



융 캠

융합공학부 X 컴퓨터공학부

INDEX

Design of Data Structure



1. Introduction

Member | Environment | Problem Design

Scenario | Implementation

2. DATA processing

3. Implementation

3-1) Graph Formation

3-2) Elevator Simulation

3-3) Shortest Path Algorithm

3-4) Path Visualization

4. RUN [Proto-type]

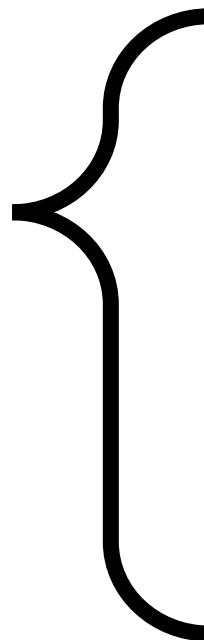
INTRODUCTION

Member | Environment | Problem Design | Scenario | Implementation

Introduction

MEMBERS

...



박지호 (컴퓨터공학부 3학년)

역할 : Main Frame/ Elevator

최영재 (컴퓨터공학부 3학년)

역할 : Graph / Shortest Path

이동민 (디지털이미징학과 2학년)

역할 : Data Processing/ Visualization

정진혁 (디지털이미징학과 2학년) [조장]

역할 : Data Processing/ Visualization

Introduction

ENVIRONMENT

...



협업에 용이한 OOP



Java GUI 활용



Processing 시각화

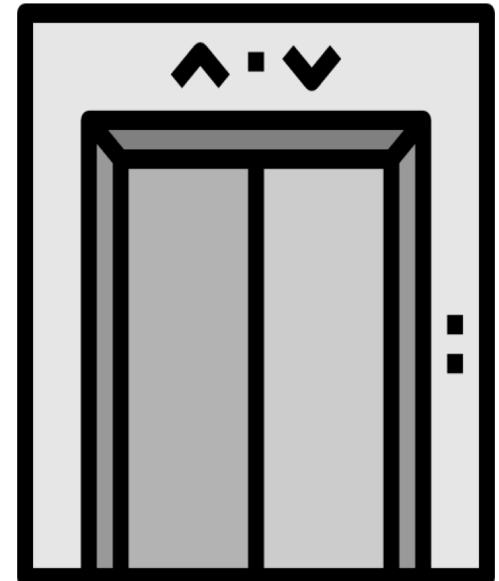
PROBLEM DEFINE

Introduction

“310관 엘리베이터의 효율적 운행”



“개인 시간표에 따른 이동경로 추천”



MAGIC TIME

Introduction

SCENARIO

...



