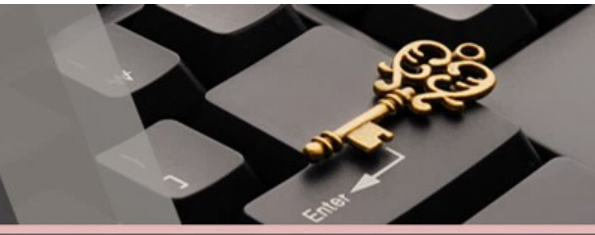
실전 알고리즘 0x13강
플로이드 알고리즘

BaaaaaaaaaaaaaaaarkingDog

목차



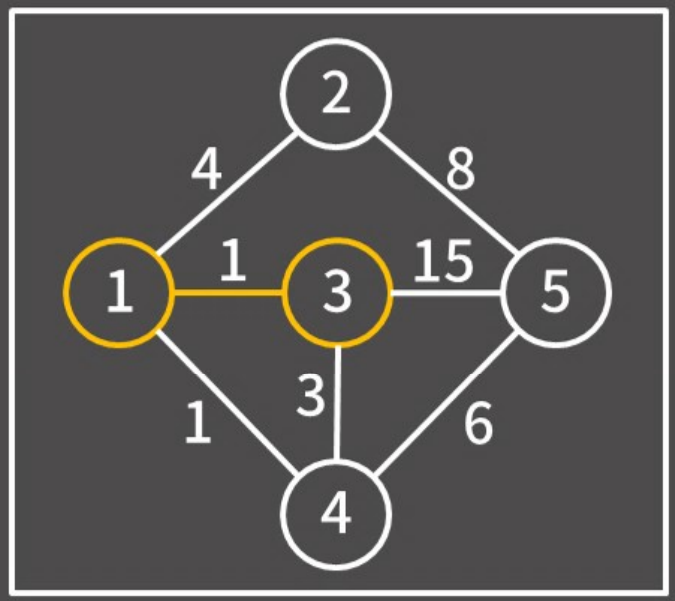
0x00 플로이드 알고리즘의 기능

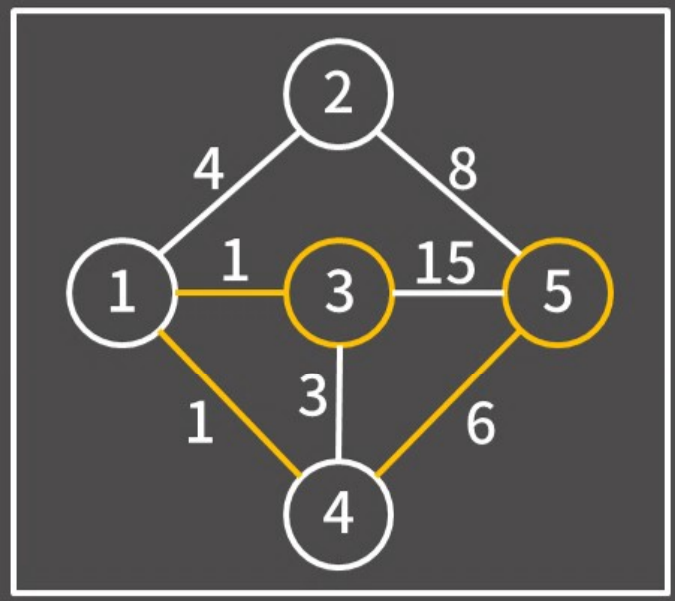
0x01 과정 살펴보기

0x02 구현 코드

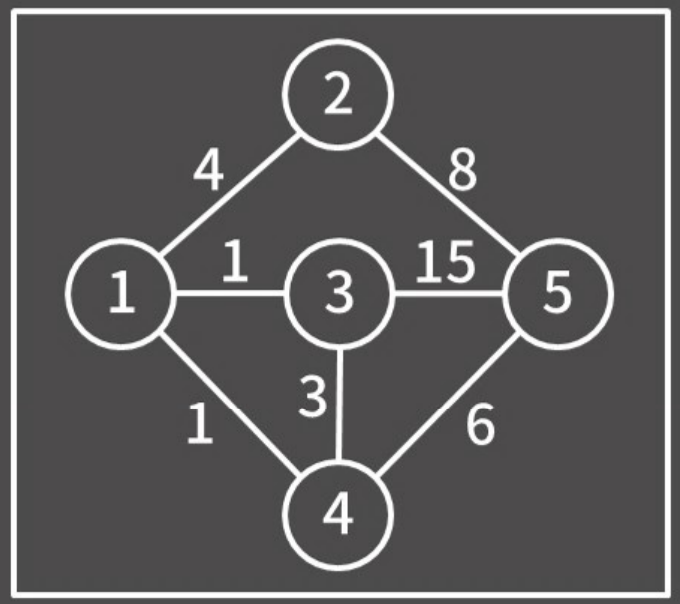
0x03 경로 복원 방법

0x00 플로이드 알고리즘의 기능





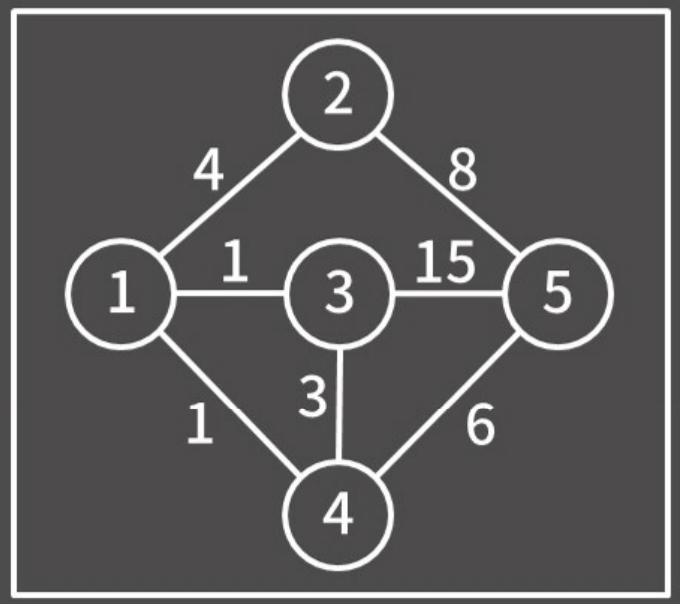
0x00 플로이드 알고리즘의 기능



	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

