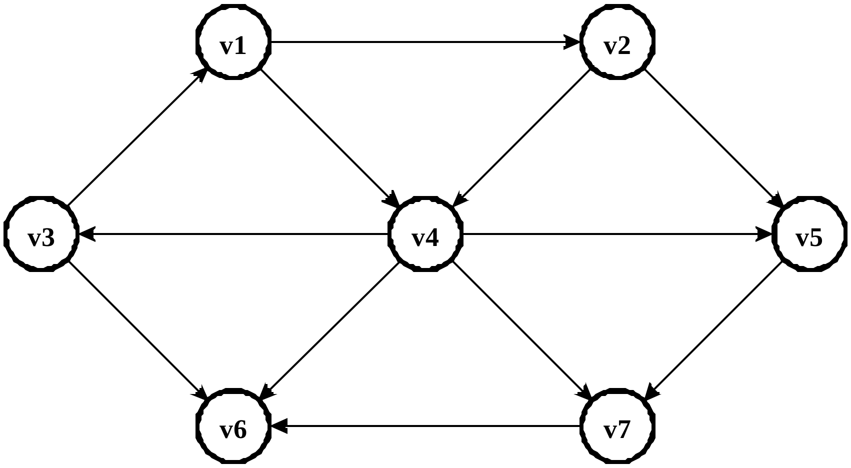

Figure 1. An unweighted directed graph G

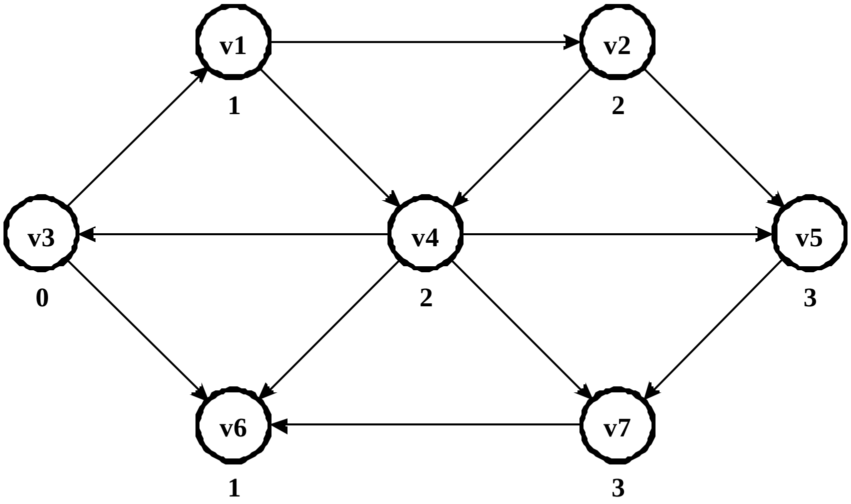

Figure 2. Final shortest paths from vertex $\boldsymbol{v_3}$ to every other vertex

	After initialization			v_3 dequeued			v_1 dequeued		
v	**known**	**dist**	**path**	**known**	**dist**	**path**	**known**	**dist**	**path**
v_1	no	BIG	-	no	1	v_3	yes	1	v_3
v_2	no	BIG	-	no	BIG	-	no	2	v_1
v_3	no	0	-	yes	0	0	yes	0	0
v_4	no	BIG	-	no	BIG	-	no	2	v_1
v_5	no	BIG	-	no	BIG	-	no	BIG	-
v_6	no	BIG	-	no	1	v_3	no	1	v_3
v_7	no	BIG	-	no	BIG	-	no	BIG	-
q	v_3			v_1 v_6			v_6 v_2 v_4		

	v_6 dequeued			v_2 dequeued			v_4 dequeued		
v	**known**	**dist**	**path**	**known**	**dist**	**path**	**known**	**dist**	**path**
v_1	yes	1	v_3	yes	1	v_3	yes	1	v_3
v_2	no	2	v_1	yes	2	v_1	yes	2	v_1
v_3	yes	0	0	yes	0	0	yes	0	0
v_4	no	2	v_1	no	2	v_1	yes	2	v_1
v_5	no	BIG	-	no	3	v_2	no	3	v_2
v_6	yes	1	v_3	yes	1	v_3	yes	1	v_3
v_7	no	BIG	-	no	BIG	-	no	3	v_4
q	v_2 v_4			v_4 v_5			v_5 v_7		

	v_5 dequeued			v_7 dequeued					
v	**known**	**dist**	**path**	**known**	**dist**	**path**			
v_1	yes	1	v_3	yes	1	v_3			
v_2	yes	2	v_1	yes	2	v_1			
v_3	yes	0	0	yes	0	0			
v_4	yes	2	v_1	yes	2	v_1			
v_5	yes	3	v_2	yes	3	v_2			
v_6	yes	1	v_3	yes	1	v_3			
v_7	no	3	v_4	yes	3	v_4			
q	v_7			**empty**					

Unweighted shortest path algorithm:

```
1.  Create queue $Q$, allocating as many entries in the queue as there are vertices in the graph.
2.  Initialize table $T$ making the input vertex $V$ the origin.
3.  Insert origin vertex $V_o$ into the queue.
4.  While queue $Q$ is not empty
    4.1.  Remove vertex $V$ from the queue.
    4.2.  Mark vertex $V$ as "known."
    4.3.  For each vertex $W$ adjacent to $V$ do
          4.3.1.  If the distance to vertex $W$ is undefined ($BIG$) then
                  4.3.1.1.  Add one to the distance to vertex $V$ and assign that value to vertex $W$ in table $T$.
                  4.3.1.2.  Assign vertex $V$ to the path to vertex $W$.
                  4.3.1.3.  Insert vertex $W$ into the queue.
5.  end while
```