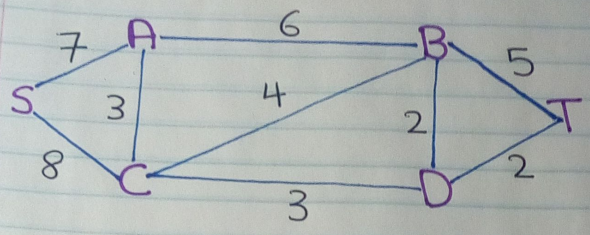
Example of Prim's Algorithm:

Consider the graph below.

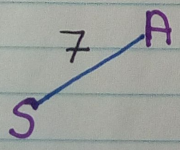


1. Start at vertex S.

Vertex	S	a	b	c	d	t
Priority	0	∞	∞	∞	∞	∞
Pred						

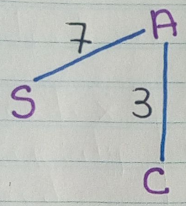
2. From S, we can go to either A or C but because the distance to A is smaller, we will go to A. Then, we remove S from the min-heap.

vertex	S	a	c	b	d	t
Priority	0	7	8	∞	∞	∞
Pred		S	S			



3. We will now visit A. From A, we update the priority of c to 3 because 3 is less than 8 and we update the priority of B to 6. Since the priority of C is less than the priority of B, we go to C. Then, we remove A from the min-heap.

vertex	~~$~~	~~A~~	C	B	D	t
Priority	~~0~~	~~7~~	3	6	∞	∞
Pred		~~$~~	A	A		

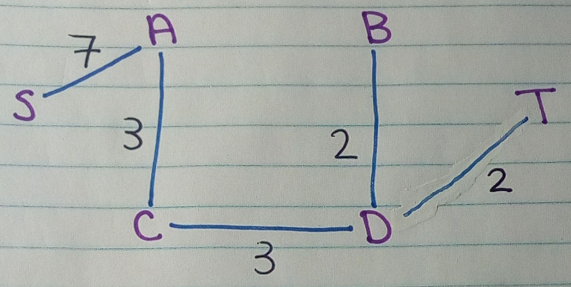


4. We will now visit C. From C, we update the priority of B to 4 because 4 is less than 6 and we update the priority of D to be 3. Then, we will go to D because it has the lowest priority. Finally we remove C from the min-heap.

vertex	~~$~~	~~A~~	~~C~~	D	B	t
Priority	~~0~~	~~7~~	~~3~~	3	4	∞
Pred		~~$~~	~~A~~	C	C	

5. We will now visit D. Then, we will update the priorities of B and T to 2. Then, I will go from D to B and then D to T. We are finished now.

vertex	~~S~~	~~A~~	~~C~~	~~D~~	~~B~~	~~T~~
Priority	~~0~~	~~7~~	~~3~~	~~3~~	~~2~~	~~2~~
Pred		~~S~~	~~A~~	~~C~~	~~D~~	~~D~~



Note: In summary, with Prim's alg, you choose any vertex and traverse the graph by taking the path with the least weight. Furthermore, if a vertex can have a smaller priority, you must update the min-heap to show the smallest possible priority it can have. Finally, we remove the vertex from the min-heap when we have visited it.