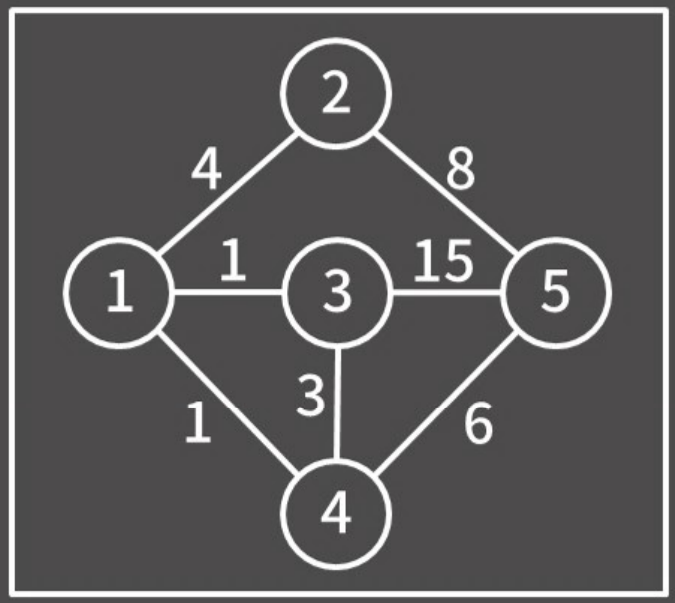
0x01 과정 살펴보기



	1	2	3	4	5
1					
2					
3					
4					
5					

최단 거리 테이블

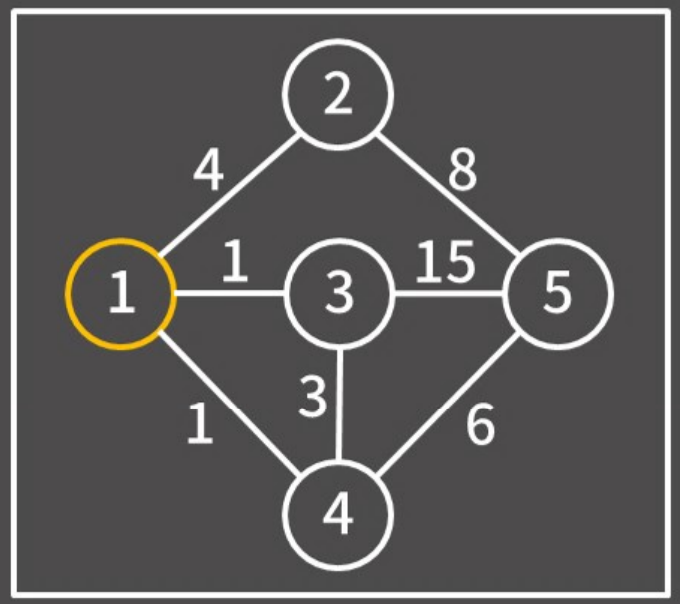
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	∞
2	4	0	∞	∞	8
3	1	∞	0	3	15
4	1	∞	3	0	6
5	∞	8	15	6	0

최단 거리 테이블

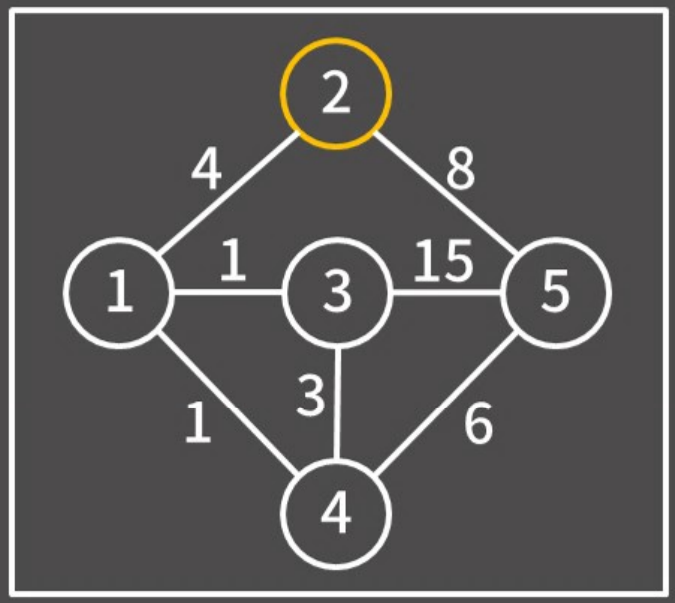
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	∞
2	4	0	5	5	8
3	1	5	0	2	15
4	1	5	2	0	6
5	∞	8	15	6	0

