Graph-Pattern-Matching-Challenge

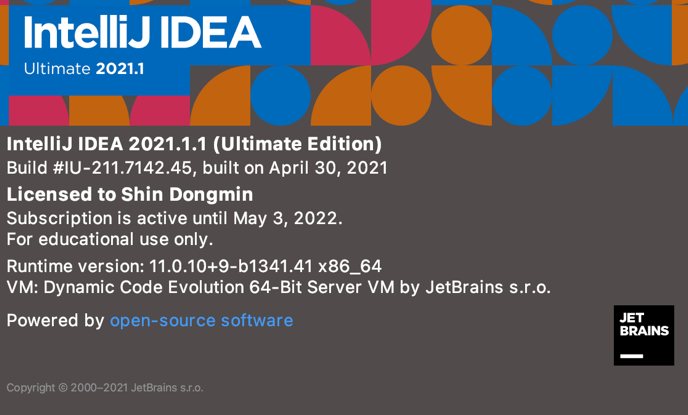
* Lec. Algorithm -

Team Supernova

2018-14637 임은성

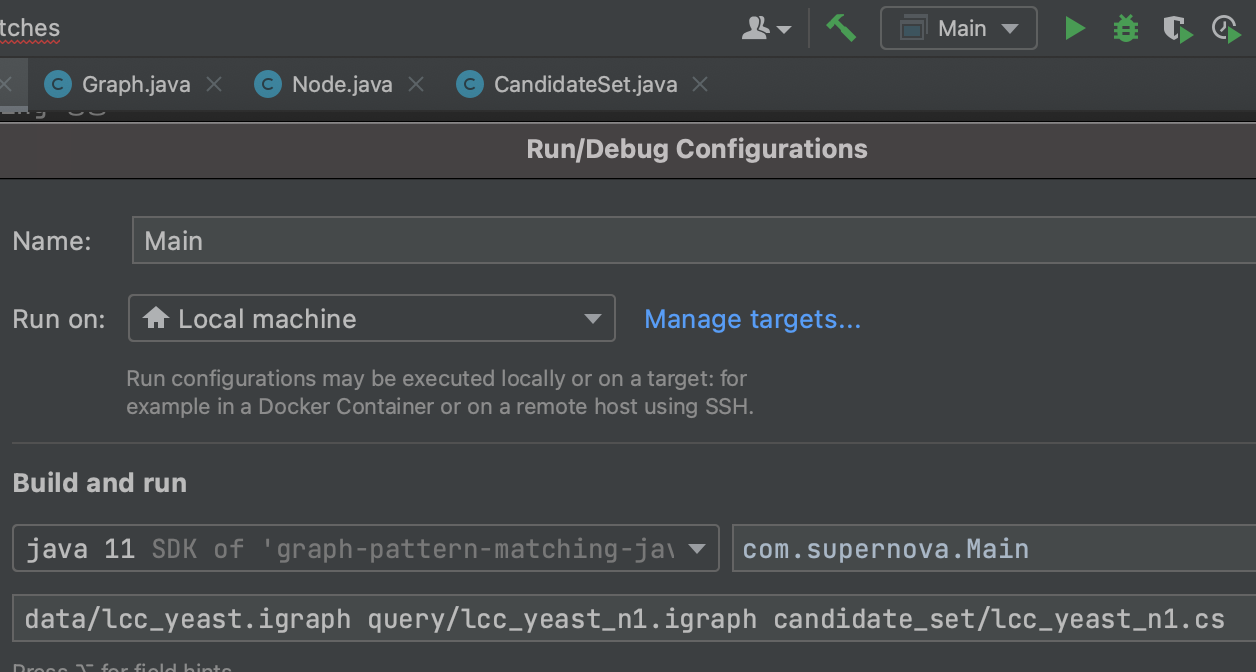
2018-16169 신동민

1. **과제환경 및 프로그램 실행방법**
   1. 과제환경 – Java 11 SDK



* 1. 프로그램 실행방법

아래와 같이 run/configurations에 들어가 argument로 주어지는 파일 경로를 작성해주고 오른쪽위에 있는 화살표모양의 run 버튼을 눌러줍니다.

