4. Selected

1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	0	1	0	1	0
1	1	0	1	0	1	0
1	1	0	1	1	1	0
1	1	1	1	1	1	0
1	1	1	1	1	1	1

distance

0	∞	∞	∞	∞	∞	∞
0	7	∞	5	∞	∞	∞
0	7	∞	5	15	6	∞
0	7	∞	5	8	6	11
0	7	8	5	7	6	11
0	7	5	5	7	6	9
0	7	5	5	7	6	9

5.

단계	정점	found	distance
1	0	1 0 0 0 0 0	0 50 45 10 ∞ ∞
2	3	1 0 0 1 0 0	0 50 45 10 25 ∞
3	4	1 0 0 1 1 0	0 45 45 10 25 ∞
4	1	1 1 0 1 1 0	0 45 45 10 25 ∞
5	2	1 1 1 1 1 0	0 45 45 10 25 ∞
6	4	1 1 1 1 1 0	0 45 45 10 25 ∞

6.

0	50	45	10	*	*
*	0	10	15	*	*
*	*	0	*	30	*
20	*	*	0	15	*
*	20	35	*	0	*
*	*	*	*	3	0

A^{-1}

0	50	45	10	*	*
*	0	10	15	*	*
*	*	0	*	30	*
20	70	65	0	15	*
*	20	35	*	0	*
*	*	*	*	3	0

A^0

$$\begin{matrix} 0 & 50 & 45 & 10 & * & * \\ * & 0 & 10 & 15 & * & * \\ * & * & 0 & * & 30 & * \\ 20 & 70 & 65 & 0 & 15 & * \\ * & 20 & 30 & 35 & 0 & * \\ * & * & * & * & 3 & 0 \end{matrix} \quad A^1$$

$$\begin{matrix} 0 & 50 & 45 & 10 & 75 & * \\ * & 0 & 10 & 15 & 40 & * \\ * & * & 0 & * & 30 & * \\ 20 & 70 & 65 & 0 & 15 & * \\ * & 20 & 30 & 35 & 0 & * \\ * & * & * & * & 3 & 0 \end{matrix} \quad A^2$$

$$\begin{matrix} 0 & 50 & 45 & 10 & 25 & * \\ 35 & 0 & 10 & 15 & 30 & * \\ * & * & 0 & * & 30 & * \\ 20 & 70 & 65 & 0 & 15 & * \\ 55 & 20 & 30 & 35 & 0 & * \\ * & * & * & * & 3 & 0 \end{matrix} \quad A^3$$

$$\begin{matrix} 0 & 45 & 45 & 10 & 25 & * \\ 35 & 0 & 10 & 15 & 30 & * \\ 85 & 50 & 0 & 65 & 30 & * \\ 20 & 35 & 45 & 0 & 15 & * \\ 55 & 20 & 30 & 35 & 0 & * \\ 58 & 23 & 33 & 38 & 3 & 0 \end{matrix} \quad A^4$$

$$\begin{matrix} 0 & 45 & 45 & 10 & 25 & * \\ 35 & 0 & 10 & 15 & 30 & * \\ 85 & 50 & 0 & 65 & 30 & * \\ 20 & 35 & 45 & 0 & 15 & * \\ 55 & 20 & 30 & 35 & 0 & * \\ 58 & 23 & 33 & 38 & 3 & 0 \end{matrix} \quad A^5$$

7.

```
typedef struct GraphNode {
    int Vertex;
    struct GraphNode * link
} GraphNode

typedef struct GraphType {
    int n
    GraphNode * adj_list[MAX]
} GraphType
```

```
Void  Shortest_path (int Start , int n) {
    int i, v, w
    for (i =0   i<g→n   i++)
        distance[i] = weight[start][i]
        found [i] = FALSE
    }
    found[Start] = TRUE
    distance [Start] =0
    for (i=0   i< n-2   i++)  {
       v= choose(distance ,n , found )
       found[v] =TRUE
       for (w =0  w<n   w++)
           if  (!found[w])
               while (g→adj_list[v] →vertex !=w)
                   g→ adj_list[v] = g→adj_list[v]→link
               if (distance[v]+ g→ adj_list[v]   <distance[w])
                  distance[w] = distance[v] + g→adj_list [v]
```

8.