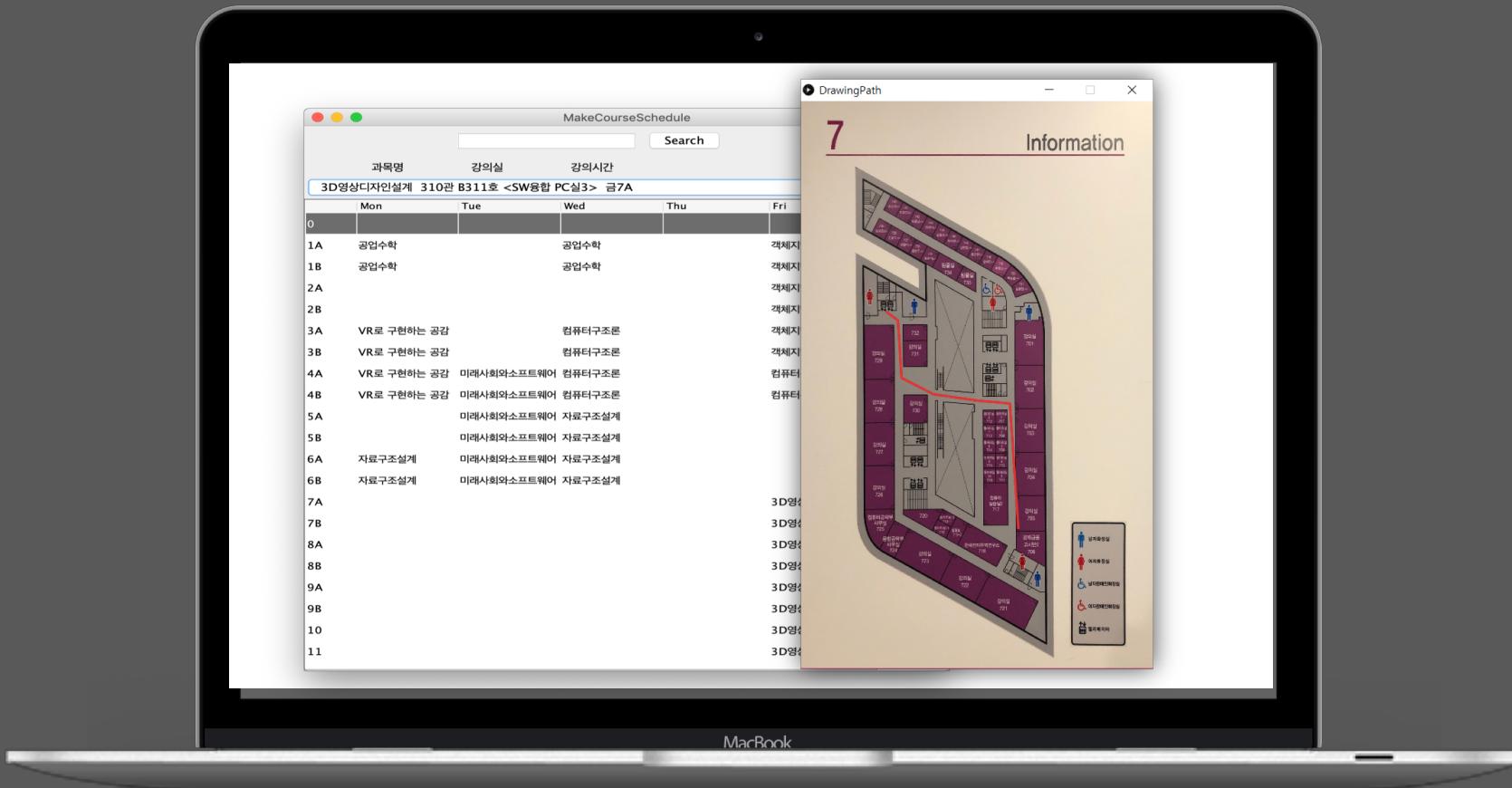
Introduction

IMPLEMENTATION



Using Swing GUI + Processing

Introduction IMPLEMENTATION



Using Swing GUI + Processing

STRUCTURE

Introduction

MAIN

GUI FRAME

GRAPH

SHORTEST-PATH

ELEVATOR-SIMUL.

VISUALIZATION

DATA - PROCESSING

Essential | Complexity | File Management

ESSENTIAL DATA

•••

❖ 필요한 정보

① 엘리베이터 대기시간 | 엘리베이터 실시간 위치

- 새로운 엘리베이터 시스템을 만듦으로써 모든 엘리베이터의 대기시간 평균화 가정.

② 층별 혼잡도

- 강의 정보를 바탕으로 시간대/층별 학생의 수 계산

③ 2학기 전체 강의 정보

- 획득 완료[학사팀].

④ 층별 단면도

- 획득 완료[시설팀] + 직접 제작[1층/B4층]

⑤ 계단 상행/하행 및 강의실 이동시간

- 직접 측정

⑥ 인접 노드 처리 | 강의실 & 교차점 좌표

- 직접 처리

Data Processing

@310 LECTURE DATA

• • •

	B6층	B5층	B3층	B2층	B1층	3층	4층	5층	6층	7층	8층	9층	합계
0교시						65					99		164
1교시A	74	118				352	99	260	343	320	111	159	1644
1교시B													
2교시A			48							32			32
2교시B						220	192	249	44	366	258	464	1793
3교시A	107	79				178	70	295	359	183			1085
3교시B													
4교시A	76	64						157	177	61	127	47	569
4교시B													
5교시A	155	323					135	251	351	192			929
5교시B						182	100	93	217	274	183	473	1522
6교시A						108		86		169			363
6교시B													
7교시A	261	287	27		14	396	328	426	514	559	272	463	2958
7교시B													
8교시A								48		81			129
8교시B										136	57	24	217
9교시A						83		59					142
9교시B													
10교시								37	43	25			105
11교시											98	311	409

Data Processing

FILE FORMAT

• • •

In-Floor DATA

310Graph.txt — 편집됨

```

6N7/N5,N5/602/603,601
6N8/N4,N7,N10/604,605
6N9/N7,N11,E2/601,602,603
6N10/N8,N12/52/
6N11
6N12
6N13
6N14
6N15
7N1/
7N2/
7N3/
7N4/
7N5/
7N6/
7N7/
7N8/
7N9/
7N10/N6,N12,E2/702,701
7N11/N7,N13,E2/

```

Node와 인접해 있는 노드

EX) 7N15/N9,N14,S3,E3/729

(Floor)(Type)(No.)
 N : 교차로
 E : 엘리베이터
 S : 계단
 C : 에스컬레이터

인접해있는 노드 사이에 있는 모든 강의실

[REAL]NodeDistance.txt

```

B6N1/B6E1/6.5
B5N1/B5N2/6.8
B5N1/B4N1/6.5
B4N1/B4N2/6.8
B4N1/B4N3/6.8
B4N1/B4N4/6.8
B4N1/B3N1/6.8
B3N1/B3N2/6.8
B3N1/B3N3/6.8
B3N1/B3N4/6.8
B3N1/B3N5/6.8
B3N2/B3S2/1.2
B3N2/B3E1/1.8

