과제 #04

데이터사이언스를 위한 컴퓨팅 1 (2022년도 1학기, M3239.005500)

Due: 2022년 05월 05일(월) 23시 59분

1 Graph Representation

1.1 graph.cpp 파일의 구현

```
#include <iostream>
#include "graph.h"

void Vertex::AddToAdjacentList(Vertex* vertex) {
  adjacency_list_.push_back(vertex);
}

Graph::~Graph() {
  for (auto vertex : vertices_)
    delete vertex;
  vertices_.clear();
}

Vertex* Graph::GenVertex(std::string name) {
  Vertex* vertex = new Vertex(name);
  vertices_.push_back(vertex);
  return vertex;
}

void Graph::GenEdge(Vertex* start, Vertex* end) {
  start->AddToAdjacentList(end);
}
```

2 Breadth-First Search (BFS)

2.1 search.cpp 파일의 구현

```
void BFS(Vertex* start, std::map<Vertex*, unsigned int> &distance) {
  Vertex* vertex;
  std::queue<std::pair<Vertex*, unsigned int>> q;
  std::map<Vertex*, bool> visited;
  unsigned int curr;

  curr = 0;
  visited[start] = true;
  q.push(std::make_pair(start, curr));
```

```
  while (q.empty() == false) {
    vertex = q.front().first;
    curr = q.front().second;
    q.pop();

    for (auto adjacent : vertex->GetAdjacencyList()) {
      if (visited[adjacent] == false) {
        visited[adjacent] = true;
        q.push(std::make_pair(adjacent, curr + 1));
      }
    }

    distance[vertex] = curr;
  }
}
```

3 Depth-First Search (DFS)

3.1 search.cpp 파일의 구현

```
void DFS(Vertex* vertex, unsigned int &timestamp,
    std::map<Vertex*, unsigned int> &discovery_time,
    std::map<Vertex*, unsigned int> &finishing_time) {
  timestamp += 1;
  discovery_time[vertex] = timestamp;

  for (auto adjacent : vertex->GetAdjacencyList()) {
    if (discovery_time.find(adjacent) == discovery_time.end()) {
      DFS(adjacent, timestamp, discovery_time, finishing_time);
    }
  }

  timestamp += 1;
  finishing_time[vertex] = timestamp;
}
```