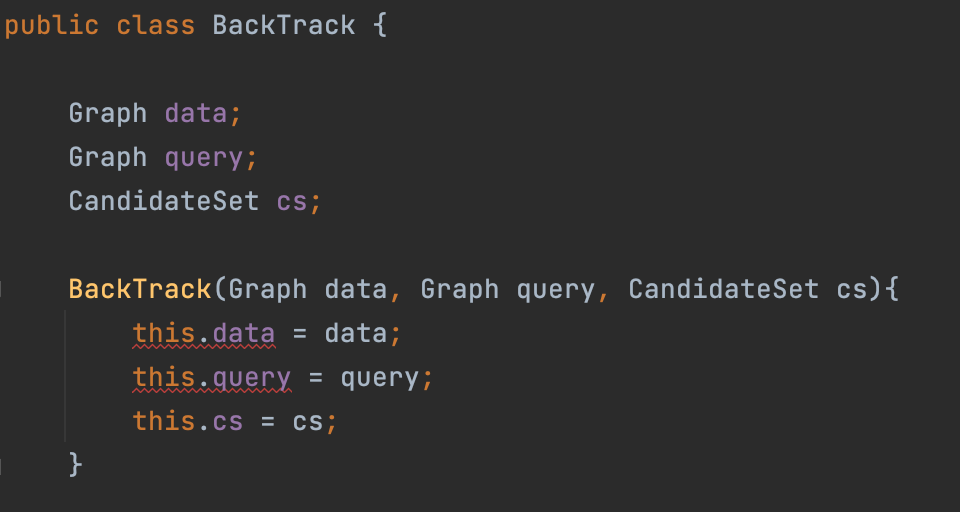


1. Initialize input data
   1. Graph – data & query graph
   2. CandidateSet – Candidate set
2. Implement BackTracking

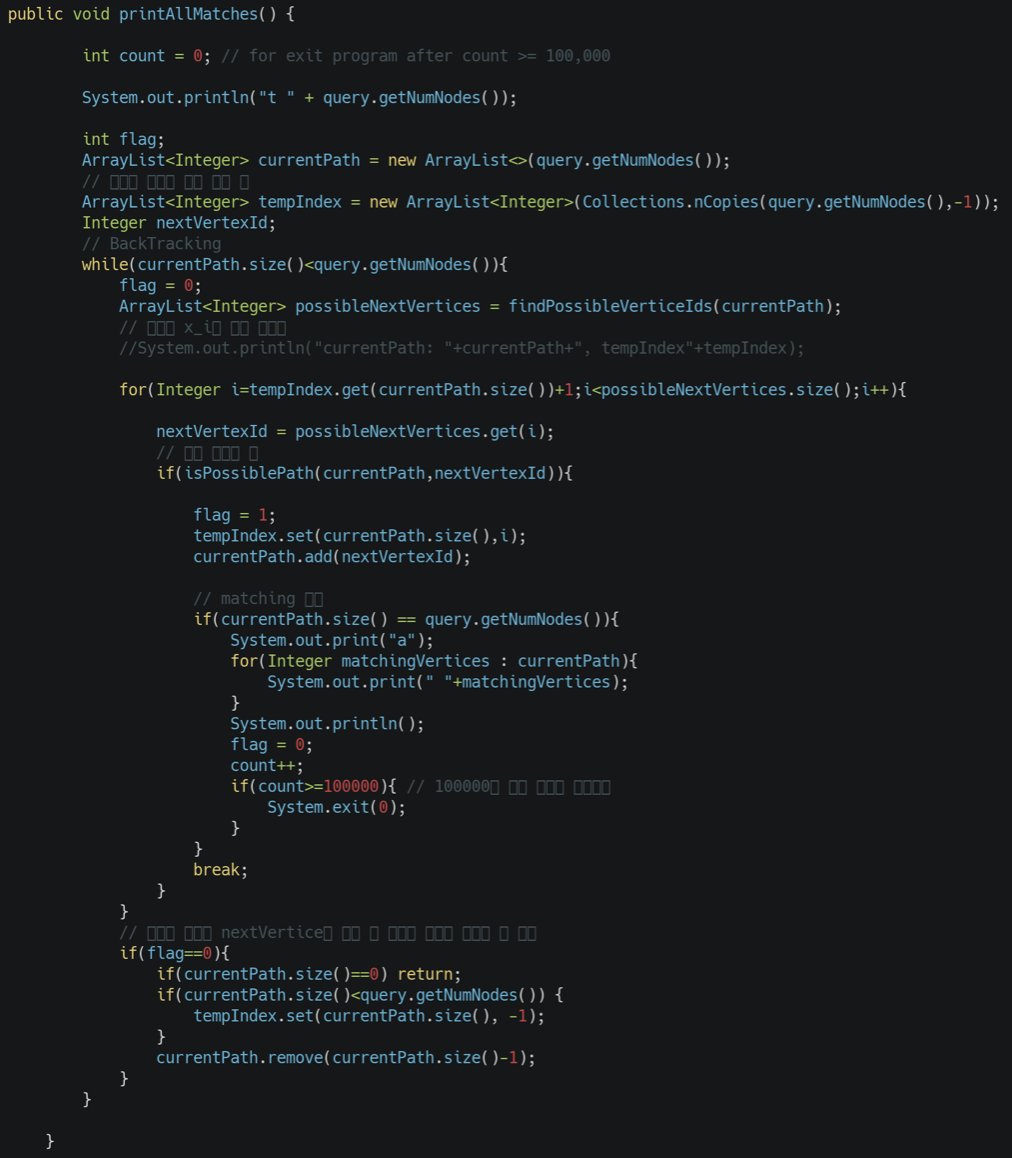
전체적인 구현방법은 다음과 같습니다. Query graph로부터 id가 0부터 끝까지 시행을 합니다. currentPath에 query id에 대응하는 data vertex id를 넣어주면서 currentPath가 모든 query id에 해당하는 vertex를 mapping했을 때 결과를 출력합니다.