최단 거리 테이블

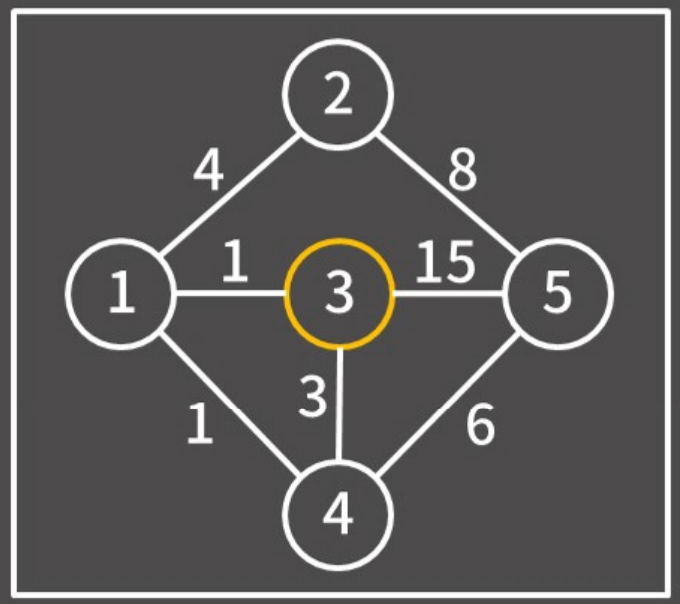
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	12
2	4	0	5	5	8
3	1	5	0	2	13
4	1	5	2	0	6
5	12	8	13	6	0

최단 거리 테이블

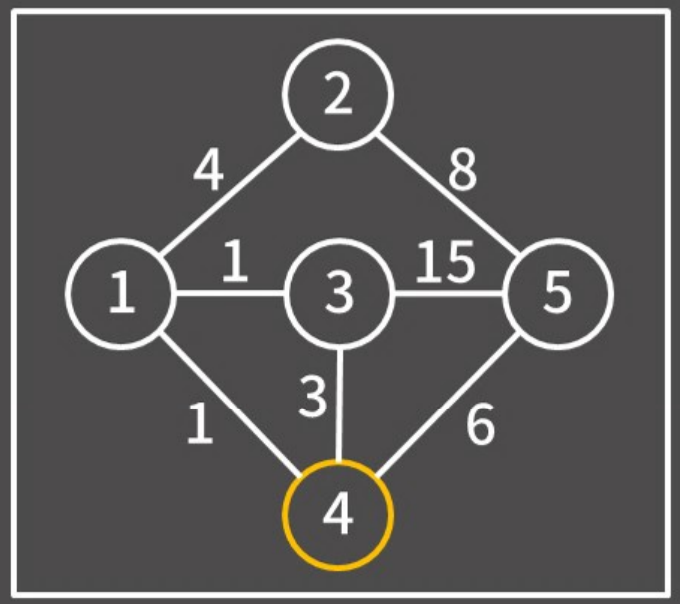
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	12
2	4	0	5	5	8
3	1	5	0	2	13
4	1	5	2	0	6
5	12	8	13	6	0

최단 거리 테이블

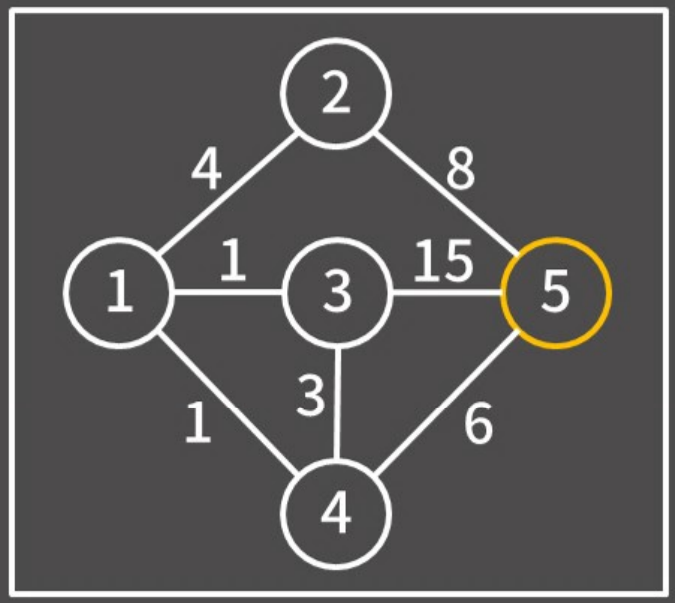
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

