01 데이터 소개

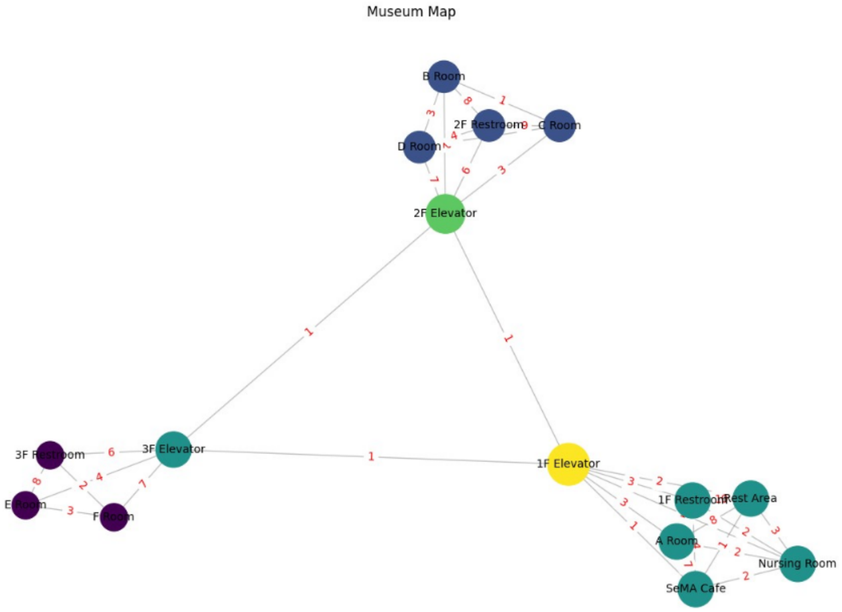
AI융합학부 20211367 오지윤

서울시립미술관 서소문본관 방문하여 사진 데이터 수집 및 해당 홈페이지에서 미술관 층별 도면 확보

미술관 층별 도면

각 층별 전시관마다 전시된 미술작품 데이터 수집

데이터 기반으로 재구성한 사회적 약자들을 위한 미술관 Map newMuseumMap.py




```
graph = {
    'A Room': {'Rest Area': 8, 'SeMA Cafe': 7, 'Nursing Room': 2, '1F Restroom': 4, '1F Elevator': 3},
    'Rest Area': {'A Room': 8, 'SeMA Cafe': 1, 'Nursing Room': 3, '1F Restroom': 10, '1F Elevator': 2},
    'SeMA Cafe': {'A Room': 7, 'Rest Area': 1, 'Nursing Room': 2, '1F Restroom': 4, '1F Elevator': 1},
    'Nursing Room': {'A Room': 2, 'Rest Area': 3, 'SeMA Cafe': 2, '1F Restroom': 2, '1F Elevator': 1},
    '1F Restroom': {'A Room': 4, 'Rest Area': 10, 'SeMA Cafe': 4, 'Nursing Room': 2, '1F Elevator': 3},
    '1F Elevator': {'A Room': 3, 'Rest Area': 2, 'SeMA Cafe': 1, 'Nursing Room': 1, '1F Restroom': 3,'2F Elevator': 1},

    '2F Elevator': {'2F Restroom': 6, 'B Room': 2, 'C Room': 3, 'D Room': 7,'1F Elevator': 1,'3F Elevator': 1},
    '2F Restroom': {'2F Elevator': 6, 'B Room': 8, 'C Room': 9, 'D Room': 4},
    'B Room': {'2F Elevator': 2, '2F Restroom': 8, 'C Room': 1, 'D Room': 3},
    'C Room': {'2F Elevator': 3, '2F Restroom': 9, 'B Room': 1, 'D Room': 2},
    'D Room': {'2F Elevator': 7, '2F Restroom': 4, 'B Room': 3, 'C Room': 2},

    '3F Elevator': {'3F Restroom': 6, 'E Room': 4, 'F Room': 7,'2F Elevator': 1, '1F Elevator': 1},
    '3F Restroom': {'3F Elevator': 6, 'E Room': 8, 'F Room': 2},
    'E Room': {'3F Elevator': 4, '3F Restroom': 8, 'F Room': 3},
    'F Room': {'3F Elevator': 7, '3F Restroom': 2, 'E Room': 3},
}
```

```
# 각 전시실의 관람시간 (작품 당 시간 1만큼 걸린다고 설정)
viewing_time = {
    'A Room': 20,
    'B Room': 27,
    'C Room': 17,
    'D Room': 8,
    'E Room': 6,
    'F Room': 13,
}
```