```

도착 노드 이름

EX) 7N15/7N14/2.3

시작 노드 이름

노드 간 거리(cm)

Classroom Dist.txt

```

5N9/502/1.5
5N9/503/3.2
5N11/501/1.1
5N13
5N14
5N14
5N14
6N1/
6N1/
6N1/
6N1/
6N2/
6N3/
6N4/
6N5/
6N5/620/2.2
6N5/621/4.6

```

인접 강의실

EX) 7N4/716/1.3

교차점 이름

강의실 거리(cm)

1. 교차점 노드 데이터

2. 교차점 → 교차점 거리

3. 교차점 → 강의실 거리

Inter-Floor DATA

1. 계단: (A/B/C) 1개 층 올라가는 시간 / 내려가는 시간 (예외: 1층 > 2층)
2. 에스컬레이터: (지하/C) 1개 층 올라가는 시간
3. 엘리베이터: 측정 기반 Magic Time Worst Case

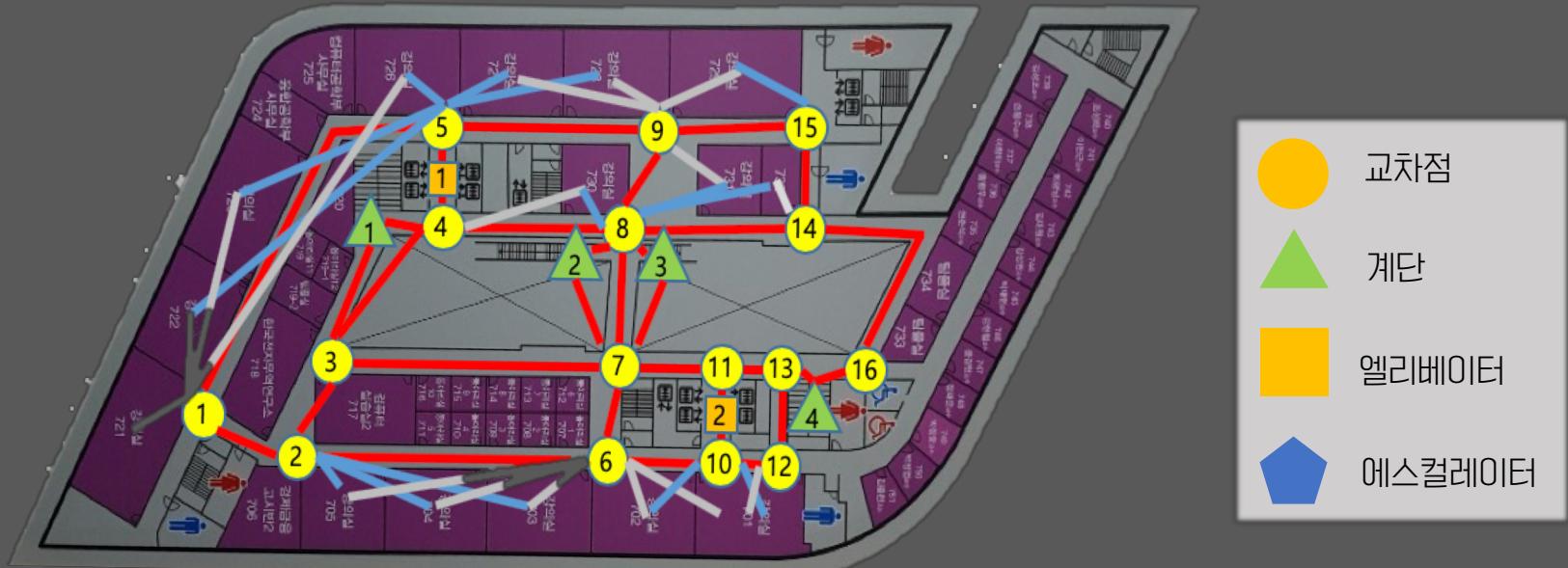
IMPLEMENTATION

Graph Formation | Elevator Simulation | Shortest Path Algorithm | Path Visualization

Implementation : Graph-Formation

GRAPH FORMATION

• • •



#RULE

- 인접한 교차점끼리 먼저 있다.
- 1교차점과 2교차점 사이에 있는 강의실을 1,2교차점에 전부 이어준다.(문 기준)
- 엘리베이터는 오직 교차점과 잇는다.
- 왼쪽 아래를 기준으로 오른쪽으로 가면서 번호를 매긴다.
- 예상 Graph 시나리오

강의실 ->교차점->교차점 -> …->엘리베이터/계단->교차점->…->강의실.