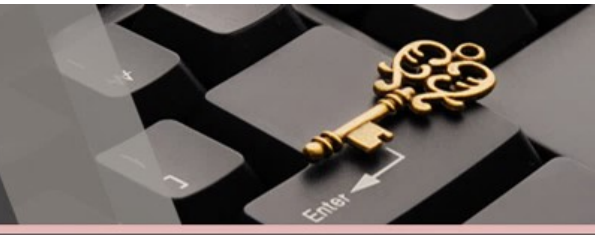
0x01 과정 살펴보기



	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

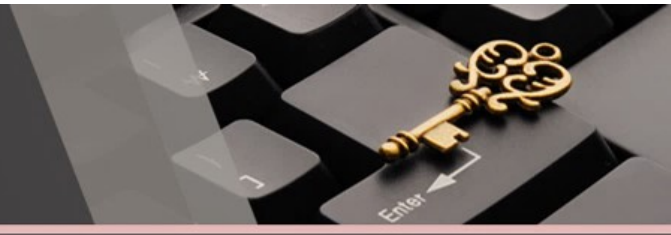
0x02 구현 코드



연습 문제 – BOJ 11404번: 플로이드

정답 코드 : http://boj.kr/8ebb89eaea7c45f79c71e7da897bbd59

0x02 구현 코드



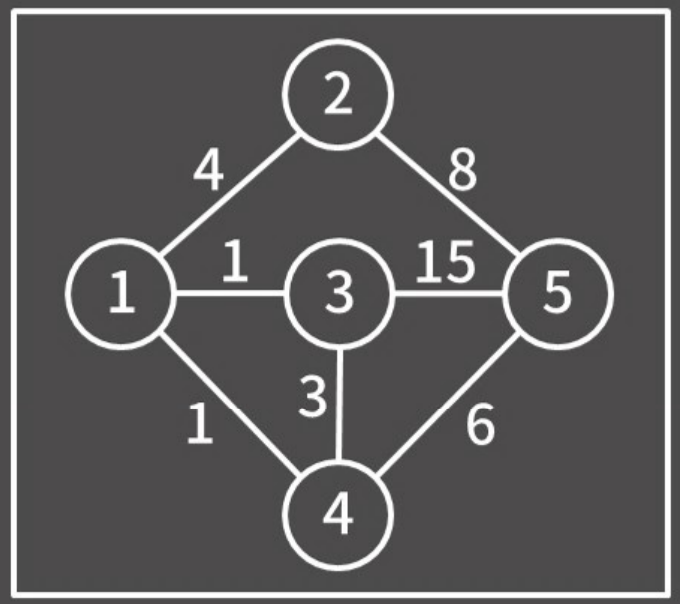
연습 문제 – BOJ 11404번: 플로이드

정답 코드 : http://boj.kr/8ebb89eaea7c45f79c71e7da897bbd59

```
#include<bits/stdc++.h>
using namespace std;
const int INF = 1e9+10;
int d[105][105];
int n,m;
int main()
{
  ios::sync_with_stdio(0);
  cin.tie(0);
  cin >> n >> m;
  for(int i = 1; i <= n; i++)
    fill(d[i], d[i]+1+n, INF);
  while(m--){
    int a,b,c;
    cin >> a >> b >> c;
    d[a][b] = min(d[a][b], c);
  }
  for(int i = 1; i <= n; i++) d[i][i] = 0;
  for(int k = 1; k <= n; k++)
    for(int i = 1; i <= n; i++)
      for(int j = 1; j <= n; j++)
        d[i][j] = min(d[i][j], d[i][k]+d[k][j]);

  for(int i = 1; i <= n; i++){
    for(int j = 1; j <= n; j++){
      if(d[i][j] == INF) cout << "0 ";
      else cout << d[i][j] << ' ';
    }
    cout << '\n';
  }
}
```