- 사회적 약자들의 실제 관람 경로만 고려한 Map
- 층별 전시실 및 엘리베이터, 층별 화장실, 놀이방& 수유실, SeMA Cafe, Rest Area만 고려
- 거리는 일정 비율로 줄여서 간선의 가중치로 설정

1층 전시실 : A Room으로 설정 (20작품 전시)
2층 천경자 전시실 : B Room (27개 작품)
2층 가나아트 컬렉션 전시실 : C Room
- 방문 당시 7080 도시현실 전시 중 (17개 작품)
2층 전시실 : D Room (8개 작품)
3층 전시실 왼쪽 : E Room (6개 작품)
3층 전시실 오른쪽 : F Room (13개 작품)

02 문제 소개

1. 유모차 및 휠체어 이용방문객을 위한 전시실 관람 최단 경로 탐색

museum.py

- 계단이 아닌 엘리베이터를 무조건 이용해야 한다는 조건 고려하여 알고리즘 작성
- 관람을 시작하고 싶은 전시실과 마지막으로 관람하고 싶은 전시실을 방문객에게 입력받아 최단 경로 탐색
- 최단경로 탐색을 위해 다익스트라 알고리즘 사용 -> 우선순위 큐 최소 힙을 사용하여 최단 거리와 최단 경로 탐색
- 최단 경로 출력을 위해 역추적을 통해 최단 경로 생성
- 완전 탐색을 통해 모든 가능한 순서의 방문 경로를 생성해 가장 최적의 경로를 탐색 -> 다익스트라만으로는 부족
- def find_shortest_path : 외판원 문제 TSP 알고리즘 + 다익스트라 알고리즘을 결합하여 문제 해결

- 모든 전시실의 순서를 고려하여 경로 생성 (파이썬 순열 함수 사용)
- 다익스트라 알고리즘을 내부적으로 사용하여 엘리베이터를 이용하는 부분과 전시실로 이동하는 부분의 최단 경로를 계산

2. 편의시설 경로 탐색에 추가하여 미술관 관람 최단 경로 탐색

museumSolution.py

- museum.py import 하여 편의시설 추가 고려하여 관람 최단 경로 탐색하는 알고리즘
- 관람 시작할 전시실, 마칠 전시실, 방문할 편의시설을 사용자가 직접 입력
- 이용객에게 관람을 시작할 전시실, 끝마칠 전시실, 방문하고 싶은 편의시설을 입력받아 기존 최단 경로에서 수유실, SeMA Cafe등 추가로 이용하고 싶은 편의시설을 고려한 최단 경로를 탐색
- find_shortest_path에 편의시설 경로만 추가한 코드 작성

```python
#museum.py에서 선택한 편의시설 고려하여 최단 경로 탐색
def find_shortest_path(graph, exhibition_rooms, start_point, end_point, selected_facilities):
    all_nodes = list(graph.keys())
    #편의시설 추가 탐색하는 코드
    facilities = [facility.strip() for facility in selected_facilities.split(",")]

    for facility in facilities:
        if facility not in all_nodes:
            return None, None

    all_stops = exhibition_rooms + facilities
    shortest_distance = float('infinity')
    shortest_path = None
```

```python
def find_shortest_path(graph, exhibition_rooms, start_point, end_point):
    # 모든 전시실의 순서를 고려한 경로
    all_paths = list(itertools.permutations(exhibition_rooms))

    # 각 경로에 대해 총 거리를 계산 후 가장 짧은 경로 선택
    shortest_distance = float('infinity')
    shortest_path = None
    for path in all_paths:
        total_distance = 0
        current_point = start_point
        current_path = [start_point]
        for room in path:
            # 현재 위치와 목표 전시실이 같은 층에 있는지 확인
            if room_to_floor_and_elevator[current_point][0] != room_to_floor_and_elevator[room][0]:
                # 같은 층에 없는 경우, 엘리베이터를 이용
                elevator = room_to_floor_and_elevator[room][1]
                distance, _ = dijkstra(graph, current_point, elevator)
                total_distance += distance if distance is not None else float('infinity')
                current_point = elevator
                if elevator not in current_path:
                    current_path.append(elevator)

            # 이제 해당 방으로 이동
            distance, _ = dijkstra(graph, current_point, room)
            if distance is None:  # 경로가 존재하지 않으면 계산을 중단
                break
            total_distance += distance
            current_point = room
            if room not in current_path:
                current_path.append(room)

        else:  # 모든 전시실을 방문한 후에는 끝점으로 돌아옴
            if room_to_floor_and_elevator[current_point][0] != room_to_floor_and_elevator[end_point][0]:
                # 현재 위치와 끝점이 같은 층에 없는 경우, 엘리베이터를 이용
                elevator = room_to_floor_and_elevator[end_point][1]
                distance, _ = dijkstra(graph, current_point, elevator)
                total_distance += distance if distance is not None else float('infinity')
                current_point = elevator
                if elevator not in current_path:
                    current_path.append(elevator)

            # 이제 끝점으로 이동 / 시작점 종료점 중복 제거
            if current_point != end_point:
                distance, _ = dijkstra(graph, current_point, end_point)
                total_distance += distance if distance is not None else float('infinity')
                if end_point not in current_path:
                    current_path.append(end_point)

            if total_distance < shortest_distance:
                shortest_distance = total_distance
                shortest_path = current_path

    return shortest_distance, shortest_path
```

