알고리즘 실습 보고서

-MCM&BF-

전공 :컴퓨터공학과

분반 : 05반

학번 :2017019888

이름 :김수빈

1. 실행 환경

본 실습은 Windows10 64bit, jdk 1.8.0_221, Eclipse EE가 설치된 환경에서 실행되었다

2. 과제 설명

출제된 과제에 대한 설명

- homework1

data11_matrix_chain.txt 파일로부터 각 행렬의 차원 값을 읽어, n개의 행렬을 곱할 때 최소 곱셈의 수를 구한다. 이때 최소 곱셈 값을 구하기 위해, 매단계 곱셈 수를 저장하고 이를 테이블 형태로 출력한다.

- homework2

data11_bellman_ford_1.txt와 data11_bellman_ford_2.txt 파일로부터 데이터를 읽어, 그래프를 그린다. 정점 0으로부터 각 정점까지의 최단 거리를 구하는 프로그램을 작성한다.
이때, 입력 데이터의 첫번째 라인은 정점, 두번째 라인부터는 간선의 정보를 포함한다.
음수 사이클이 존재할 경우 음수 사이클이 존재함을 출력한다.

3. 문제 해결 방법

- 문제를 해결하기 위해 자신이 사용한 방법, 아이디어에 대한 설명

#Homework1

먼저, 행렬의 dimension을 표현하기 위해, int형 값 m과 n을 Matrix 객체를 생성했다.

주어진 문제를 해결하기 위해 필요한 메소드는 Matrix의 리스트를 매개변수로 받아, 각 단계별 곱셈의 수를 구하고 이를 테이블에 저장하는 메소드 matrix_chain가 있다.

data11_matrix_chain.txt 파일을 읽어 ArrayList<Matrix>을 구성한다. ArrayList<Matrix>를 매개변수로 하여 matrix_chain 메소드를 호출한다.

matrix_chain 메소드에서는 int[][] M에 곱셈의 수를 저장하여 재사용함으로써 문제를 해결한다. M[i][j]는 I번째 행렬부터 j번째 행렬까지 곱할 때 필요한 곱셈 연산의 개수이다.

$i\,(0 \leqq i < n-l)$와 $j\,(j = i+l)\,(1 \leqq l < n)$에 다음을 반복한다.

$k(i \leqq k < j)$에 대해, $M[i][k]+M[k+1][j]+pqr\,(A_{ik} = p \times q, A_{(k+1)j} = q \times r)$가 가장 최소가 되는 값을 M[i][j]에 삽입한다. A_{ik}는 I번째 행렬부터 k번째 행렬까지 곱했을 때의 행렬이며 dimension 값이 p by q이고 $A_{(k+1)j}$는 k+1번째 행렬부터 j번째 행렬까지 곱했을 때의 행렬이며 dimension 값이 q by r이다.

M을 출력하면 결과와 같은 화면이 나온다.

Homework2

먼저, Graph을 표현하기 위해, Edge와 Vertex가 필요하며, Vertex는 int값을 갖는 숫자로, Edge는 시작 정점 u와 도착 정점 v, 이 사이간 거리 distance를 갖는 Edge객체를 생성했다.
Graph는 Vertex의 리스트인 ArrayList<Integer>와, Vertex의 리스트에서 각 Vertex가 어느 인덱스에 있는지 알아내기 위한 HashMap<Integer, Integer>가 필요하다 Vertex의 값에 따른 nodeList 내의 인덱스를 알 수 있다. 또 그래프 전체의 Edge를 가지고 있는 ArrayList<Edge>를 가지고 있다.

주어진 문제를 해결하기 위해 필요한 메소드는 정점 0에서부터 다른 정점들까지의 최소 거리를 구해주며 negative cycle의 존재여부를 알려주는 메소드인 findShortestPath와 정점 0에서 Edge를 추가했을 때 정점 0에서 Edge의 도착 정점까지 거리를 구해주는 RELAX, 정점 0에서부터 각 정점까지의 거리값을 가지고 있는 배열 D를 초기화 해주는 메소드,Init_Single_Source가 있다.