(shortest path algorithm)

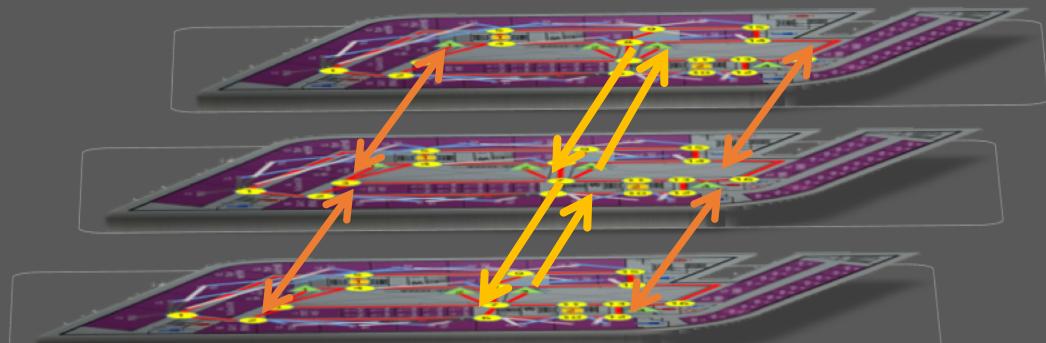
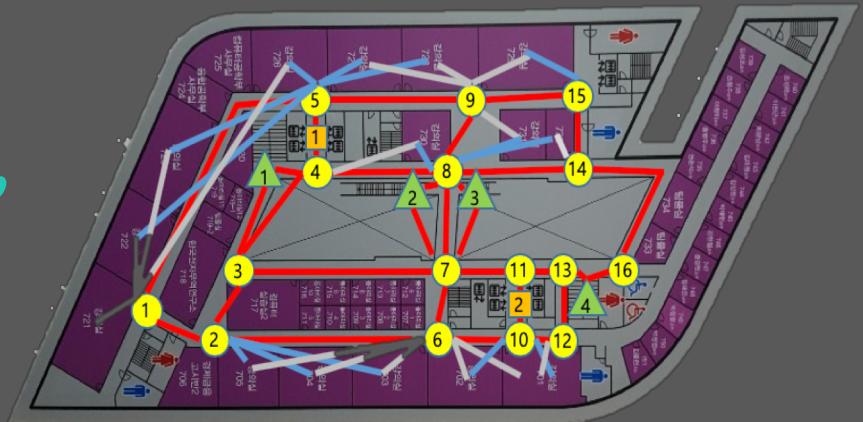
FLOOR INTERCONNECTION

...

“인접 교차점을 이용한 GRAPH 형성”

강의실 → 교차점 → 교차점 → … → 엘리베이터 / 계단

→ 교차점 → … → 강의실. (shortest path algorithm)

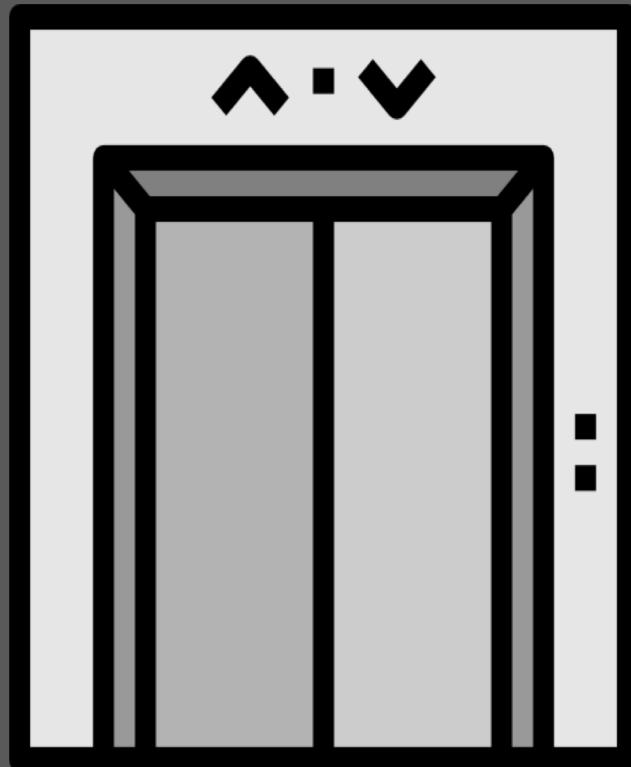


#RULE

- 계단 (A/B구역)은 직선으로 잇는다.
- 계단 (C 구역)은 해당 층 올라옴 = 아래 층 올라감
- 엘리베이터 : Magic Time Simulation 정보 이용
- 에스컬레이터 : 단 방향 연결(지하 / 1>2>3)

ELEVATOR SYSTEM

...