03 결과 및 효과

https://github.com/askjiyun/DataStructure_Project.git

1. 유모차 및 휠체어 이용방문객을 위한 전시실 관람 최단 경로 탐색

museum.py

관람 시작 전시실과 마칠 전시실을 이용객에게 입력 받아
모든 전시실을 관람하는 최단 경로 및 최단 거리(관람시간 포함) 출력

층별 전시실 이동이므로 최단 경로에 엘리베이터 경로 포함하여 출력

```
C:\Users\8x8\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\8x8\PycharmProjects\pythonProject\museum.py
관람을 시작할 전시실을 입력해주세요: A Room
관람을 마칠 전시실을 입력해주세요: F Room
전시실을 모두 관람하는 최단 경로: A Room -> 2F Elevator -> B Room -> C Room -> 3F Elevator -> D Room -> E Room -> F Room
최단 거리(관람시간 포함): 105

C:\Users\8x8\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\8x8\PycharmProjects\pythonProject\museum.py
관람을 시작할 전시실을 입력해주세요: C Room
관람을 마칠 전시실을 입력해주세요: D Room
전시실을 모두 관람하는 최단 경로: C Room -> B Room -> 1F Elevator -> A Room -> 3F Elevator -> E Room -> F Room -> D Room
최단 거리(관람시간 포함): 108

Process finished with exit code 0
```

2. 편의시설 경로 탐색에 추가하여 미술관 관람 최단 경로 탐색

museumSolution.py

```
관람을 시작할 전시실을 입력해주세요: A Room
관람을 마칠 전시실을 입력해주세요: F Room
전시실을 모두 관람하는 최단 경로: A Room -> 2F Elevator -> B Room -> C Room -> 3F Elevator -> D Room -> E Room -> F Room
최단 거리(관람시간 포함): 105
관람을 시작할 전시실을 입력해주세요: A Room
관람을 마칠 전시실을 입력해주세요: F Room
방문하고 싶은 편의시설을 입력하세요 (쉼표로 구분): Nursing Room,SeMA Cafe
선택한 편의시설을 포함한 최단 경로: A Room -> Nursing Room -> SeMA Cafe -> 2F Elevator -> B Room -> C Room -> 3F Elevator -> D Room -> E Room -> F Room
최단 거리(관람시간 포함): 107
```

```
관람을 시작할 전시실을 입력해주세요: C Room
관람을 마칠 전시실을 입력해주세요: D Room
전시실을 모두 관람하는 최단 경로: C Room -> B Room -> 1F Elevator -> A Room -> 3F Elevator -> E Room -> F Room -> D Room
최단 거리(관람시간 포함): 108
관람을 시작할 전시실을 입력해주세요: F Room
관람을 마칠 전시실을 입력해주세요: B Room
방문하고 싶은 편의시설을 입력하세요 (쉼표로 구분): 1F Restroom,Rest Area
선택한 편의시설을 포함한 최단 경로: F Room -> E Room -> D Room -> 1F Elevator -> A Room -> 1F Restroom -> Rest Area -> 2F Elevator -> C Room -> B Room
최단 거리(관람시간 포함): 120
```

- 유모차 이용 관람객의 최단 경로 탐색

: 유모차 이용 관람객이 이용하고 싶은 편의시설을 입력받아 편의시설경로까지 고려한 미술관 관람 최단 경로 탐색 출력

EX) 현재 이용객은 A Room에서 관람을 시작하여 F Room을 끝으로 관람 완료하고 싶으며 수유실과 SeMA Cafe 이용 예정

- 휠체어 이용 관람객의 최단 경로 탐색

: 휠체어 이용 관람객이 이용하고 싶은 편의시설을 입력받아 편의시설경로까지 고려한 미술관 관람 최단 경로 탐색 출력

- 조건 : 1F Restroom만 휠체어 이용객이 사용할 수 있는 화장실임