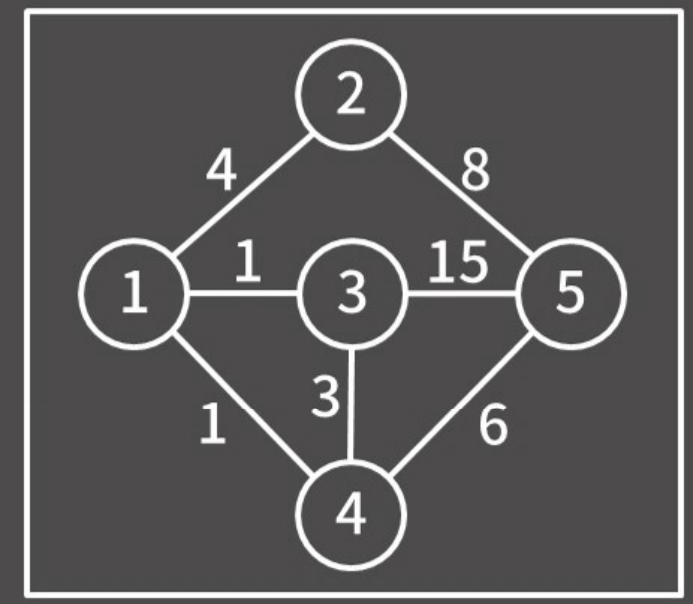
0x03 경로 복원 방법



	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

0x03 경로 복원 방법



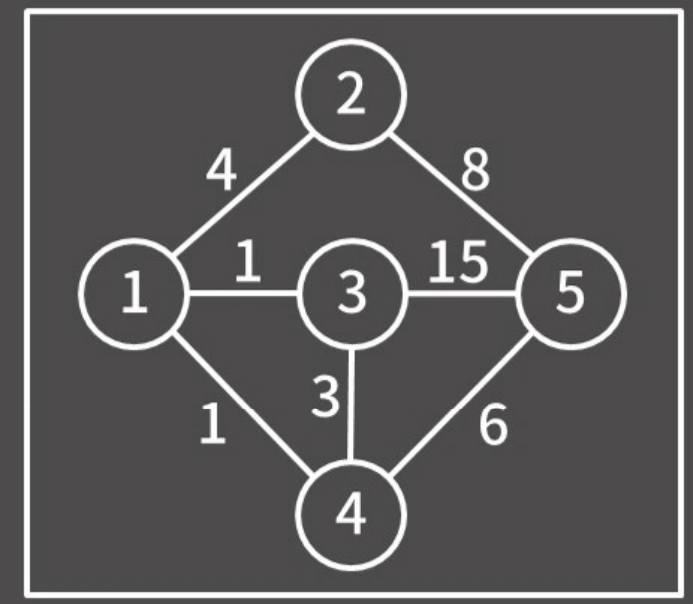
	1	2	3	4	5
1					
2					
3					
4					
5					

최단 거리 테이블

	1	2	3	4	5
1					
2					
3					
4					
5					

nxt 테이블

0x03 경로 복원 방법



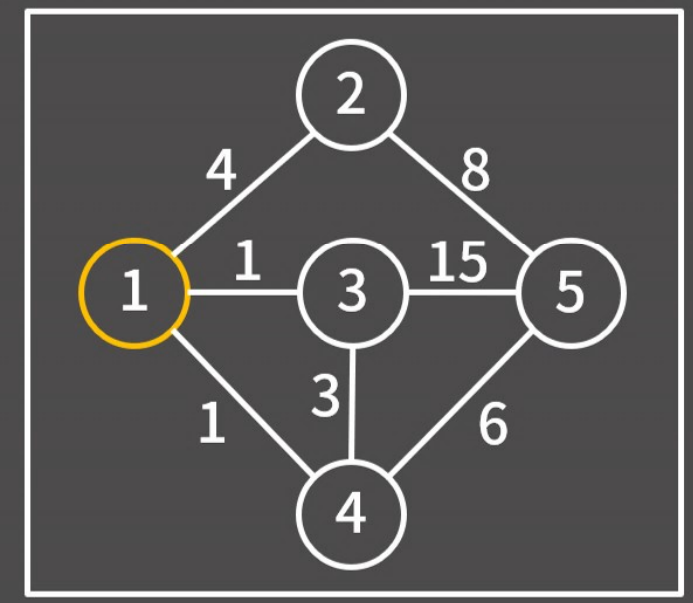
	1	2	3	4	5
1	0	4	1	1	∞
2	4	0	∞	∞	8
3	1	∞	0	3	15
4	1	∞	3	0	6
5	∞	8	15	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	
2	1				5
3	1			4	5
4	1		3		5
5		2	3	4	

nxt 테이블

0x03 경로 복원 방법



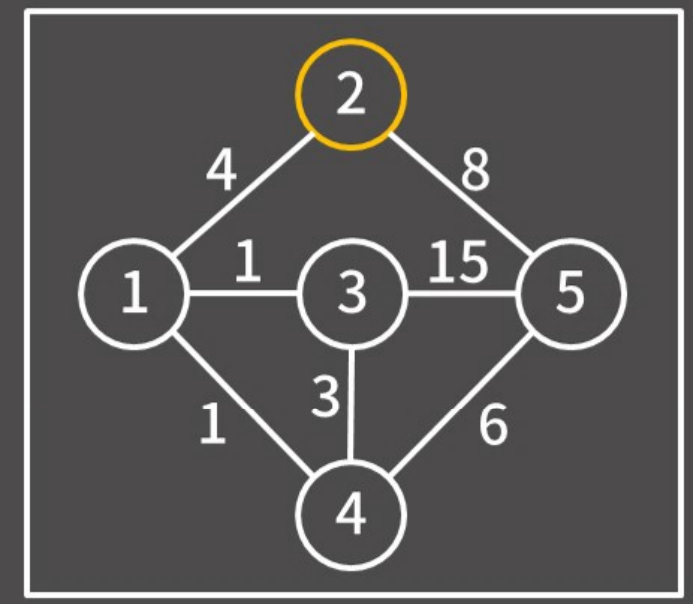
	1	2	3	4	5
1	0	4	1	1	∞
2	4	0	5	5	8
3	1	5	0	2	15
4	1	5	2	0	6
5	∞	8	15	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	
2	1		1	1	5
3	1	1		1	5
4	1	1	1		5
5		2	3	4	

