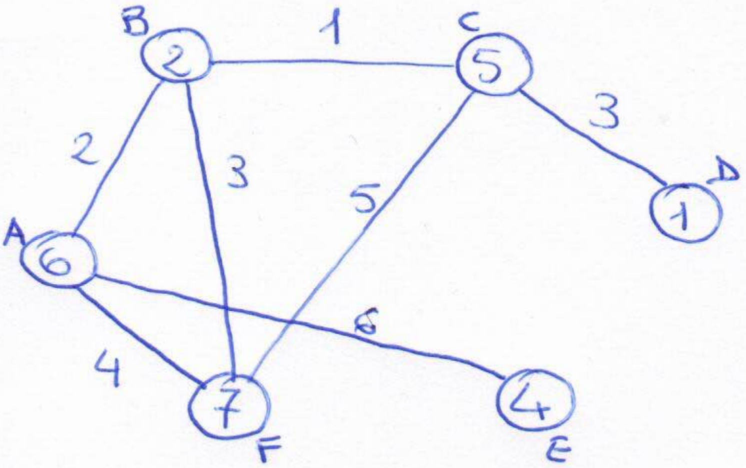
· current = ϕ



In the nodes are indicated $\pi(u)$, whereas outside there is u.

min π	max π	$c(u,v)$	u	v
1	5	3	C	D
2	5	1	B	C
2	6	2	A	B
2	7	3	B	F
4	6	6	A	E
5	7	5	C	F
6	7	4	A	F

this is actually a priority queue

↓

So we sort it first by min π then by $c(u,v)$.

Extract min Q $\longrightarrow$ $\langle 1, 5, 3, C, D\rangle \longrightarrow u = 1, v = 5$

since current $\neq 1 = u \longrightarrow$ current $= 1$, ~~redir~~ relinkTo $= 5$

output $\langle C, D\rangle$

$1 \longrightarrow 5$

There two nodes are merged as indicated

Extract min Q $\rightarrow$ $\langle 2, 5, 1, B, C\rangle \rightarrow u = 2, v = 5$

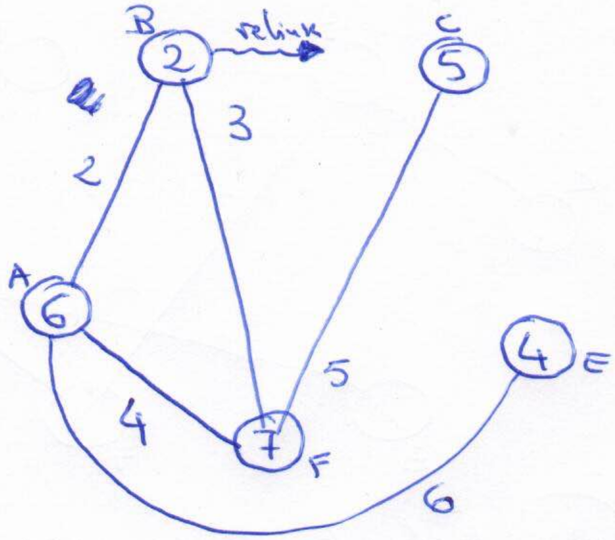
since current $\neq 2 \overset{u}{\longrightarrow}$ current $= 2$, relinkTo $= 5$

output $\langle B, C\rangle$

$2 \longrightarrow 5$

all edges incident in 2 will be moved to 5 by the next steps.

We draw the graph for simplicity at this point:



min π	max π	c	u	v
2	6	2	A	B
2	7	3	B	F
4	6	6	A	E
5	7	5	C	F
6	7	4	A	F

current = 2 , relinkTo = 5

The algorithm at this point proceeds as follows:

min Q → < 2, 6, 2, A, B > → u = 2, v = 6

since current = 2 $\overset{\shortparallel}{u}$ ⟹ we have to relink the edge changing u → relinkTo

Q.insert (5, 6, 2, A, B)

