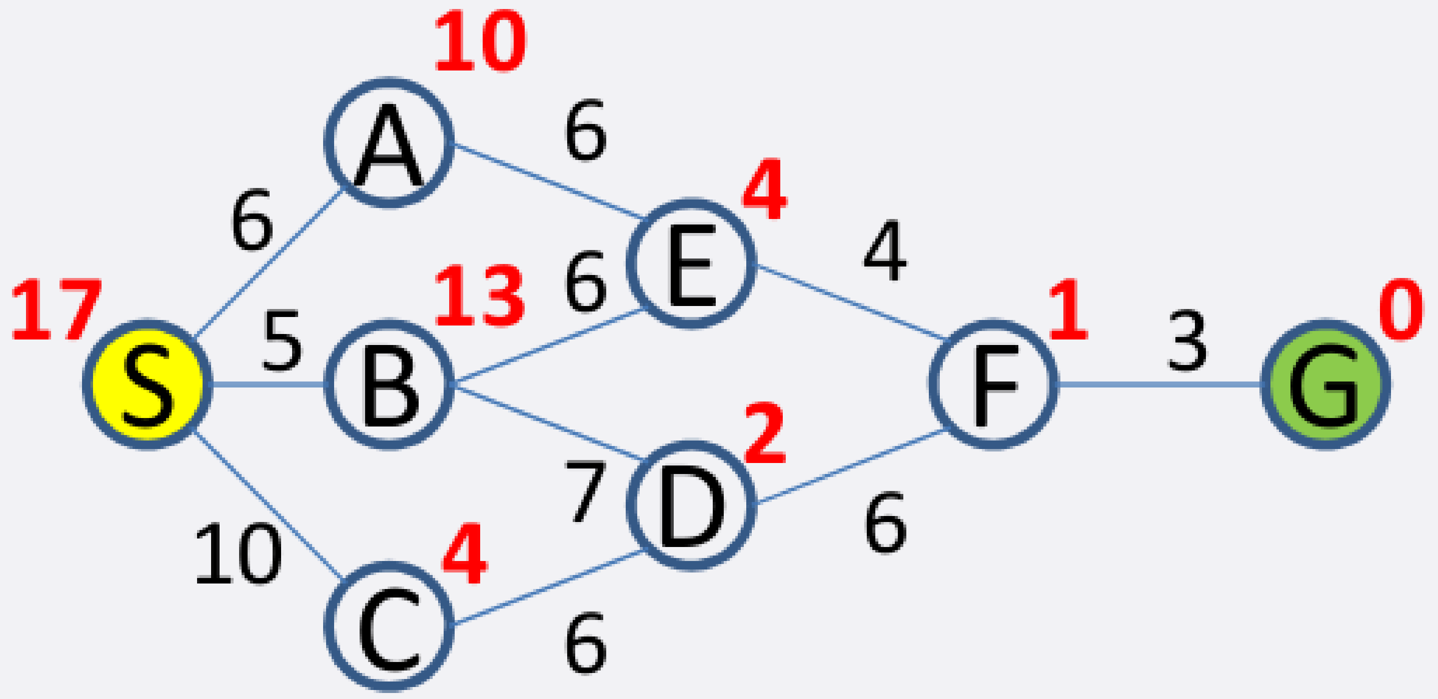
SA

SB

Exercise 3.13

- Step 3





QUEUE:

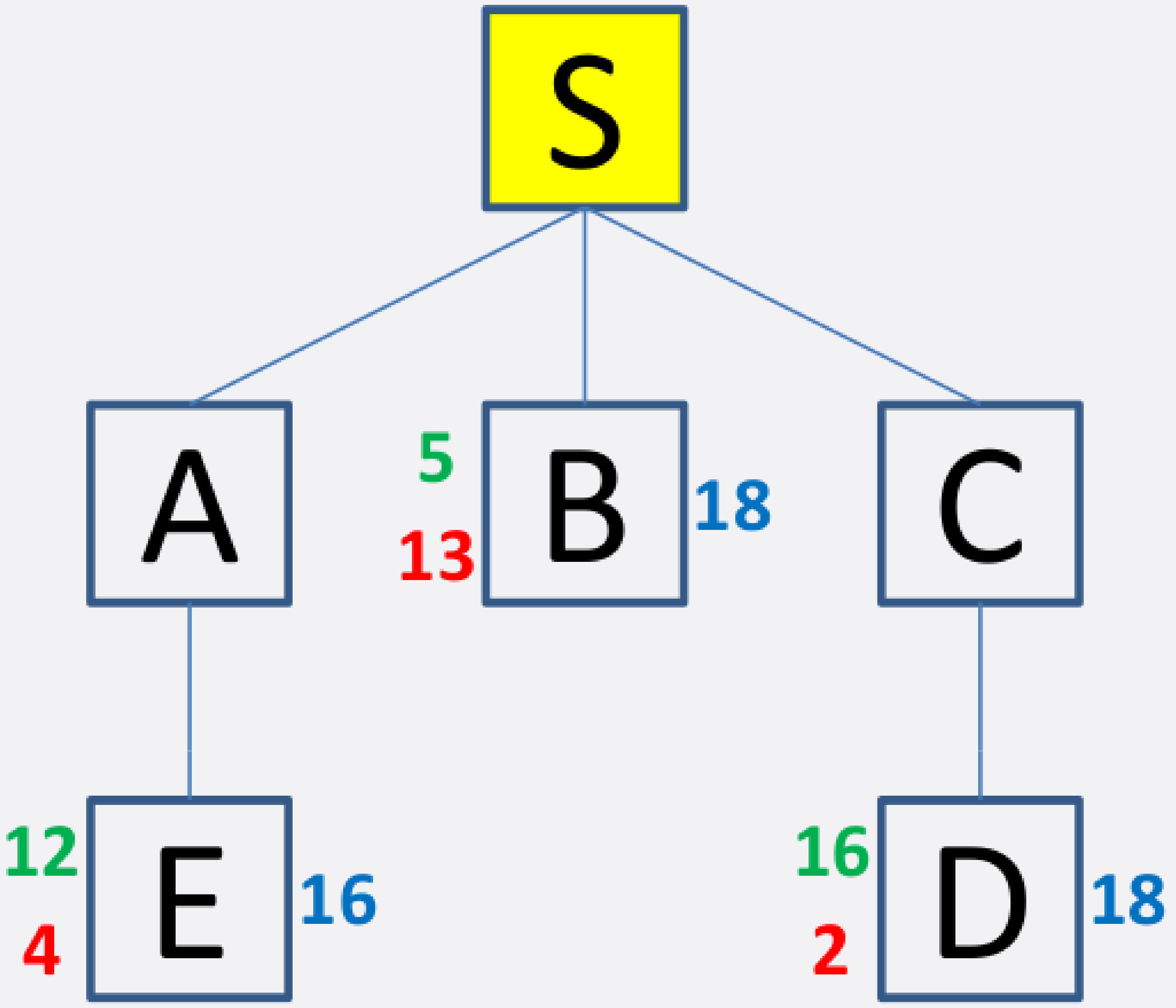
SA

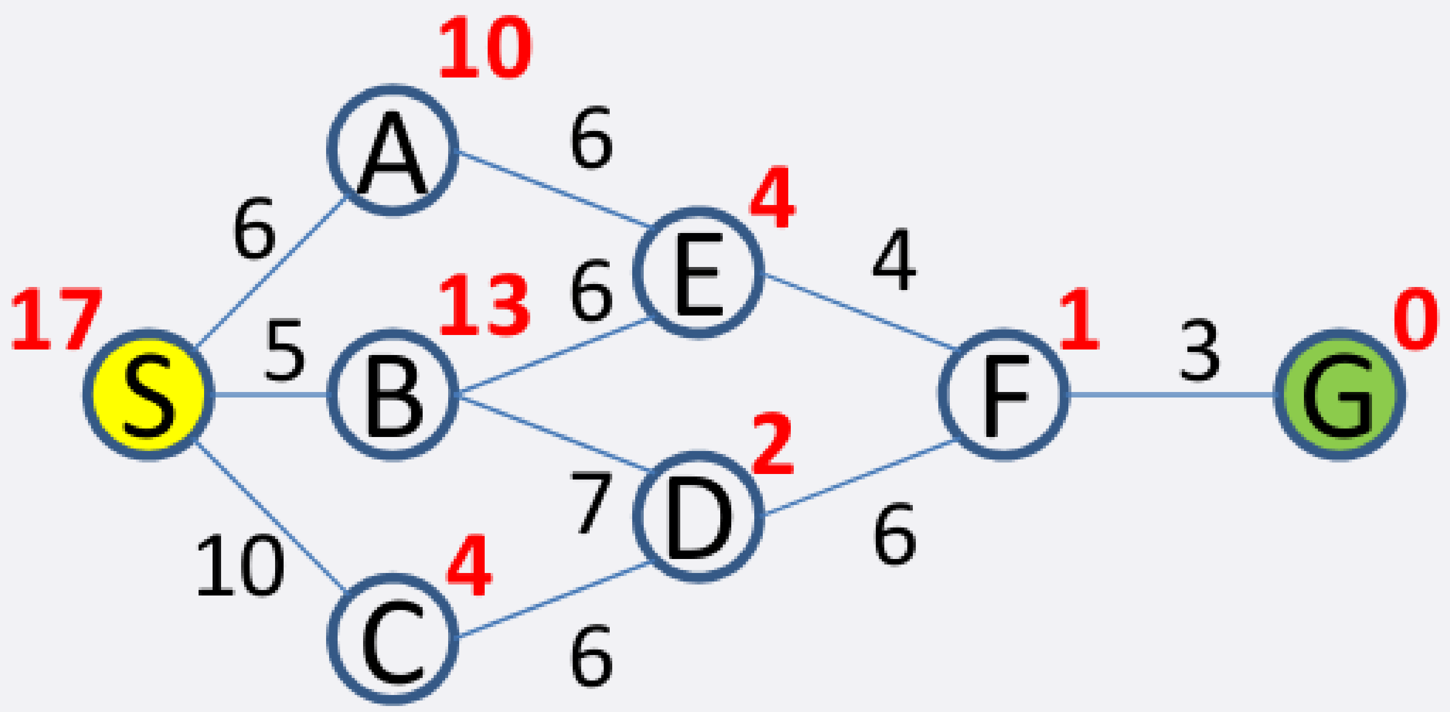
SCD

SB

Exercise 3.13

- Step 4





QUEUE:

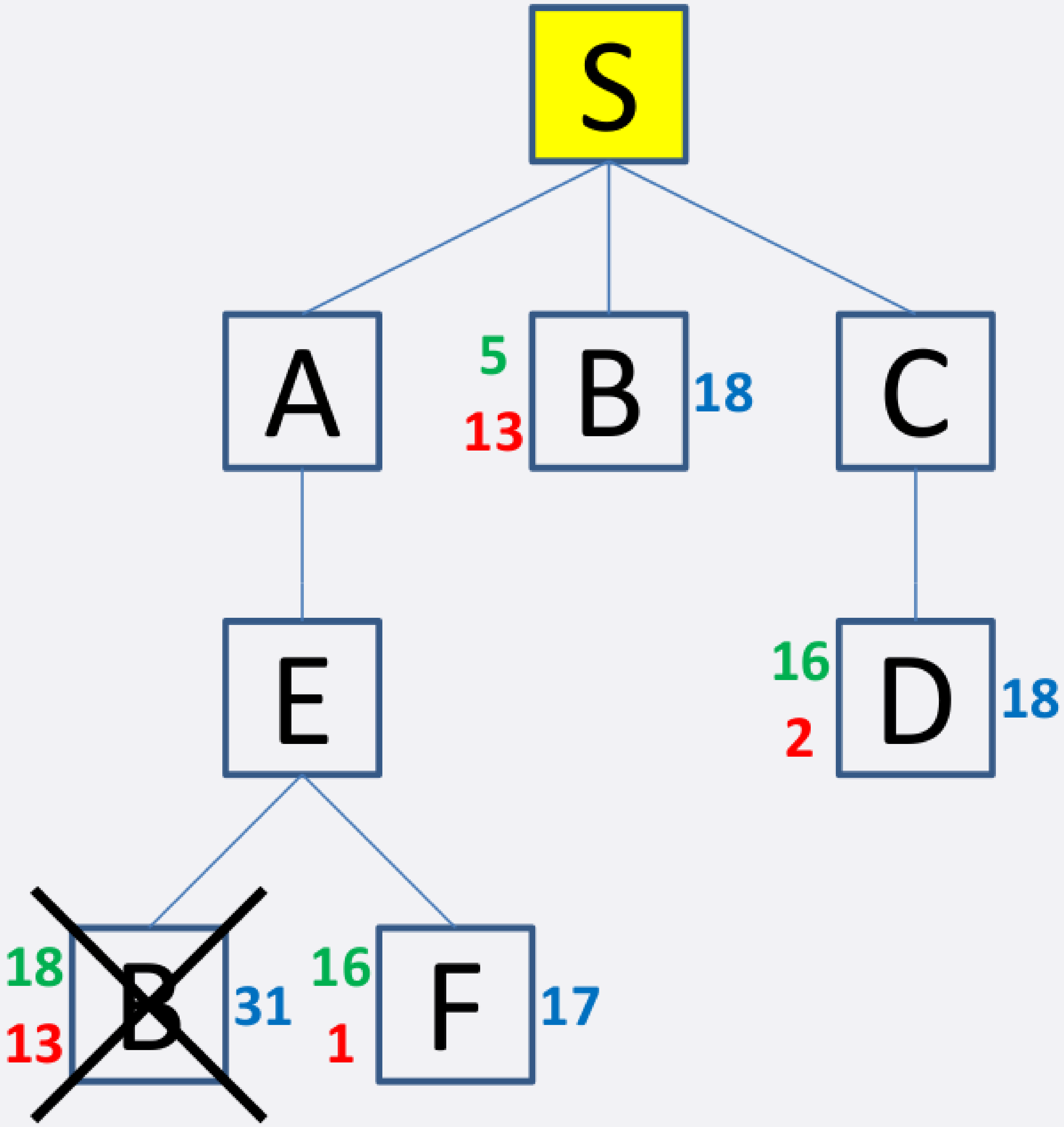
SAE

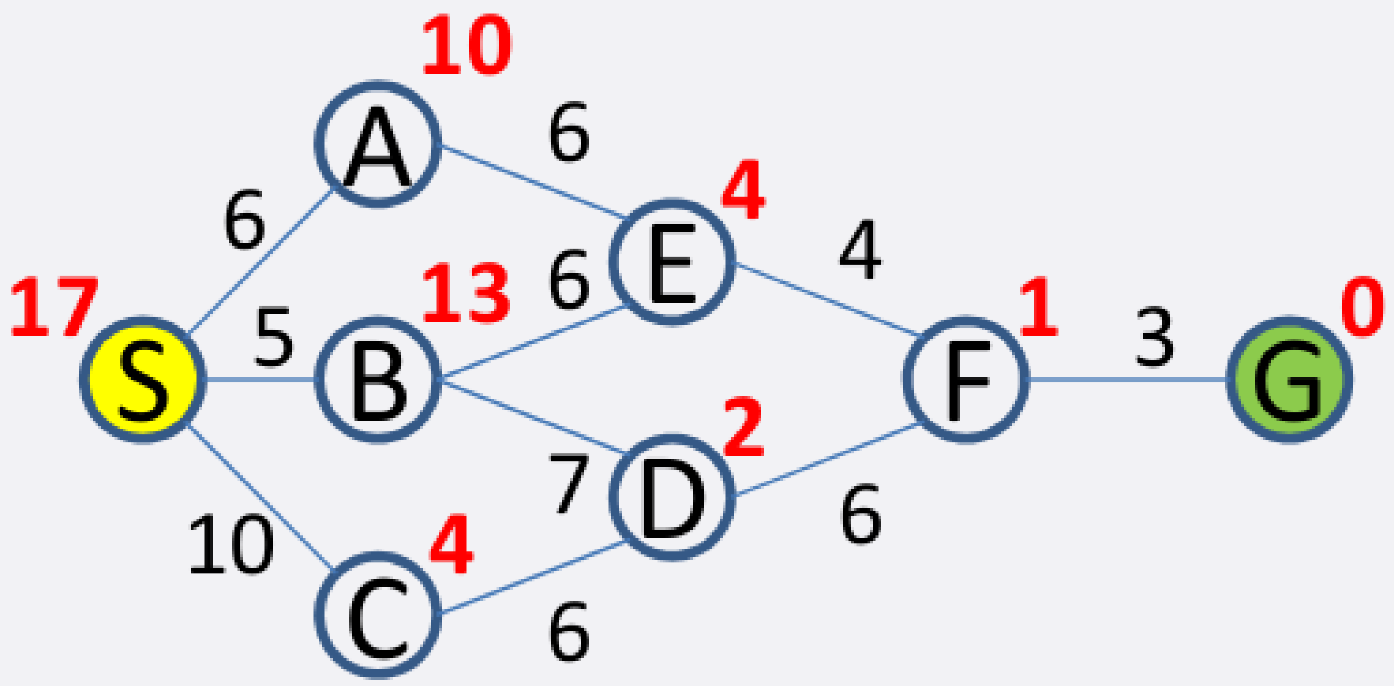
SCD

SB

Exercise 3.13

- Step 5





<u>QUEUE:</u>

SAEF

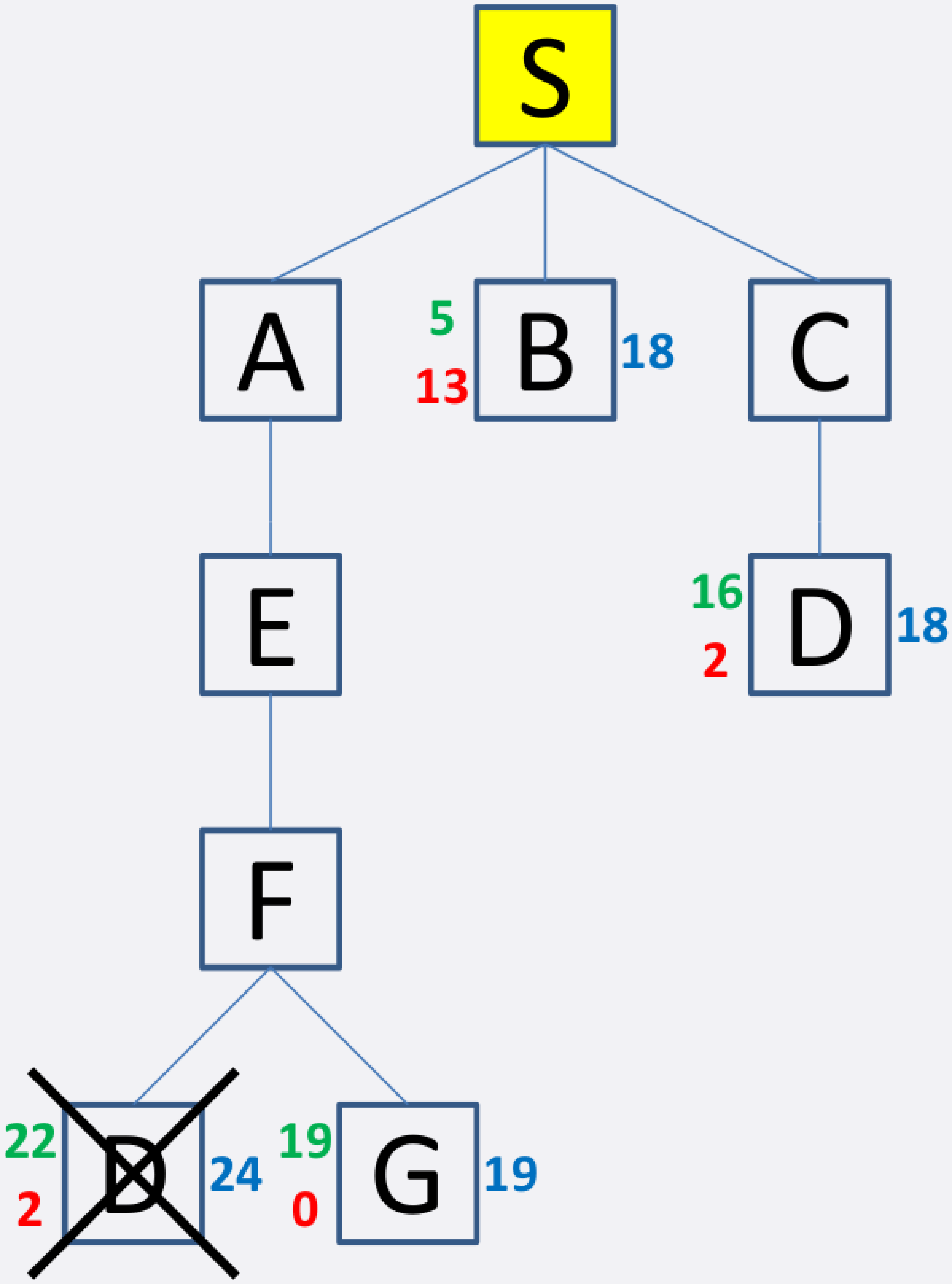
SCD

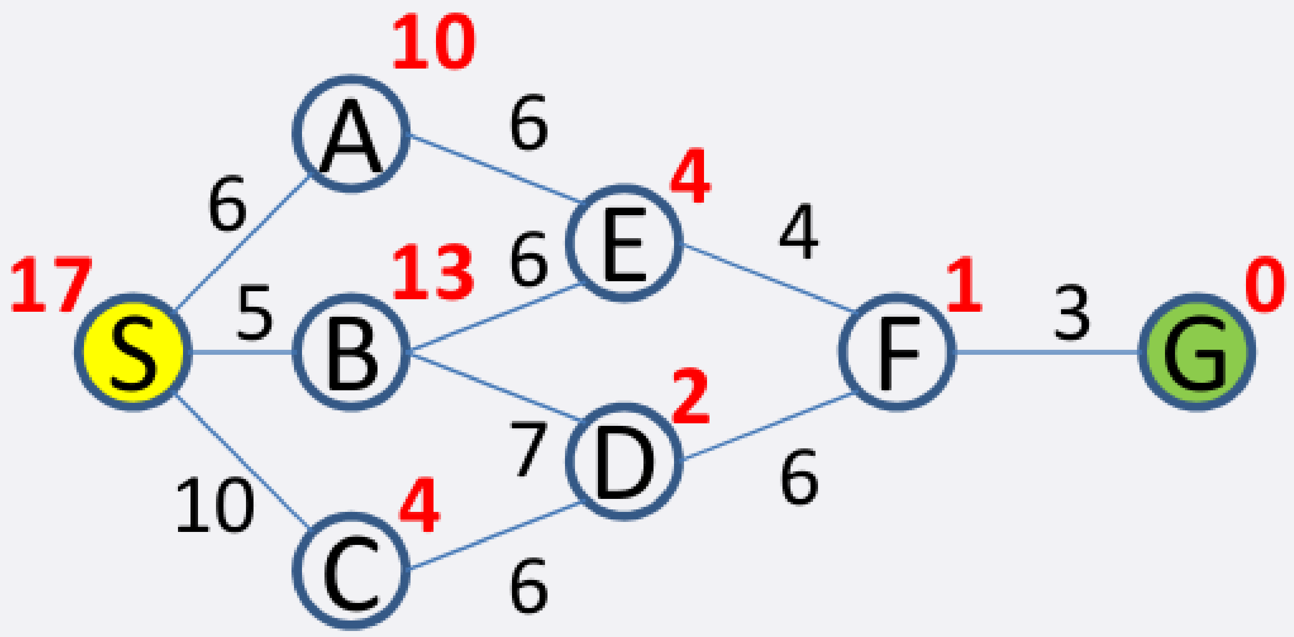
SB

<u>SAEB</u>

Exercise 3.13

- Step 6





QUEUE:

SCD

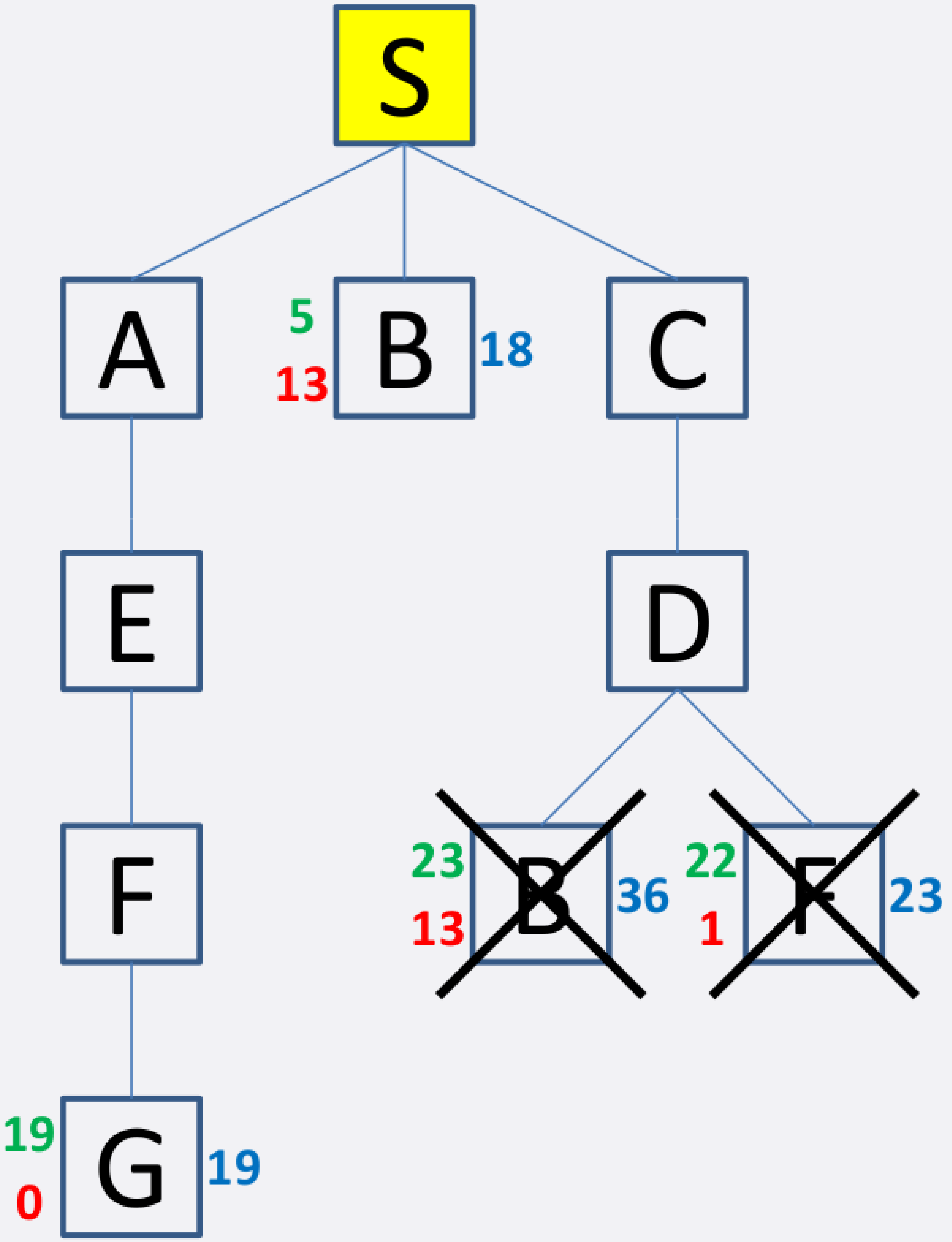
SB

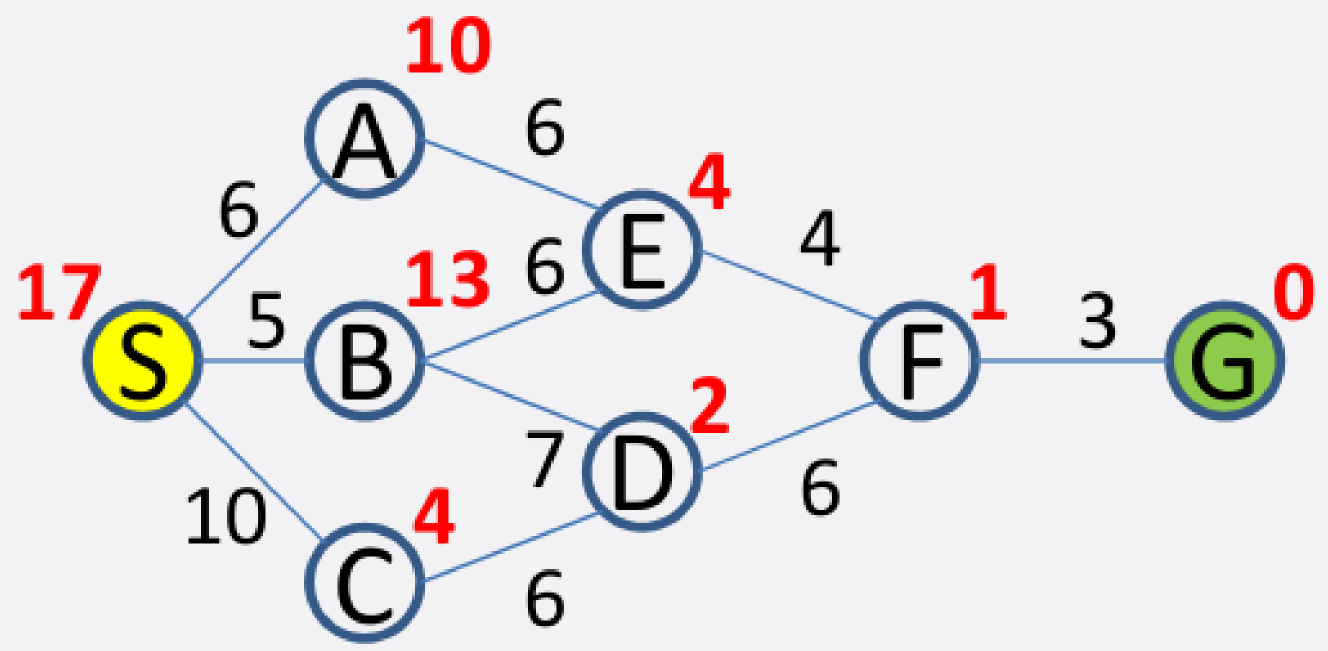
SAEFG

SAEFD

Exercise 3.13

- Step 7





QUEUE:

SB

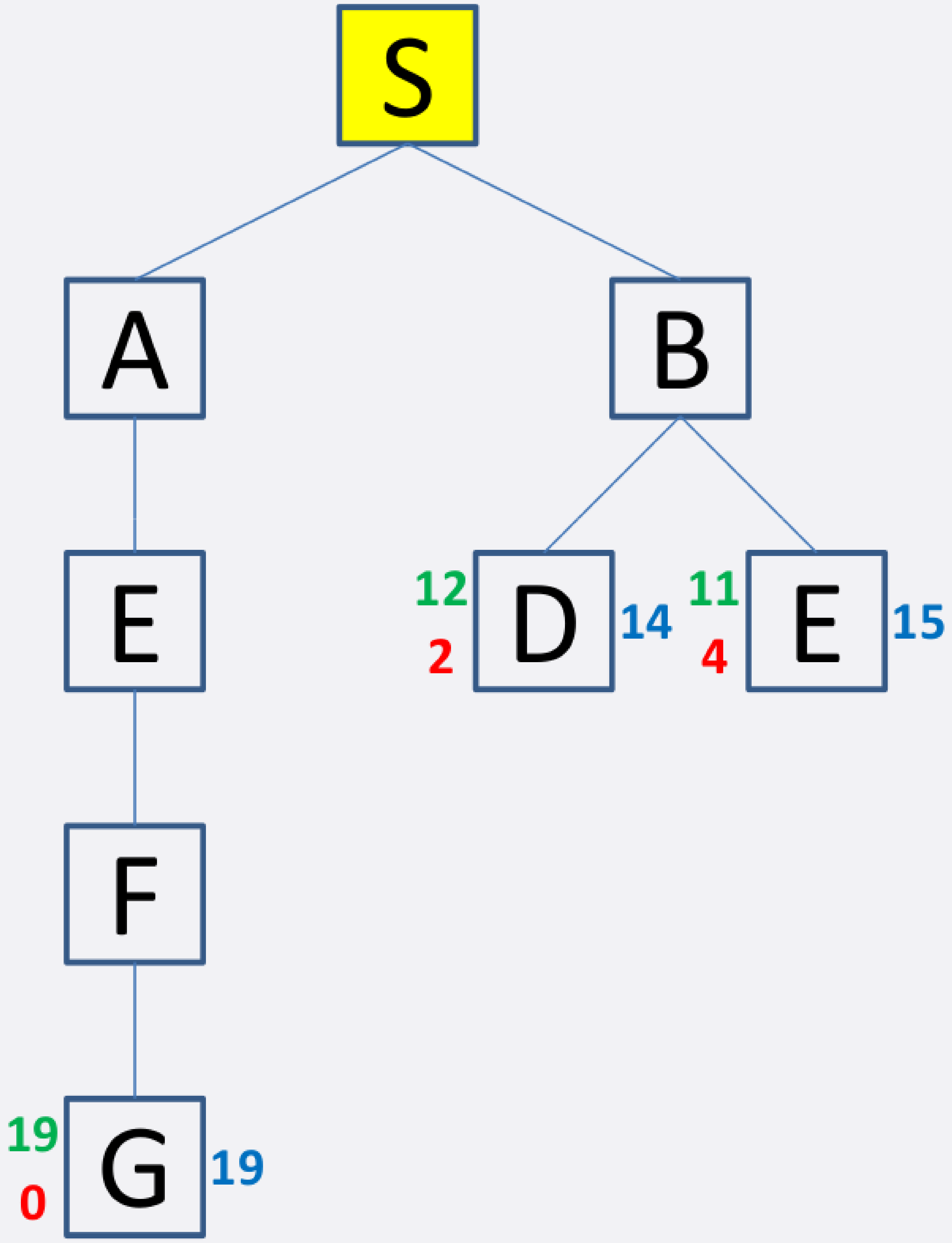
SAEFG

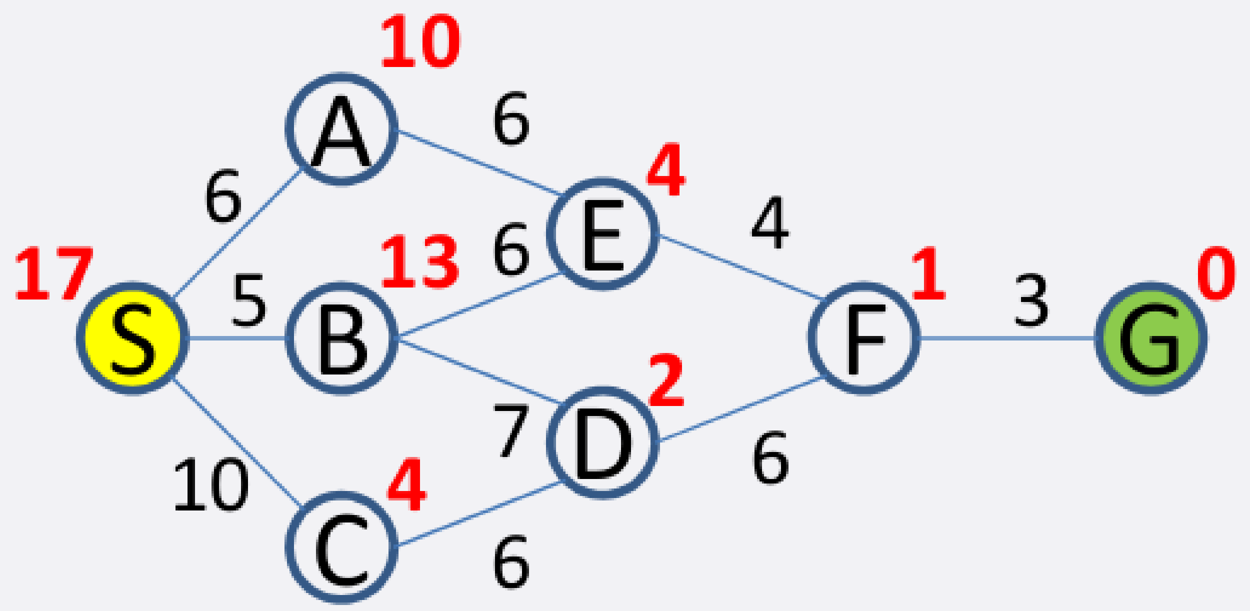
SCDF

SCDB

Exercise 3.13

- Step 8





QUEUE:

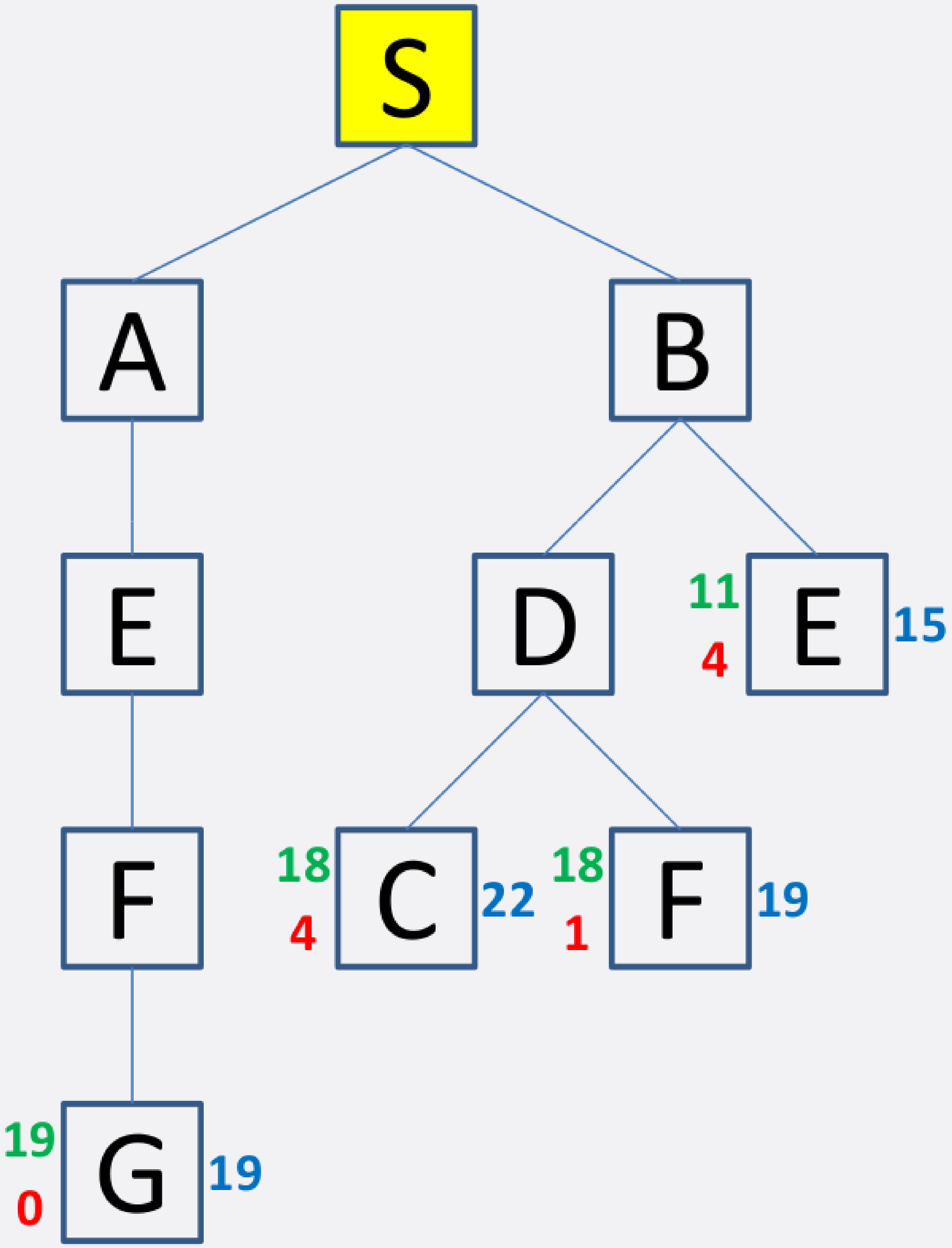
SBD

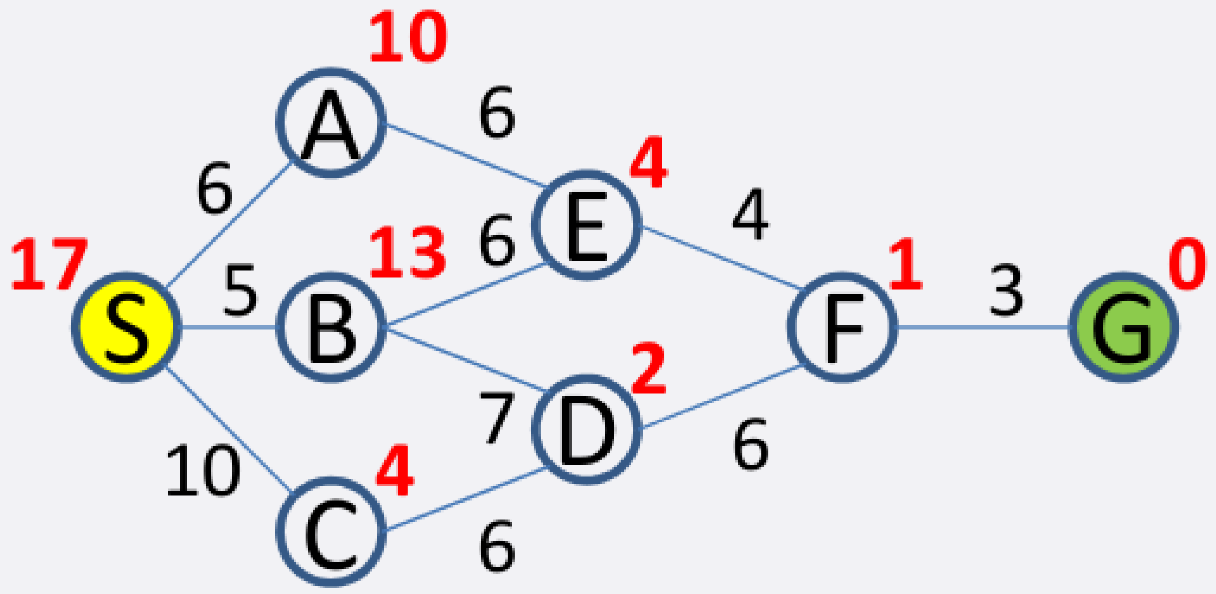
SBE

SAEFG

Exercise 3.13

- Step 9





QUEUE:

SBE

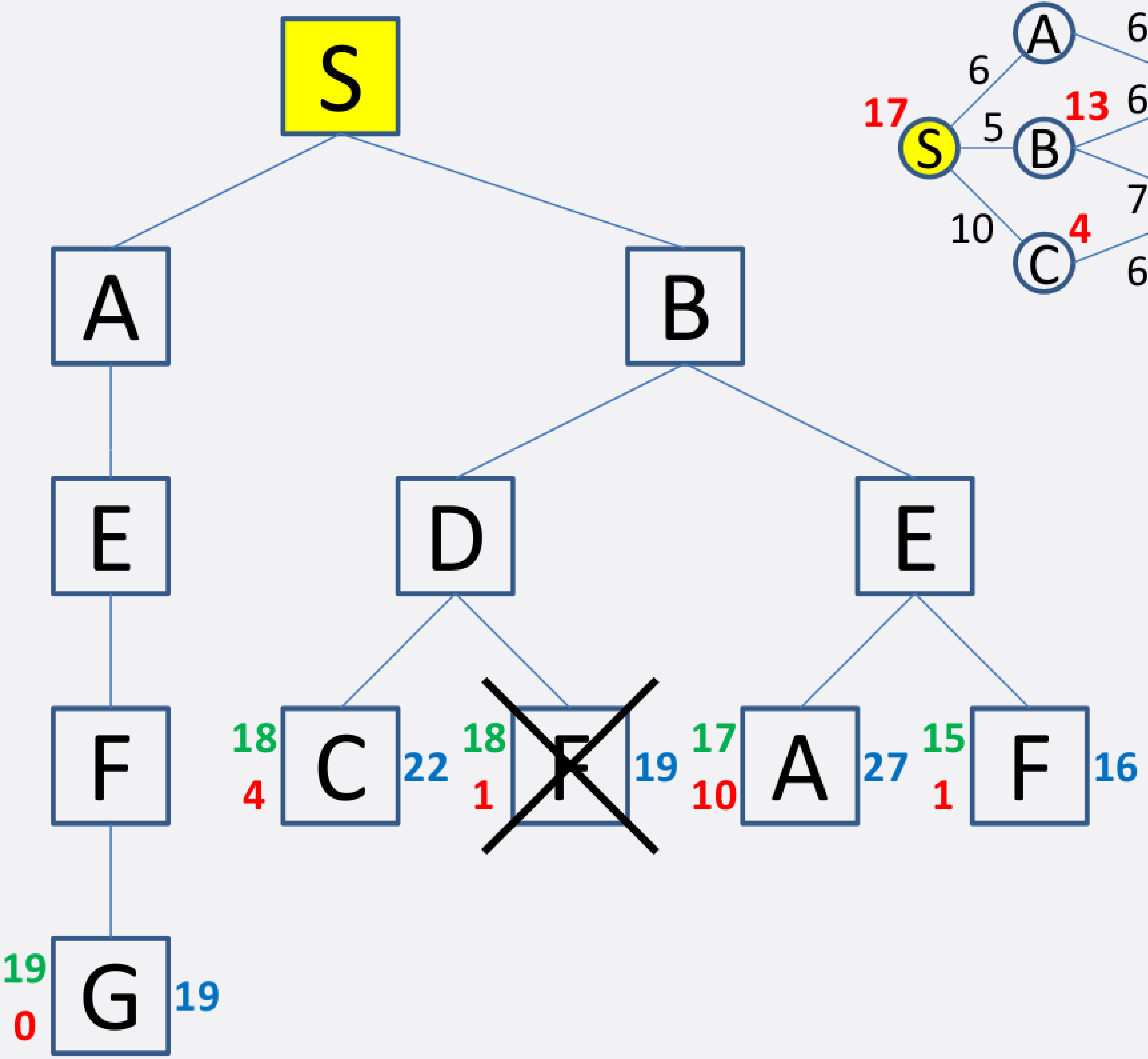
SBDF

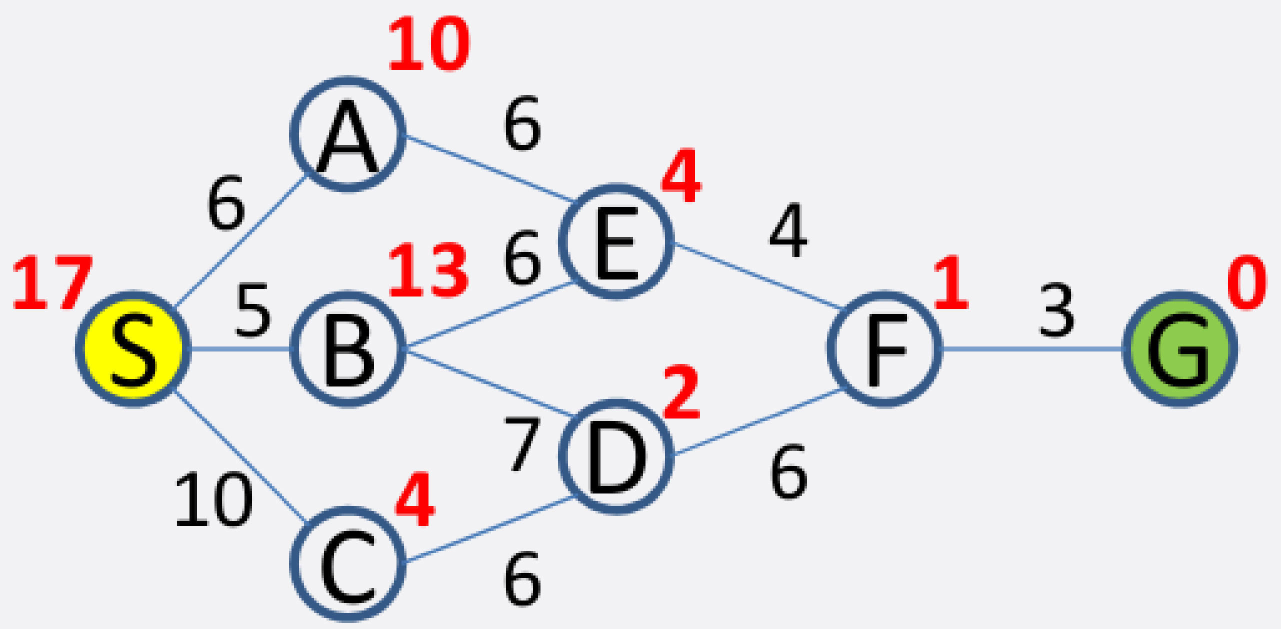
SAEFG

SBDC

Exercise 3.13

- Step 10





<u>**QUEUE:**</u>

SBEF

SAEFG

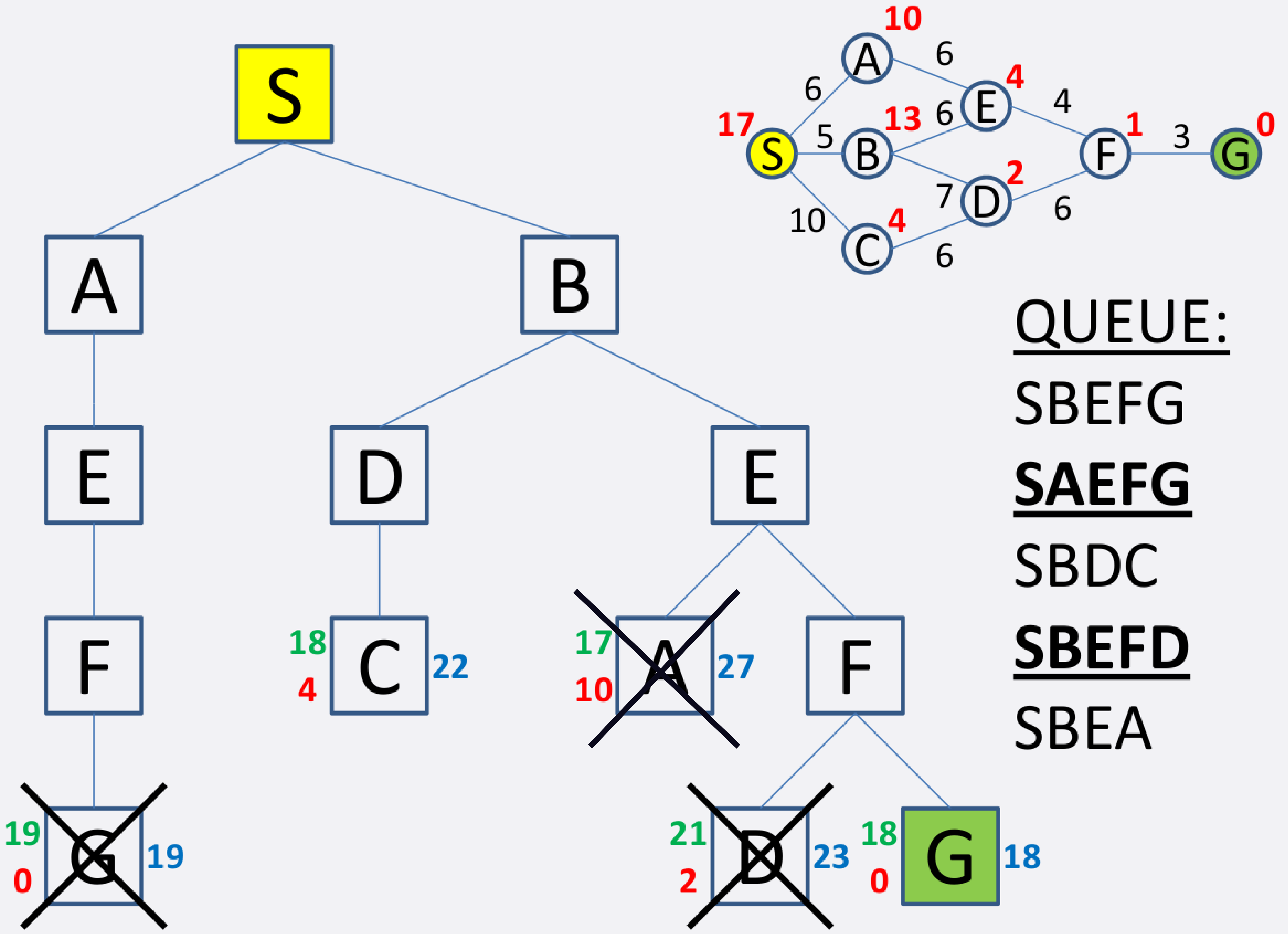
<u>**SBDF**</u>

SBDC

SBEA

Exercise 3.13

- Step 11



QUEUE:

SBEFG

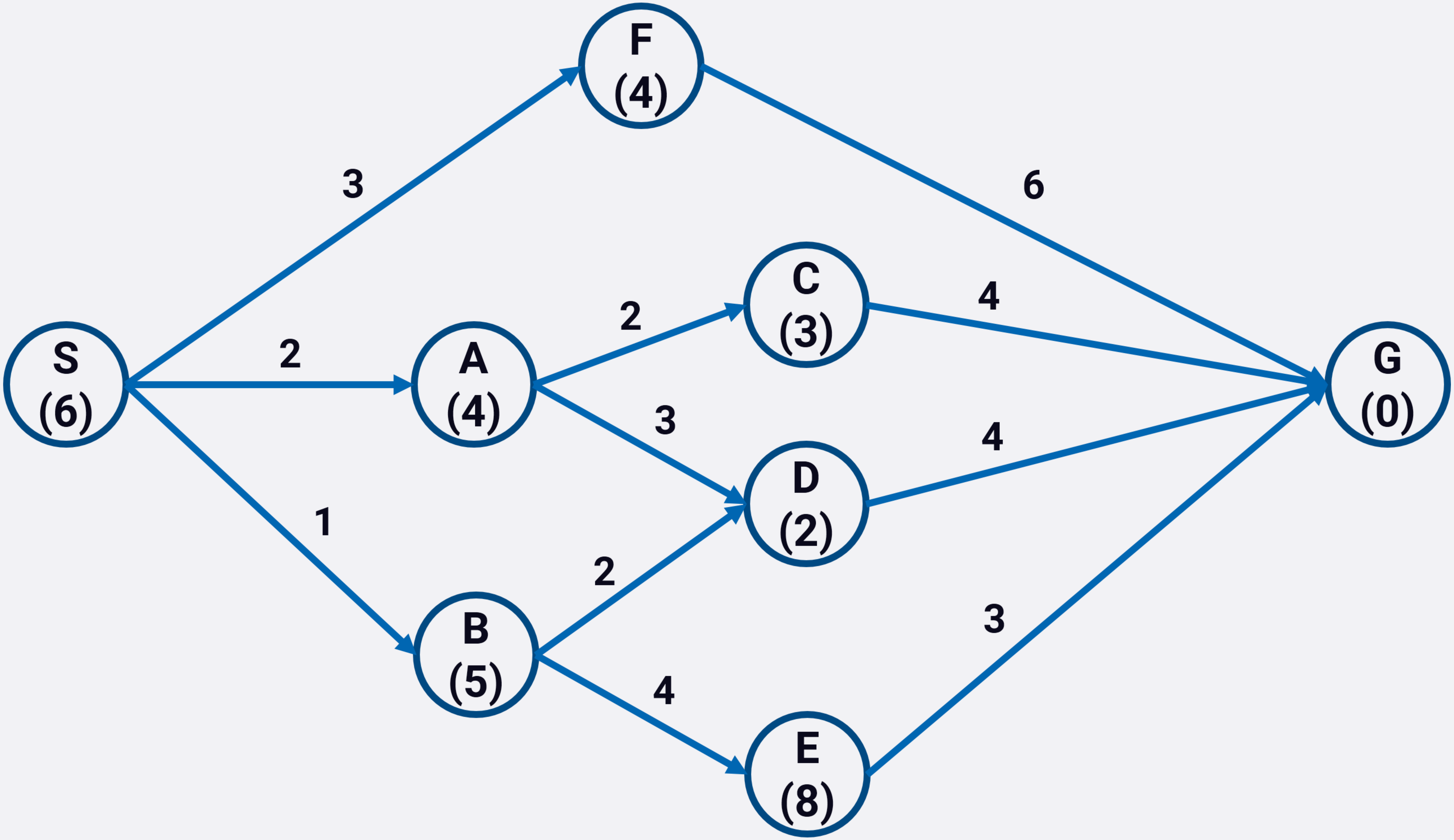
SAEFG

SBDC

SBEFD

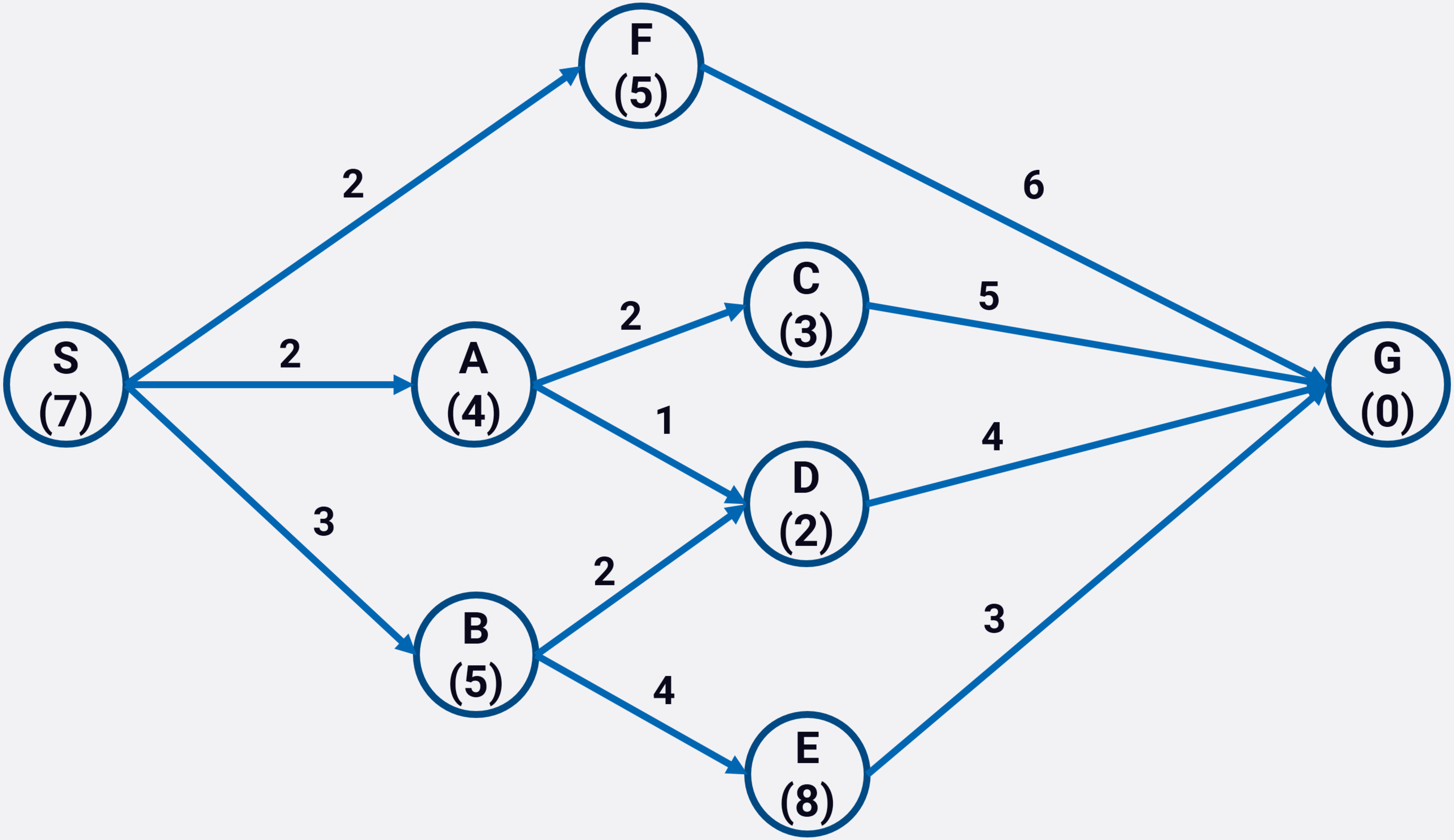
SBEA

Exercise 3.14



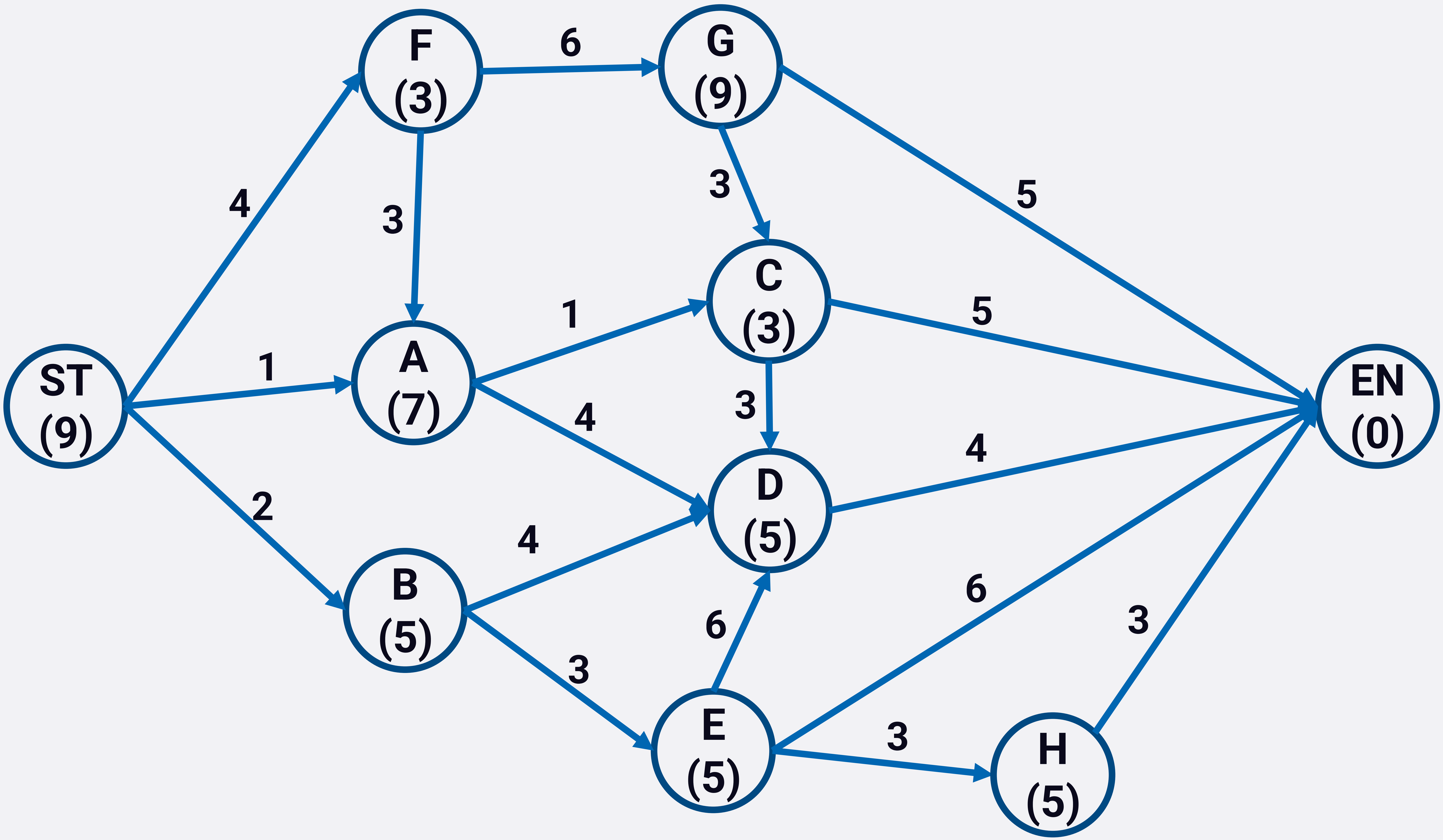
Exercise 3.14

Exercise 3.15