MAGIC TIME

- ✓ Condition ① 해당 시간 310관 이용자 X명 이상일 때,
② 각 층의 이용자 비율 Y%이상인 층 => Noisy 층
(예외) X명 이하일 때, 특정 층이 X/u => Noisy 층
- ✓ Duration : [20분] 15min Before Class ~ 5min After Class
- ✓ Scale : (A구역-지상 / C구역) 1층과 Noisy 층만 운행
(A구역-지하 / B구역-지하) 정상운행
(B구역 -지상) Noisy 층 제외한 훌/짝 운행
- ✓ Limit ① 정상운행 시에도 B구역을 훌/짝 운행.
② 하행은 제외한다. (지하는 Noisy층이 될 수 없다.)

ALGORITHM

•••

✓ Condition

- ① 해당 시간 310관 이용자 X명 이상일 때,
- ② 각 층의 이용자 비율 Y%이상인 층 => Noisy 층
(예외) X명 이하일 때, 특정 층이 Z명 이상 사용 => Noisy 층



모든 요일&시간에서
최대인원 운반 X/Y/Z 반환

1. 요일/시간별 Magic Floor 적용 층 저장 `ArrayList[day][time]`
2. 소요시간 삼차원 배열 저장 `this.Elevator[day][time][Elevator_num]`
3. 20분간 왕복 운행 횟수 계산 `[roundRunningNum()]`
4. 20분간 운반 인원 계산 `[this.totalPeopleNum[day][row]]`
5. 시간/요일별 운반 인원의 평균 계산 `[this.avgPeoplenum]`

단위 시간당 최대의 평균 운반 인원
» X/Y/Z

`runningElevator_Normal`
`runningElevator_MagicTime`
» Graph

Implementation : Elevator Simulation

TEST EVALUATION

• • •

```
<terminated> Main [Java Application] C:\Program Files
```

```
x: 0.0, y: 0.01, z: 0.0, 1445명  
x: 0.0, y: 0.02, z: 0.0, 1446명  
x: 0.0, y: 0.03, z: 0.0, 1449명  
x: 0.0, y: 0.04, z: 0.0, 1452명  
x: 0.0, y: 0.049999997, z: 0.0, 1455명  
x: 0.0, y: 0.059999995, z: 0.0, 1456명  
x: 0.0, y: 0.06999999, z: 0.0, 1463명  
x: 0.0, y: 0.07999999, z: 0.0, 1466명  
x: 0.0, y: 0.08999999, z: 0.0, 1473명  
x: 0.0, y: 0.09999999, z: 0.0, 1478명  
x: 0.0, y: 0.109999985, z: 0.0, 1487명  
x: 0.0, y: 0.11999998, z: 0.0, 1503명  
x: 0.0, y: 0.12999998, z: 0.0, 1519명  
x: 0.0, y: 0.13999999, z: 0.0, 1534명  
x: 0.0, y: 0.14999999, z: 0.0, 1550명  
x: 0.0, y: 0.16, z: 0.0, 1567명  
x: 0.0, y: 0.17, z: 0.0, 1575명  
x: 0.0, y: 0.18, z: 0.0, 1600명  
x: 0.0, y: 0.19000001, z: 0.0, 1611명  
x: 0.0, y: 0.20000002, z: 0.0, 1632명  
x: 0.0, y: 0.21000002, z: 0.0, 1657명
```

```
<terminated> Main [Java Application] C:\Program Files\Java
```

```
x: 100.0, y: 0.08999999, z: 0.0, 1392명  
x: 100.0, y: 0.09999999, z: 0.0, 1397명  
x: 100.0, y: 0.109999985, z: 0.0, 1406명  
x: 100.0, y: 0.11999998, z: 0.0, 1423명  
x: 100.0, y: 0.12999998, z: 0.0, 1438명  
x: 100.0, y: 0.13999999, z: 0.0, 1453명  
x: 100.0, y: 0.14999999, z: 0.0, 1469명  
x: 100.0, y: 0.16, z: 0.0, 1487명  
x: 100.0, y: 0.17, z: 0.0, 1495명  
x: 100.0, y: 0.18, z: 0.0, 1520명  
x: 100.0, y: 0.19000001, z: 0.0, 1531명  
x: 100.0, y: 0.20000002, z: 0.0, 1552명  
x: 100.0, y: 0.21000002, z: 0.0, 1577명
```

```
x: 930.0, y: 0.2, z: 100.0, 1356명  
x: 930.0, y: 0.2, z: 103.0, 1357명  
x: 930.0, y: 0.2, z: 104.0, 1360명  
x: 930.0, y: 0.2, z: 106.0, 1363명  
x: 930.0, y: 0.2, z: 109.0, 1366명  
x: 930.0, y: 0.2, z: 111.0, 1369명  
x: 930.0, y: 0.2, z: 113.0, 1372명  
x: 930.0, y: 0.20300001, z: 113.0, 1375명  
x: 930.0, y: 0.20500001, z: 111.0, 1377명  
x: 930.0, y: 0.20500001, z: 113.0, 1380명  
x: 930.0, y: 0.20600002, z: 113.0, 1382명  
x: 930.0, y: 0.20800002, z: 111.0, 1383명  
x: 930.0, y: 0.20800002, z: 113.0, 1386명  
x: 930.0, y: 0.20900002, z: 113.0, 1389명
```

```
x: 700.0, y: 0.20600002, z: 113.0, 1380명  
x: 700.0, y: 0.20700002, z: 113.0, 1382명  
x: 700.0, y: 0.20800002, z: 111.0, 1383명  
x: 700.0, y: 0.20800002, z: 113.0, 1385명  
x: 700.0, y: 0.20900002, z: 111.0, 1386명  
x: 700.0, y: 0.20900002, z: 113.0, 1389명
```