currentPath가 새로운 vertex를 받을 때는 이 vertex에 해당하는 query에 대하여 이전 query id들에 연결되는 것이 있는지 확인하고, 있다면 해당하는 query id를 mapping하고 있는 data vertex가 currentPath안에서 존재여부를 확인합니다. 만약 모든 경우에 대해서 존재한다면, 이 새로운 vertex는 현재의 currentPath에 포함시켜도 되는 상황이기 때문에 포함시키고, 어떠한 query edge에 대해 해당하는 vertex가 currentPath안에 없을 때는 invalid한 경우이기 때문에 패스합니다.

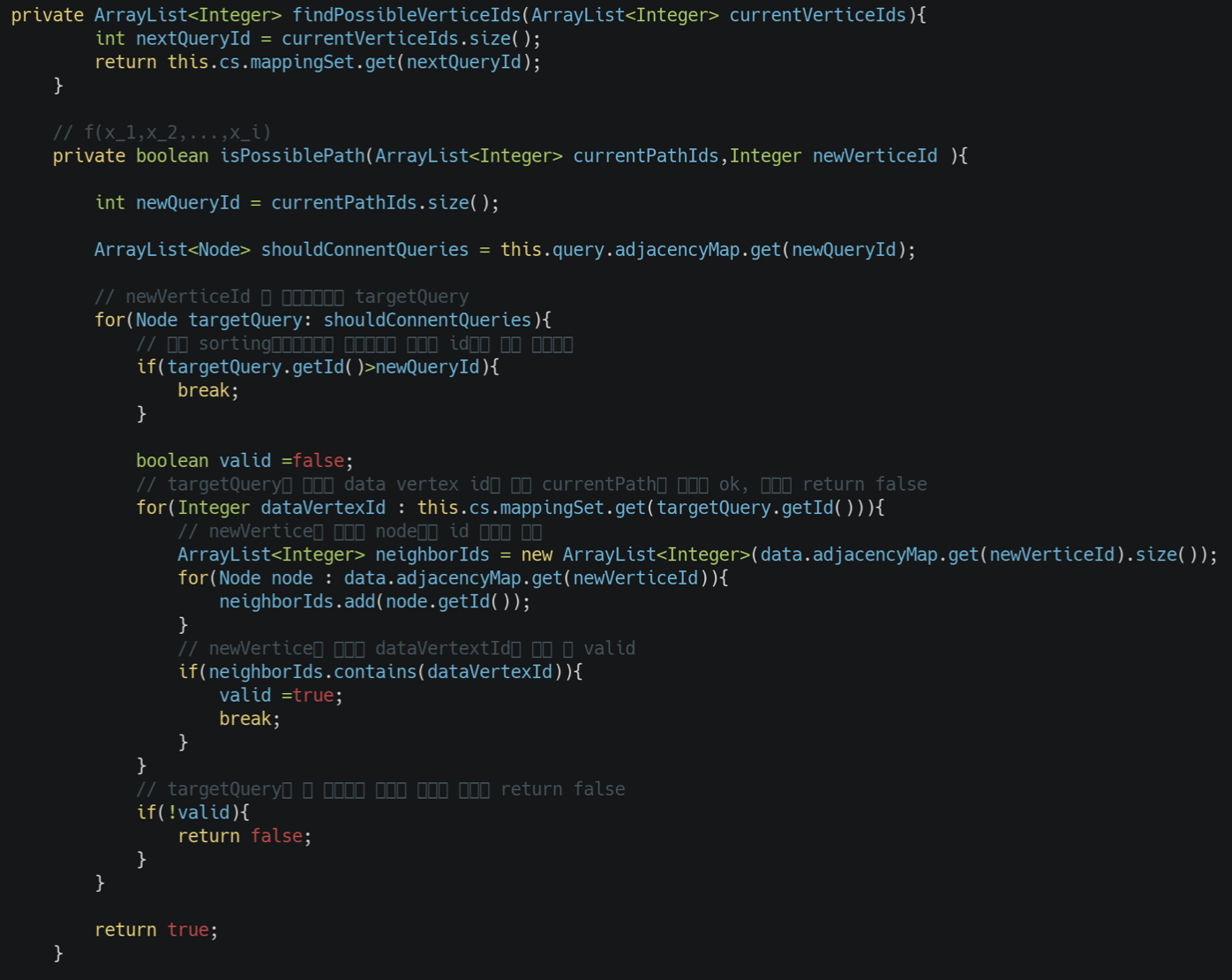
아래에서부터 자세한 코드설명을 하겠습니다.



위와 같이 각 input data를 구조화한 data graph, query graph, candidateSet을 클래스 변수로 정의합니다. 그리고 생성자에서 주어진 값을 모두 받아 클래스를 정의내립니다.



BackTracking을 구현한 핵심파트입니다. currentPath에는 tracking을 하면서 valid한 set를 담고있는 경로이고, data vertex id의 배열입니다. tempIndex는 currentPath에 넣을 때 candidate set에서 query id에 해당하는 data vertex id의 배열 중의 index를 넣는 자리입니다. BackTracking을 할 때 어느부분에서 다시 시작하면 되는지 알려주는 지표입니다. BackTracking은 