nxt 테이블

0x03 경로 복원 방법



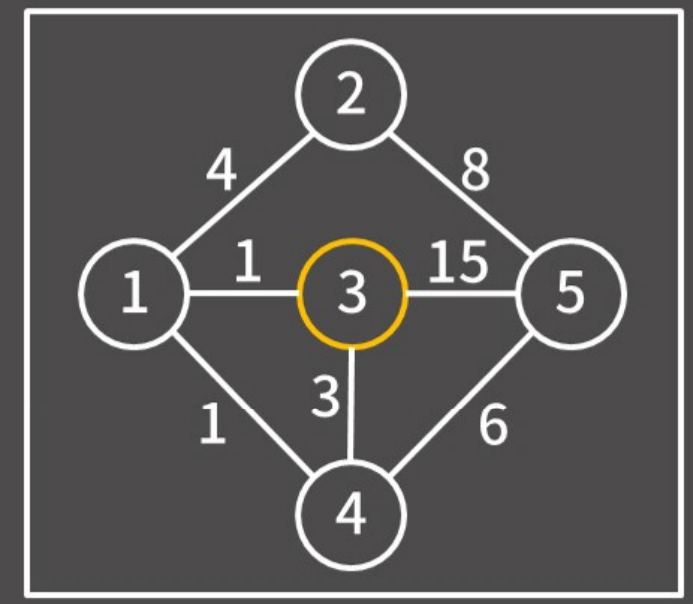
	1	2	3	4	5
1	0	4	1	1	12
2	4	0	5	5	8
3	1	5	0	2	13
4	1	5	2	0	6
5	12	8	13	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	2
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	2	2	2	4	

nxt 테이블

0x03 경로 복원 방법



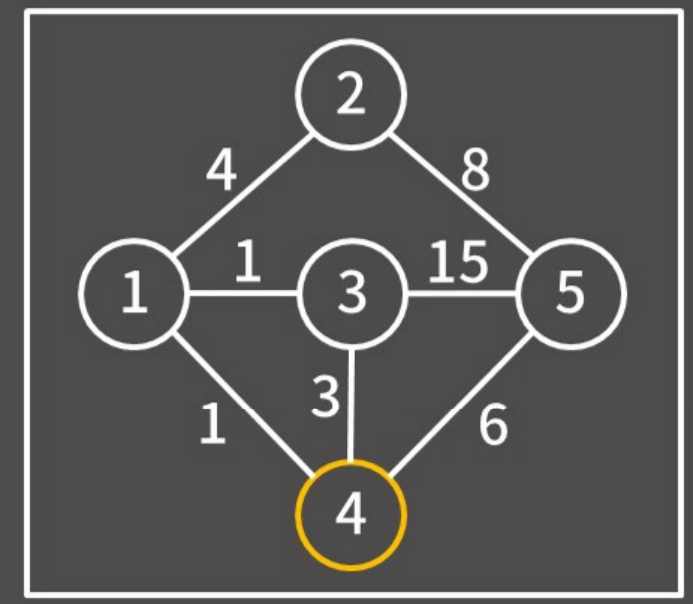
	1	2	3	4	5
1	0	4	1	1	12
2	4	0	5	5	8
3	1	5	0	2	13
4	1	5	2	0	6
5	12	8	13	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	2
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	2	2	2	4	

nxt 테이블

0x03 경로 복원 방법



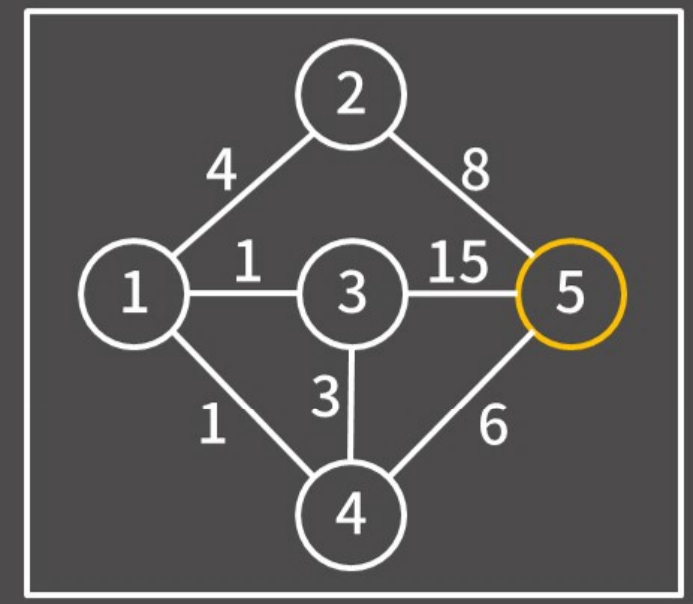
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

nxt 테이블

0x03 경로 복원 방법



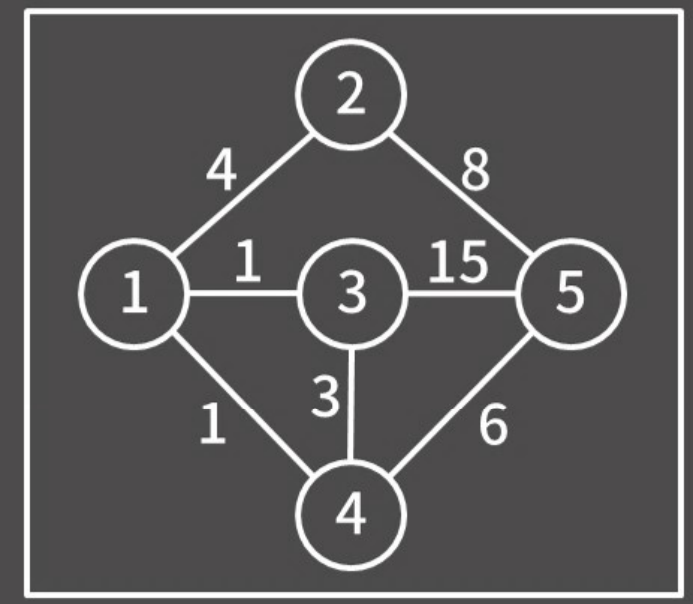
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