NORMAL CASE
940 명

VS

MAGIC TIME
1356~1657 명

#Conclusion

: Magic Time 자체는 효율적이다. 하지만
학생 편의성을 위해 추가 조건이 필요하다.

Implementation : Elevator Simulation

TEST FEEDBACK

• • •

```
<terminated> Main [Java Application] C:\Program Files  
x: 0.0, y: 0.01, z: 0.0, 1445명  
x: 0.0, y: 0.02, z: 0.0, 1446명  
x: 0.0, y: 0.03, z: 0.0, 1449명  
x: 0.0, y: 0.04, z: 0.0, 1452명  
x: 0.0, y: 0.049999997, z: 0.0, 1455명  
x: 0.0, y: 0.059999995, z: 0.0, 1456명  
x: 0.0, y: 0.06999999, z: 0.0, 1463명  
x: 0.0, y: 0.07999999, z: 0.0, 1466명  
x: 0.0, y: 0.08999999, z: 0.0, 1473명  
x: 0.0, y: 0.09999999, z: 0.0, 1478명  
x: 0.0, y: 0.109999985, z: 0.0, 1487명  
x: 0.0, y: 0.11999998, z: 0.0, 1503명  
x: 0.0, y: 0.12999998, z: 0.0, 1519명  
x: 0.0, y: 0.13999999, z: 0.0, 1534명  
x: 0.0, y: 0.14999999, z: 0.0, 1550명  
x: 0.0, y: 0.16, z: 0.0, 1567명  
x: 0.0, y: 0.17, z: 0.0, 1575명  
x: 0.0, y: 0.18, z: 0.0, 1600명  
x: 0.0, y: 0.19000001, z: 0.0, 1611명  
x: 0.0, y: 0.20000002, z: 0.0, 1632명  
x: 0.0, y: 0.21000002, z: 0.0, 1657명
```

```
<terminated> Main [Java Application] C:\Program Files\Java  
x: 100.0, y: 0.08999999, z: 0.0, 1392명  
x: 100.0, y: 0.09999999, z: 0.0, 1397명  
x: 100.0, y: 0.109999985, z: 0.0, 1406명  
x: 100.0, y: 0.11999998, z: 0.0, 1423명  
x: 100.0, y: 0.12999998, z: 0.0, 1438명  
x: 100.0, y: 0.13999999, z: 0.0, 1453명  
x: 100.0, y: 0.14999999, z: 0.0, 1469명  
x: 100.0, y: 0.16, z: 0.0, 1487명  
x: 100.0, y: 0.17, z: 0.0, 1495명  
x: 100.0, y: 0.18, z: 0.0, 1520명  
x: 100.0, y: 0.19000001, z: 0.0, 1531명  
x: 100.0, y: 0.20000002, z: 0.0, 1552명  
x: 100.0, y: 0.21000002, z: 0.0, 1577명
```

```
x: 930.0, y: 0.2, z: 100.0, 1356명  
x: 930.0, y: 0.2, z: 103.0, 1357명  
x: 930.0, y: 0.2, z: 104.0, 1360명  
x: 930.0, y: 0.2, z: 106.0, 1363명  
x: 930.0, y: 0.2, z: 109.0, 1366명  
x: 930.0, y: 0.2, z: 111.0, 1369명  
x: 930.0, y: 0.2, z: 113.0, 1372명  
x: 930.0, y: 0.20300001, z: 113.0, 1375명  
x: 930.0, y: 0.20500001, z: 111.0, 1377명  
x: 930.0, y: 0.20500001, z: 113.0, 1380명  
x: 930.0, y: 0.20600002, z: 113.0, 1382명  
x: 930.0, y: 0.20800002, z: 111.0, 1383명  
x: 930.0, y: 0.20800002, z: 113.0, 1386명  
x: 930.0, y: 0.20900002, z: 113.0, 1389명
```

```
x: 700.0, y: 0.20600002, z: 113.0, 1380명  
x: 700.0, y: 0.20700002, z: 113.0, 1382명  
x: 700.0, y: 0.20800002, z: 111.0, 1383명  
x: 700.0, y: 0.20800002, z: 113.0, 1385명  
x: 700.0, y: 0.20900002, z: 111.0, 1386명  
x: 700.0, y: 0.20900002, z: 113.0, 1389명
```