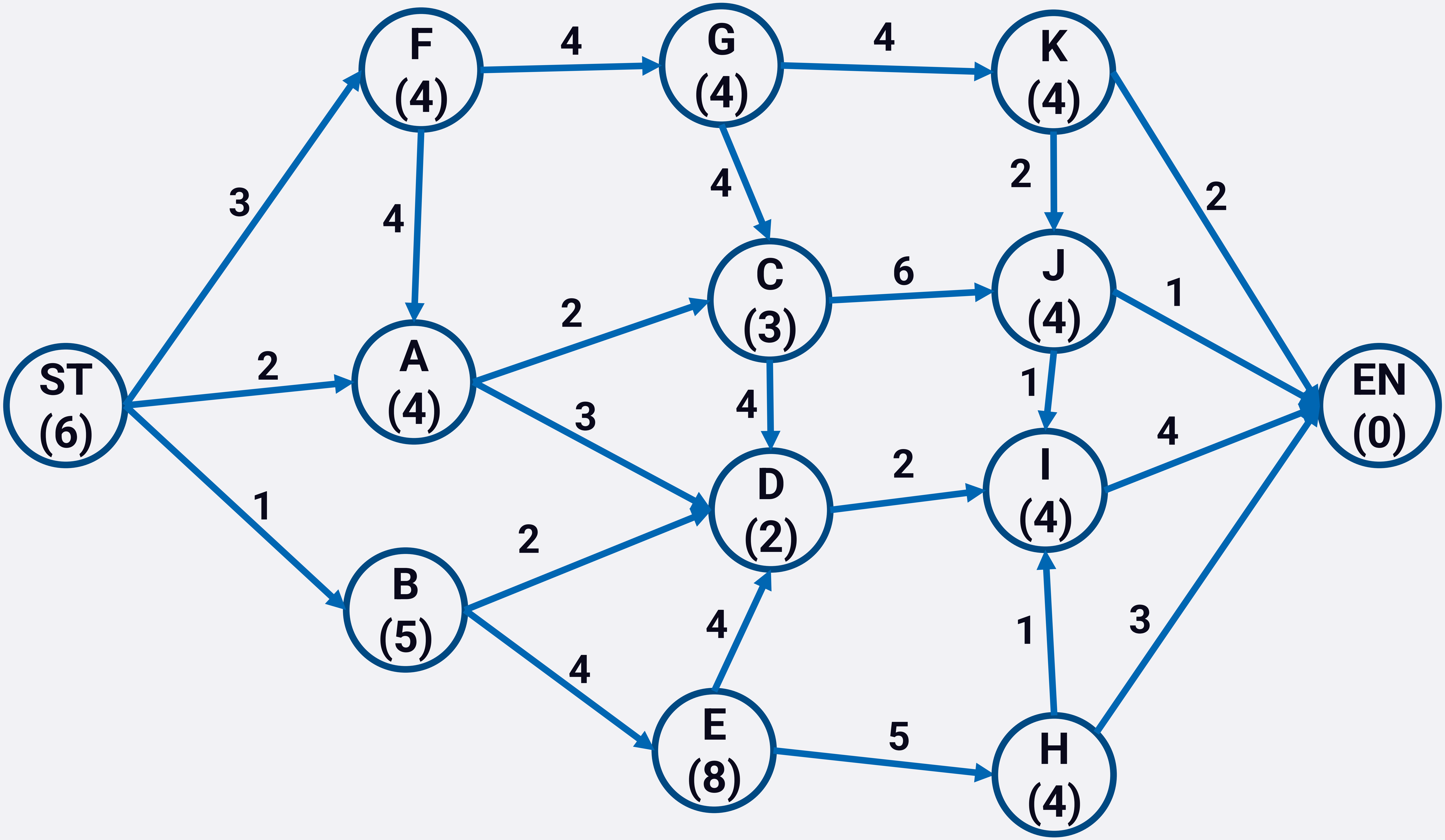


Exercise 3.15

Exercise 3.16



Exercise 3.17


F
(4)
4
G
(4)
4
K
(4)
3
4
4
2
2
C
(3)
6
J
(4)
2
1
ST
(6)
2
A
(4)
3
4
1
EN
(0)
1
D
(2)
2
I
(4)
4
2
B
(5)
4
1
3
4
E
(8)
5
H
(4)