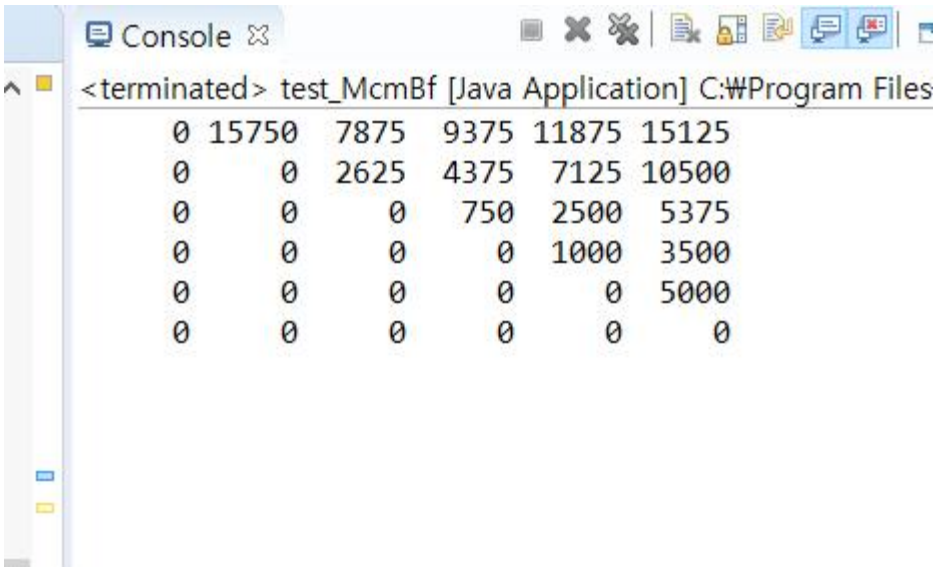
data11_bellman_ford_1.txt 파일을 읽어 Graph를 구성한다. Graph와 시작 정점의 인덱스 s를 매개변수로 하여 findShortestPath메소드를 호출한다.
아래를 정점의 개수만큼 반복한다.
문제 조건에는 정점의 개수-1만큼 반복한 후, 마지막으로 반복했을 때, 값이 Update 되는 경우 음수 사이클이 존재한다고 판단한다. 출력 예시 또한 정점의 개수만큼 값을 출력하므로, RELAX를 호출하는 횟수는 정점의 개수이며, 마지막 반복에서 값이 변경되는 경우, 음수 사이클 존재 여부 확인을 위한 flag 값 returnValue를 false로 한다.
그래프 내 Edge를 모두 읽어, 각 Edge에 대해 시작 정점 0부터 Edge의 도착 정점까지 거리 newValue와 기존의 값 preValue를 비교하여 newValue가 더 작을 시 이를 업데이트하고 아니면 기존의 값을 유지한다.
반복이 끝나면 D의 값들을 출력하고 returnValue가 false인 경우 음수 사이클이 존재함을 출력한다.

RELAX 함수는 매개변수로 전달된 Edge에 대해 시작 정점 0부터 Edge의 도착 정점까지 거리를 시작 정점 0부터 Edge의 시작 정점까지 거리에 Edge의 distance를 더해 구한다.

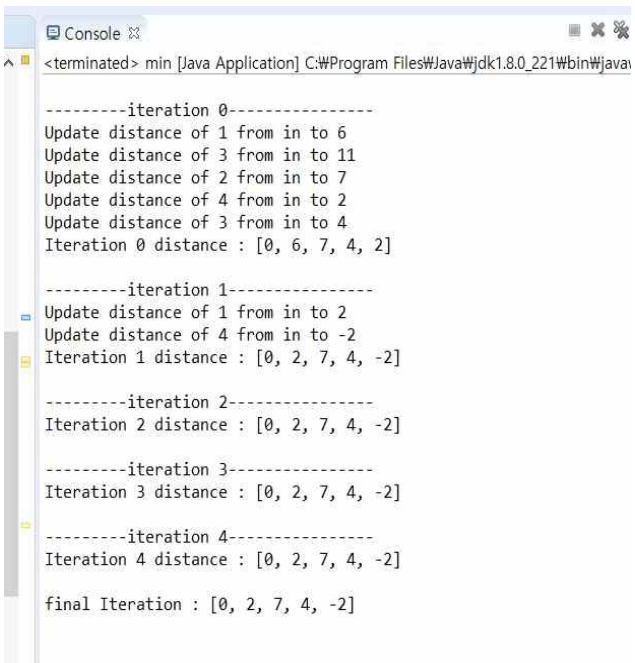
Init_Single_Source 메소드는 배열 D를 생성하고 시작 정점 s에 대해 D[s]를 0으로 놓아 s까지의 거리를 0으로 초기화하며 나머지 인덱스에 대해서는 Integer.MAXVALUE로 초기화한다.

결과 화면

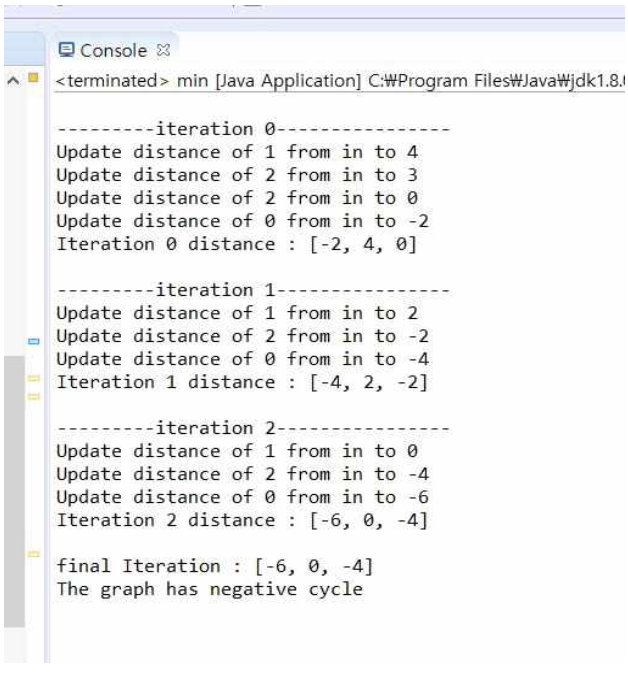
homework 1



homework 2



data11_bellman_ford_1.txt 결과



data11_bellman_ford_2.txt 결과

5. 느낀점 및 고찰

homework2는 이론에서 배웠던 방법은 모든 정점에서 목적지 t까지의 거리를 구하는 문제였으나, 실습은 정점 0에서 모든 정점까지의 거리를 구하는 문제라, 조금 달라 어려웠으나, 그렇게 크게 어렵지는 않았습니다. 하지만 pseudo code를 이해하는 부분은 오래걸렸ㄴ습니다.