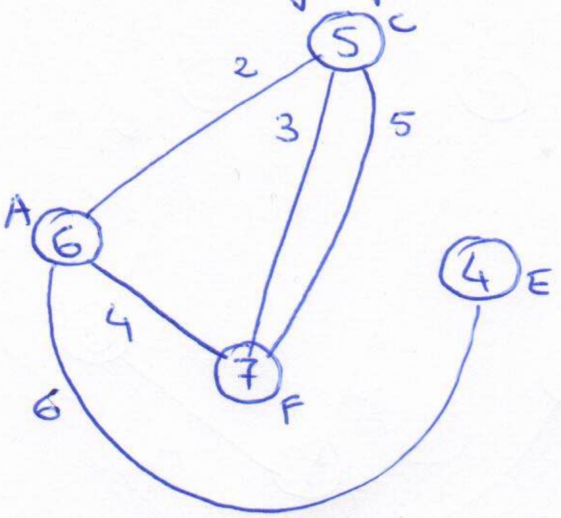
Before plotting the next graph and priority queue, let us do another step, since the next edge is also outgoing from 2 and thus it'll be relinkto 5 ;

min Q → < 2, 7, 3, B, F > → u = 2, v = 7

since current = 2 = u ⟹ we have to relink 2 — 7 to 5 — 7

Q.insert (5, 7, 3, B, F)

We draw the graph at this point, and the current PQ



min π	max π	c	u	v
4	6	6	A	E
5	6	2	A	B
5	7	3	B	F
5	7	5	C	F
6	7	4	A	F

current = 2, relinkTo = 5

The algorithm at this point does:

min Q → < 4, 6, 6, A, E > ⟶ u = 4, v = 6

since current ≠ 4 = v ⟹ we have to ~~relink~~ change ⟹ $\boxed{\text{current} = 4 \quad \text{relinkTo} = 6}$

output < A, E >

This way ④ ⇝ ⑥ hence every edge linked to ④ has to be relinked to ⑥, if any (we actually do not have them).

notice that min Q is extracted and ~~thus~~ thus removed from Q

min Q → < 5, 6, 2, A, B > ⟹ u = 5, v = 6

since current ≠ 5 = u ⤳ current = 5, relinkTo = 6

This way ⑤ ⇝ ⑥ hence every edge incident in ⑤ will be relinked to ⑥

output < A, B >

extract min Q

min Q → $\langle 5, 7, 3, B, F \rangle$ → $u = 5$, $v = 7$

since current = 5 = u ⇒ we have to relink

5 — 7 ⇝ 6 — 7

Q.insert (6, 7, 3, B, F)

=

min Q → $\langle 5, 7, 5, C, F \rangle$ → $u = 5$, $v = 7$

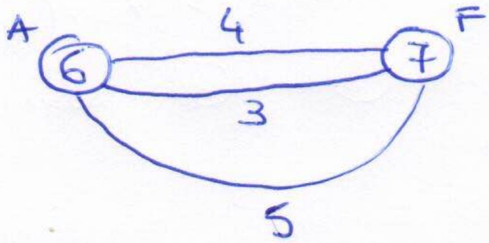
since current = 5 = u ⇒ we have to relink

5 — 7 ⇝ 6 — 7

Q.insert (6, 7, 5, C, F)