근본적으로 DFS와 같이 시행됩니다. currentPath다음에 들어올 vertex는 currentPath 다음 query id에 해당하는 data vertex를 candidate set에서 고르게 됩니다. 이 때 들어올 수 있는 vertex를 findPossibleVerticeIds 함수를 통해 구하고, 각 vertex들에 대해 isPossiblePath함수를 통해서 currentPath에 추가시켜도 되는지에 대한 여부를 확인합니다. 그리고 추가시킬 수 있는 vertex가 없으면 flag를 조회함으로써 currentPath에서 없애줍니다. DFS와 같이 될 수 있는 path를 뒤에서부터 바꾸어가며 찾습니다. 아래는 위에서 사용한 findPossibleVerticeIds 함수와 isPossiblePath 함수에 대한 설명입니다.

.

findPossibleVerticeIds 함수는 currentPath의 vertex를 인자로 받아서 다음 query id에 매핑이 되는 vertex set을 배열로 return 합니다.

isPossiblePath는 새로 들어올 newVerticeId에 대하여 기존에 있던 currentPath의 vertex와 비교하면서 query graph에 해당하는 edge가 mapping된 형태로 있는지에 대한 여부를 확인합니다. 만약 모두 있으면 true를 반환하고, 하나라도 연결되는 edge가 없으면 false를 반환함으로써 새로 들어올 vertex가 valid한지 invalid한지 여부를 결정할 수 있습니다.