NORMAL CASE
940 명

VS

MAGIC TIME
1389 명

#Conclusion

해당시간에 건물을 이용하는 사람이 930명 이상,
전체 비율의 20.9% 이상인 층들을 Noisy Floor
[930명 미만일 때, 이용자 113명 이상인 층 N.F]

Implementation : Shortest Path Algorithm

DIJKSTRA ALGORITHM

Case 1 : [상행] 엘리베이터 이용

Case 2 : [상행] 에스컬레이터 이용

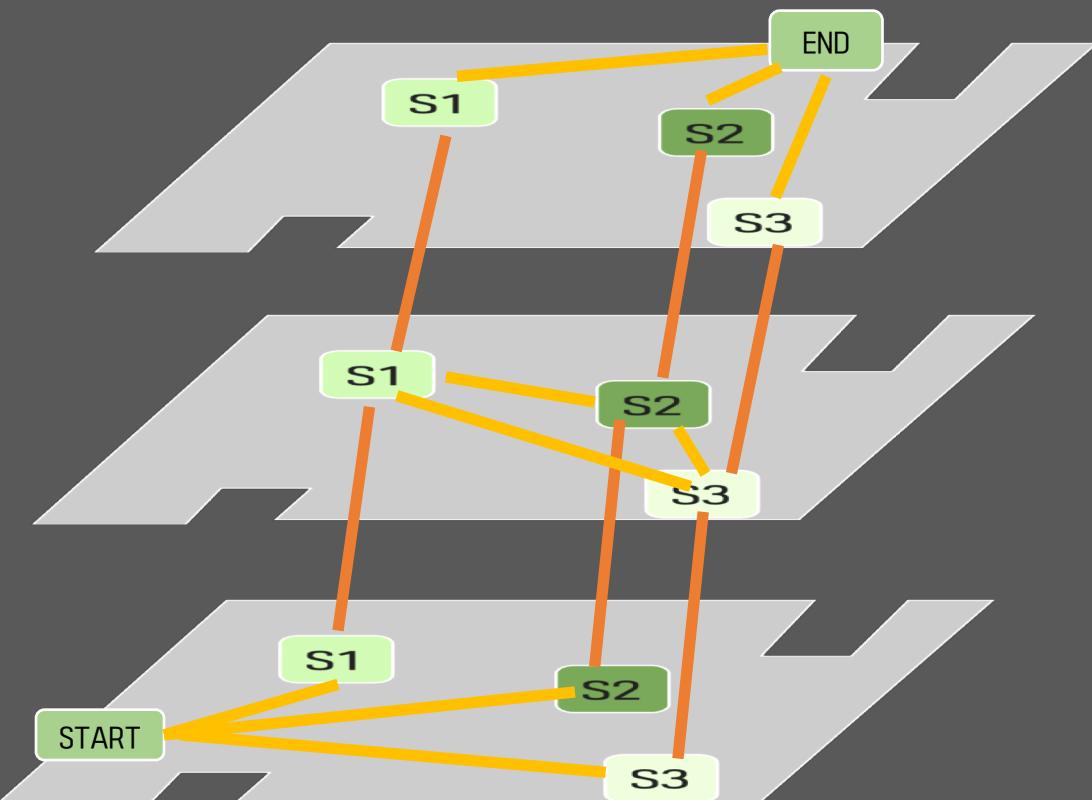
Case 3 : [상행] 에스컬레이터 + 계단 이용

Case 4 : [상행] 계단 이용 (3개 층 이내)