=

We draw the current graph and PQ

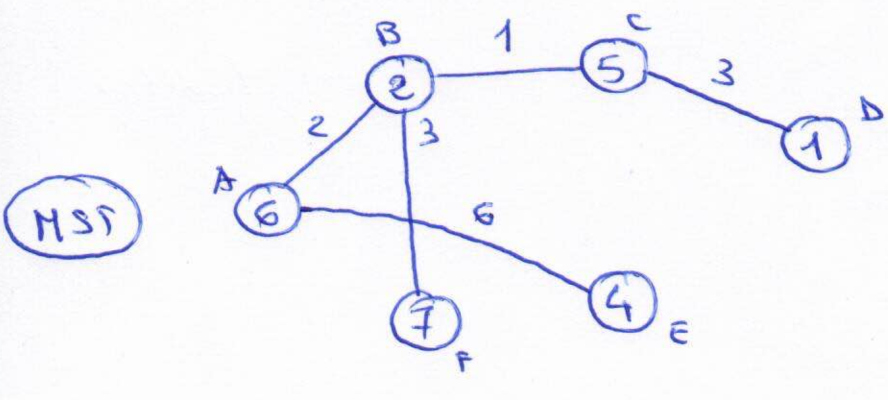


min	max	c	u	v
6	7	3	B	F
6	7	4	A	F
6	7	5	C	F

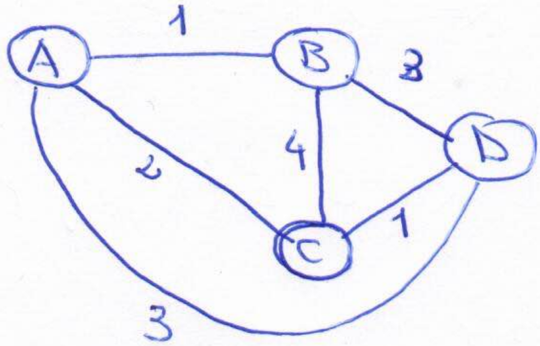
If M = 2 then we can solve MST in memory of this graph which hence output $\langle B, F \rangle$ which corresponds to

$\langle 6, 7, \underset{\uparrow}{3}, B, F \rangle$

min weight



Esercizio 2



current = ϕ

π = Identical Perm.

min π	max π	cost	u	v
A	B	1	A	B
A	C	2	A	C
A	D	3	A	D
B	D	3	B	D
B	C	4	B	C
C	D	1	C	D

min Q $\rightarrow$ < A, B, 1, A, B > $\rightarrow$ u = A, v = B

<u>since</u> current $\neq$ A = u $\Longrightarrow$ current = **A**, relinkTo = B

output < A, B >

extract min

min Q $\rightarrow$ < A, C, 2, A, C > $\rightarrow$ u = A, v = C

since current = A = u $\Longrightarrow$ Q.insert (B, C, 2, A, C)

[we have relinked A $\rightarrow$ B]

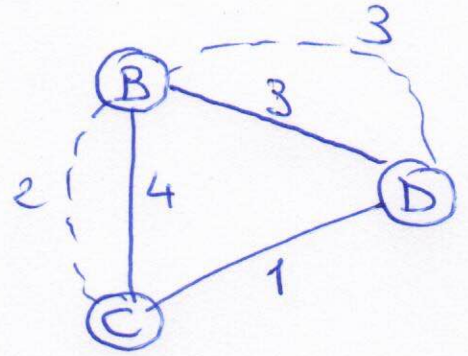
min Q → < A, D, 3, A, D > → u = A, v = D

<u>since</u> current = A = u ⇒ Q.insert (B, D, 3, A, D)

[we have relinked (A) ⟶ (B)]

═

The new graph is:



min π	max π	cost	u	v
B	C	2	A	C
B	C	4	B	C
B	D	3	B	D
B	D	3	A	D
C	D	1	C	D

goes here

min Q → < B, C, 2, A, C > ⇒ u = B, v = C

<u>since</u> current ≠ B = u ⟶ current = B, relinkTo = C

output < A, C >

extract min

═

This goes *

min Q → < B, C, 4, B, C > ⇒ u = B, v = C

<u>since</u> current = B = u ⇒ ~~insert (C, C, 4, B, C)~~

no op because

relinkTo = C = v.

min Q → < B, D, 3, B, D > ⇒ u = B, v = D

since current = B = u ⇒ Q.insert (C, D, 3, B, D)

=

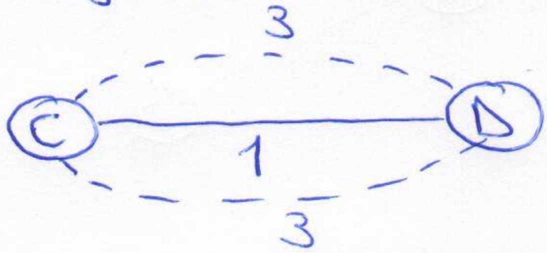
min Q → < B, D, 3, A, D > ⇒ u = B, v = D

since current = B = u ⇒ Q.insert (C, D, 3, A, D)

(*) → MOVED EDGE

=

The current graph and priority queue is:



min π	max π	cost	u	v
C	D	1	C	D
C	D	3	A	D
C	D	3	B	D

If M = 2 then we apply Kruskal and thus output < C, D, 1, C, D >