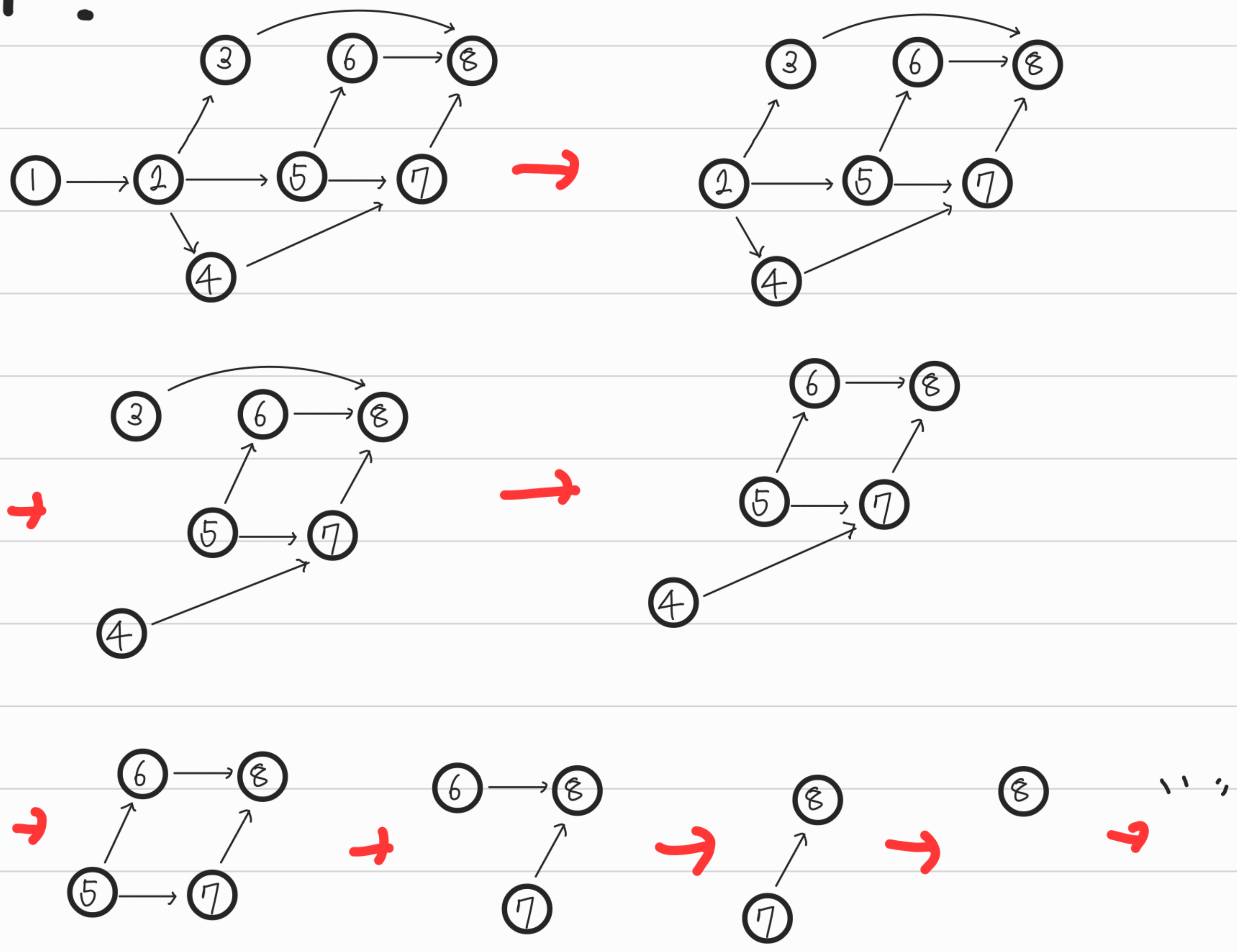
```
void  Shortest_path (int Start , int n)
    int i, v, w
    for ( i=0   i<n   i++)
        distance [i] = weight [Start][i]
        found[i] = FALSE
    found [start] = TRUE
    distance [start] = 0

    for ( i=0   i < n-2   i++)
        u = choose (distance , n, found )
        found [u] = TRUE
        printf ( u)
        for (w=0   w <n   w++)
            if ( !found [w] )
                if ( distance[u] + weigt[u][w] <distance[w] )
                    distance[w] = distance [u] + weight [u][w]
        }
    }
```

9. distance : 시작 정점으로부터 최단경로의 거리

0	50	45	10	*	*
0	50	45	10	25	*
0	45	45	10	25	*
0	45	45	10	25	*
0	45	45	10	25	*

10.



$1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8$