nxt 테이블

0x03 경로 복원 방법



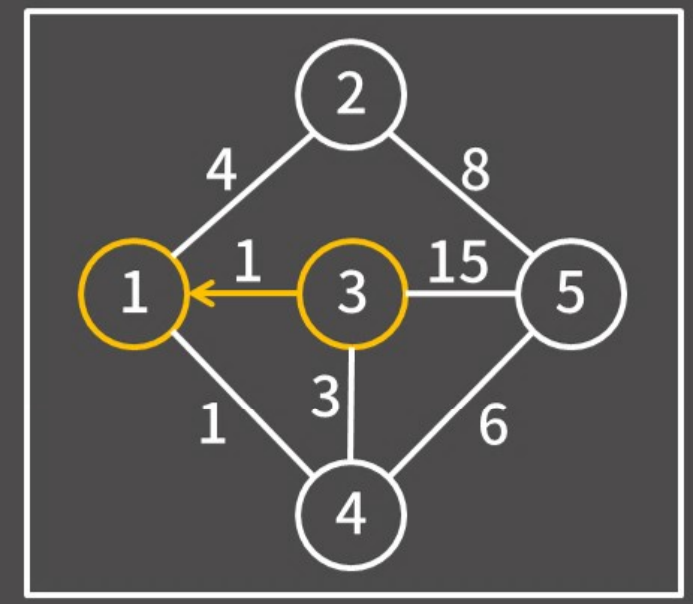
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

nxt 테이블

0x03 경로 복원 방법



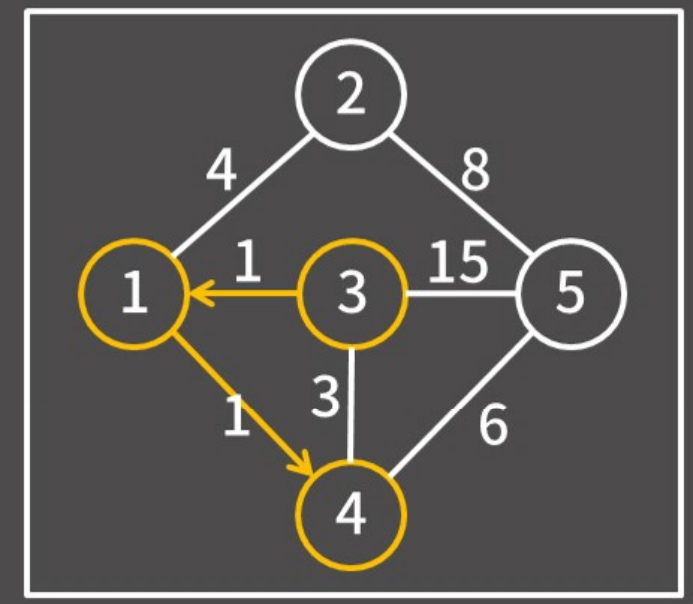
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

nxt 테이블

0x03 경로 복원 방법



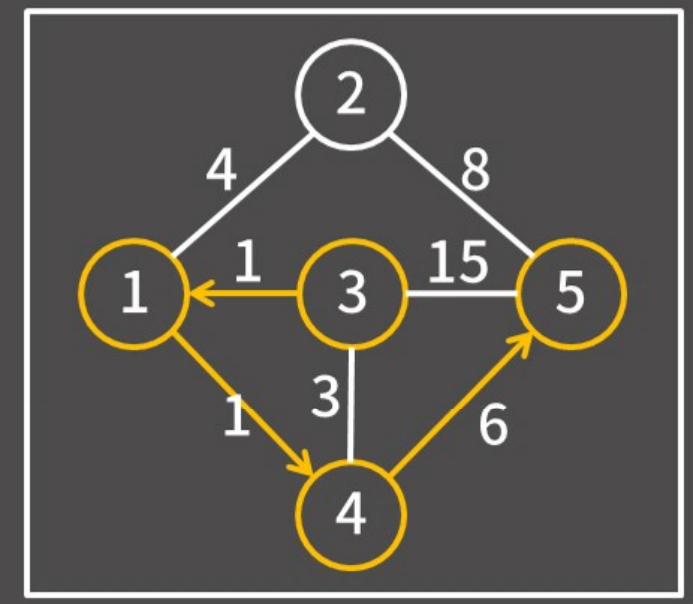
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

nxt 테이블

0x03 경로 복원 방법



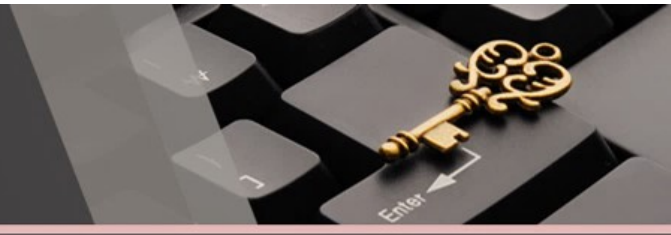
	1	2	3	4	5
1	0	4	1	1	7
2	4	0	5	5	8
3	1	5	0	2	8
4	1	5	2	0	6
5	7	8	8	6	0

