



(이산수학) 프로젝트 과제

(제출일: 2025년 12월 12일 오후 23시 59분)

목적

1. 그래프의 이해 및 응용
2. 그래프 탐색 알고리즘 구현
3. 최단 거리 알고리즘 구현

1. 프로그램: 그래프 탐색 및 최단거리 알고리즘 구현

프로그램은 입력 파일에 정의된 그래프 정보를 바탕으로, (1) 그래프 탐색 및 (2) 최단 거리 구하기의 두 가지 기능을 수행한다. 자세한 구현 사항은 다음과 같다.

- (1) 하나의 프로그램으로 구현하며, 위의 두 가지 기능을 순차적으로 수행한 후, 종료한다.
- (2) 그래프는 인접 행렬로 (인접 리스트 허용하지 않음) 구현한다.
- (3) **input1.txt**에 정의된 그래프 정보를 바탕으로 그래프 상의 모든 정점을 방문할 수 있는 두 가지 알고리즘인 깊이 우선 탐색 및 너비 우선 탐색 알고리즘을 구현한다. input1.txt 파일에는 여러 개의 그래프 정보가 포함될 수 있으며, 파일에 있는 각 그래프에 대해서, **시작점 (정점 1)**에서 **깊이 우선 탐색(DFS)**과 **너비 우선 탐색(BFS)**을 수행한 결과를 출력한다. 출력 형식은 문제에 주어진 출력형식을 따른다.
- (4) **input2.txt**에 정의된 그래프 정보를 바탕으로 Dijkstra 알고리즘을 이용하여 최단 거리를 구한다. input2.txt 파일에는 여러 개의 그래프 정보가 포함될 수 있으며, 파일에 있는 각 그래프에 대해서, **시작점 (정점 1)**에서 **모든 다른 정점에 이르는 최단 경로 및 경로의 길이를** 출력한다. 출력 형식은 문제에 주어진 출력형식을 따른다.

2. 입력 파일 생성: 입력 파일의 이름은 반드시 **input1.txt**와 **input2.txt**로 한다.

- (1) 입력 그래프는 **단순 연결 그래프 (simple connected graph)**만 고려한다.
- (2) 입력 파일에서 인접 노드는 **방향성을 갖는다**. 따라서, 무방향 그래프를 표현하기 위해서는, 각 정점 별로 인접한 정점을 중복해서 모두 명시하여야 한다.
- (3) 그래프 당 정점의 최대 개수는 **100**이다.
- (4) **input1.txt**에 작성되는 그래프 정보는 다음과 같다.
 - 그래프의 **정점 (또는 노드)**는 아래의 예와 같이 **1에서부터 1씩 증가하는 정수 값**으로 한다.
 - 그래프 정보는 다음에 주어진 예제 (Ex 1과 Ex 2)와 같이, 먼저 **그래프를 구성하는 전체 정점 개수**가 주어지고, 다음으로 **각 정점 별로**, 해당 정점에 **인접한 노드 정보들을 한 줄에 빈칸으로 구분하여 나열**한다. 예를 들어, EX 1)의 경우, 입력 파일은 하나의 그래프를 포함하고 있고, 총 8개의 정점으로 구성되어 있으며, 정점 1은 정점 2, 3과 인접해 있고, 정점 2는 정점 1, 4, 5와 인접하고 있음을 나타낸다.
 - 인접 정점은 오름차순으로 나열한다.
 - 입력 파일은 위에서 제시한 입력 파일 형식을 반드시 따르되, 다양한 형태의 그래프를 포함할 수 있도록 자유롭게 생성하여 사용한다.

Ex 1) 한 개의 그래프만 포함하는 경우	Ex 2) 두 개 이상의 그래프를 포함하는 경우
8 1 2 3 2 1 4 5 3 1 6 7 4 2 8 5 2 8 6 3 8 7 3 8 8 4 5 6 7	3 1 2 3 2 1 3 3 1 2 2 1 2 2 1 4 1 2 3 2 4 3 2 4

- (5) input2.txt에 작성되는 그래프 정보는 다음과 같다.
- 그래프의 정점 (또는 노드)는 아래의 예와 같이 1에서부터 1씩 증가하는 정수 값으로 한다.
 - 정점간 거리는 0보다 큰 정수 값으로 한다.
 - 입력 파일 형식은 다음에 주어진 예제 (Ex 3과 Ex 4)와 같이, 그래프 별로, 그래프를 구성하는 전체 노드 개수와 각 노드 별 인접 노드와 인접 노드까지의 가중치 값들을 한 줄에 빈칸으로 구분하여 나열한다 (정점 - 인접 정점 - 가중치 - 인접 정점 - 가중치 순으로 나열되어 있음). Ex 3)의 경우, 입력 파일은 하나의 그래프를 포함하고 있고, 총 5개의 정점으로 구성되어 있으며, 정점 1은 정점 2, 3, 4, 5와 인접해 있고, 정점 2까지의 가중치는 7, 3까지의 가중치는 4, 4까지의 가중치는 6, 5까지의 가중치는 1임을 나타낸다.
 - 인접 정점은 오름차순으로 나열한다.
 - 입력 파일은 위에서 제시한 입력 파일 형식을 반드시 따르되, 다양한 형태의 그래프를 포함할 수 있도록 자유롭게 생성한다.