최단 거리 테이블

	1	2	3	4	5
1		2	3	4	4
2	1		1	1	5
3	1	1		1	1
4	1	1	1		5
5	4	2	4	4	

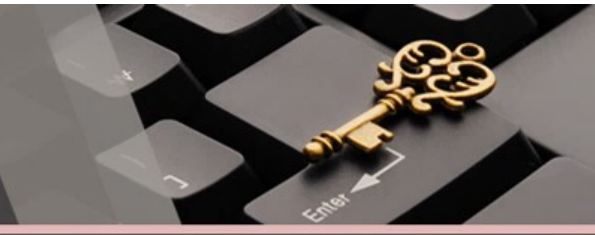
nxt 테이블

0x03 경로 복원 방법




```
01  vector<int> path;
02  int cur = s;
03  while(cur != t){
04    path.push_back(cur);
05    cur = nxt[cur][t];
06  }
07  path.push_back(cur);
```

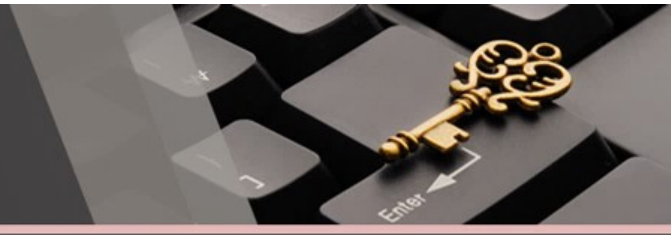
0x03 경로 복원 방법



연습 문제 – BOJ 11780번: 플로이드 2

정답 코드 : http://boj.kr/edcd051aa52a4d04a0abd4b2fe9e230c

0x03 경로 복원 방법



연습 문제 – BOJ 11780번: 플로이드 2

정답 코드 : http://boj.kr/3a5b494a79764b87bad72eda0e05f2df

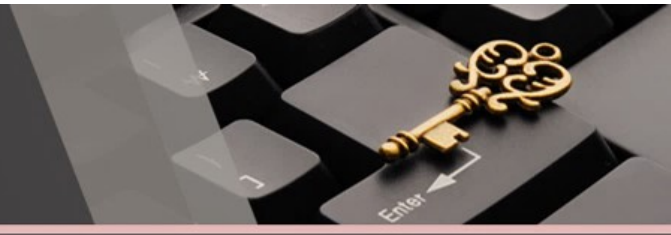
```
#include<bits/stdc++.h>
using namespace std;

const int INF = 1e9+10;
int d[105][105];
int nxt[105][105];
int n,m;
int main()
{
  ios::sync_with_stdio(0);
  cin.tie(0);
  cin >> n >> m;
  for(int i = 1; i <= n; i++)
    fill(d[i], d[i]+1+n, INF);
```

```
while(m--){
    int a,b,c;
    cin >> a >> b >> c;
    d[a][b] = min(d[a][b], c);
    nxt[a][b] = b;
  }
  for(int i = 1; i <= n; i++) d[i][i] = 0;

  for(int k = 1; k <= n; k++)
    for(int i = 1; i <= n; i++)
      for(int j = 1; j <= n; j++)
        if(d[i][j] > d[i][k]+d[k][j]){
          nxt[i][j] = nxt[i][k];
          d[i][j] = d[i][k]+d[k][j];
        }
```

0x03 경로 복원 방법



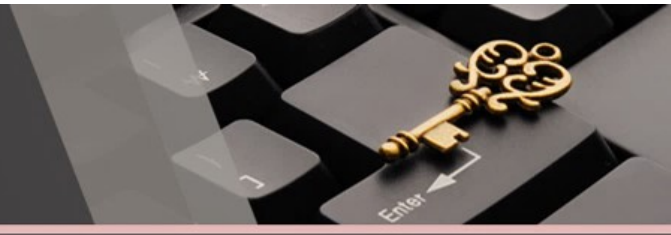
연습 문제 – BOJ 11780번: 플로이드 2

정답 코드 : http://boj.kr/edcd051aa52a4d04a0abd4b2fe9e230c

```
30    for(int i = 1; i <= n; i++){
31      for(int j = 1; j <= n; j++){
32        if(d[i][j] == INF) cout << "0 ";
33        else cout << d[i][j] << ' ';
34      }
35      cout << '\n';
36    }
37    for(int i = 1; i <= n; i++){
38      for(int j = 1; j <= n; j++){
39        if(d[i][j] == 0 or d[i][j] == INF){
40          cout << "0\n";
41          continue;
42        }
```

```
43        vector<int> path;
44        int st = i;
45        while(st != j){
46          path.push_back(st);
47          st = nxt[st][j];
48        }
49        path.push_back(j);
50        cout << path.size() << ' ';
51        for(auto x : path) cout << x << ' ';
52        cout << '\n';
53      }
54    }
55  }
```

강의 정리



- 플로이드 알고리즘을 익혔다.
- 구현 방법과 경로 복원 방법을 익혔다.