Ex 3) 한 개의 그래프만 포함하는 경우	Ex 4) 두 개 이상의 그래프를 포함하는 경우
5 1 2 7 3 4 4 6 5 1 2 3 2 2 4 5 4 2 3 5 4 1	5 1 2 10 4 30 5 100 2 3 50 3 5 10 4 3 20 5 60 5 6 1 2 20 4 30 2 3 20 5 40 3 6 20 4 3 20 5 20 6 30 5 6 10 6

3. 출력 형식: 프로그램의 출력은 반드시 다음의 출력 형식을 따른다.

- 프로그램은 input1.txt에 작성된 각 그래프에 대해서, **깊이 우선 탐색 수행 결과 및 너비 우선 탐색 수행 결과를 순차적으로 출력한다.**
- 그래프 탐방이 끝나면, input2.txt에 작성된 각 그래프에 대해서, 다른 모든 정점에 이르는 **최단 경로 및 최단 경로의 길이를 출력한다.**
- **경로의 길이가 같을 때는, 먼저 찾은 경로를 출력한다.** (즉, 알고리즘 구현 시, **경로의 길이가 작을 경우 (<)에만 갱신되도록 구현한다**)

출력 예 1) input1.txt의 Ex 1) 과 input2.txt의 Ex 3)이 입력으로 주어진 경우

1. 그래프 탐방 수행 결과

그래프 [1]

깊이 우선 탐색

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

너비 우선 탐색

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

=====

2. 최단 경로 구하기 수행 결과

그래프 [1]

시작점: 1

정점 [2]: 1 - 5 - 4 - 2, 길이: 5

정점 [3]: 1 - 3, 길이: 4

정점 [4]: 1 - 5 - 4, 길이: 2

정점 [5]: 1 - 5, 길이: 1

출력 예 1) input1.txt의 Ex 2)와 input2.txt의 Ex 4)가 입력으로 주어진 경우

1. 그래프 탐방 수행 결과

그래프 [1]

깊이 우선 탐색

1 - 2 - 3

너비 우선 탐색

1 - 2 - 3

=====

그래프 [2]

깊이 우선 탐색
1 - 2
너비 우선 탐색
1 - 2
=====

그래프 [3]

깊이 우선 탐색
1 - 2 - 4 - 3
너비 우선 탐색
1 - 2 - 3 - 4
=====

2. 최단 경로 구하기 수행 결과

그래프 [1]

시작점: 1
정점 [2]: 1 - 2, 길이: 10
정점 [3]: 1 - 4 - 3, 길이: 50
정점 [4]: 1 - 4, 길이: 30
정점 [5]: 1 - 4 - 3 - 5, 길이: 60
=====

그래프 [2]

시작점: 1
정점 [2]: 1 - 2, 길이: 20
정점 [3]: 1 - 2 - 3, 길이: 40
정점 [4]: 1 - 4, 길이: 30
정점 [5]: 1 - 4 - 5, 길이: 50
정점 [6]: 1 - 4 - 6, 길이: 60
=====

4. 프로그래밍 언어 및 실행 환경: C 또는 Java 로 구현한다.

5. 제출 사항

- (1) 소스 파일
- (2) 최종 보고서
 - 보고서 내용: 다음의 내용을 포함하여 상세하게 기술
 - ▶ 문제 정의
 - ▶ 시스템 환경 및 사용 언어
 - ▶ 그래프 표현 기법
 - ▶ 주요 변수 및 함수에 대한 설명
 - ▶ 다양한 그래프 형태에 대한 수행 결과 및 분석 (5 가지 이상)
 - ▶ 토의 사항
 - ▶ 기여도: 각 팀원의 역할을 상세하게 기술하고, 각 팀원의 참여율을 백분율(%)로 표시한다.
- (3) 제출 방법: 소스 파일 및 보고서를 (팀명)TP.zip으로 압축하여 팀 대표가 eCampus 팀 프로젝트 과제함에 제출

※ 구현 결과 평가 시에는 교수자가 생성한 입력 파일 input1.txt 및 input2.txt로 평가합니다. 입력 파일명 오류 등으로 프로그램이 정상 실행되지 않는 경우, 구현 점수가 0점 처리되니, 반드시 문제에 주어진 입력 파일명을 사용합니다.