Case 5 : [하행] 계단 이용

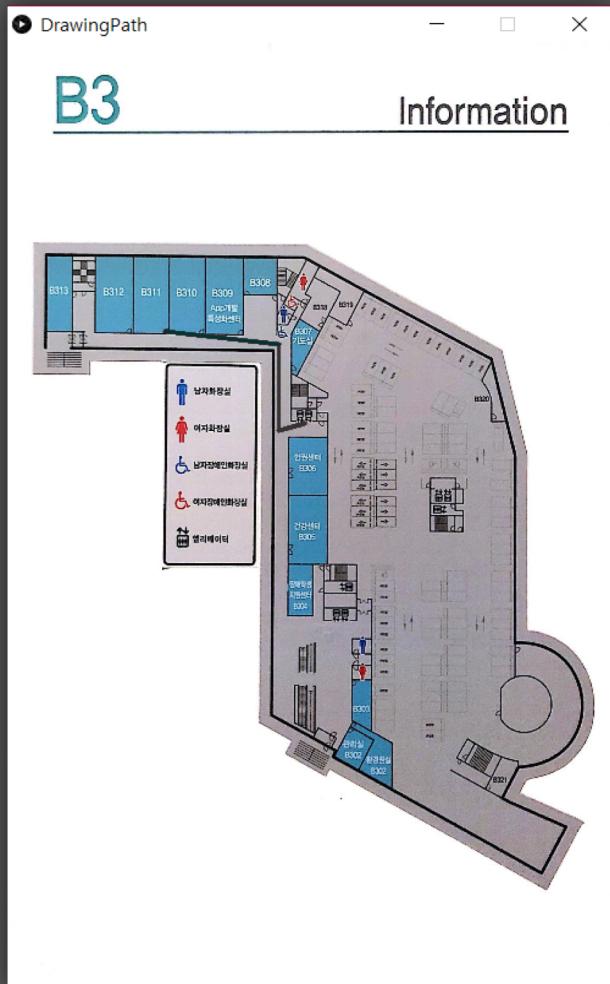
•••

goal



Implementation : Path-Visualization (VER.1) PROCESSING

• • •



Ex. B311 → 705

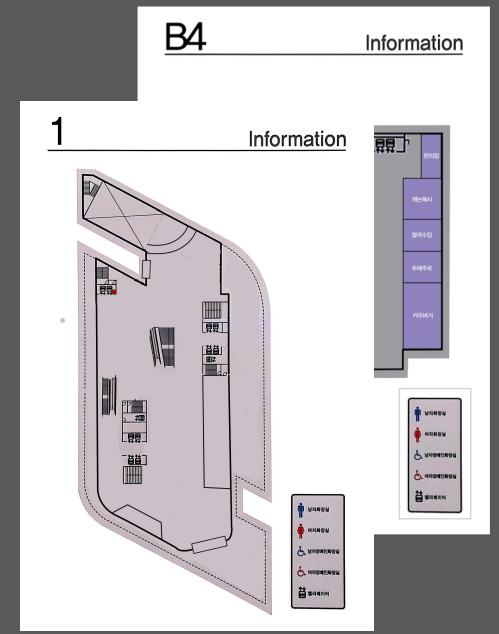
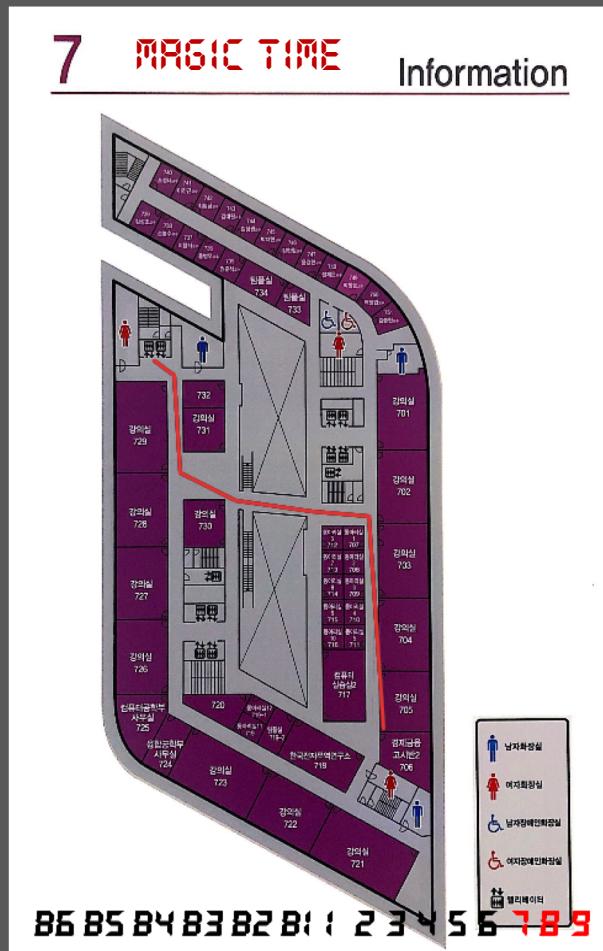
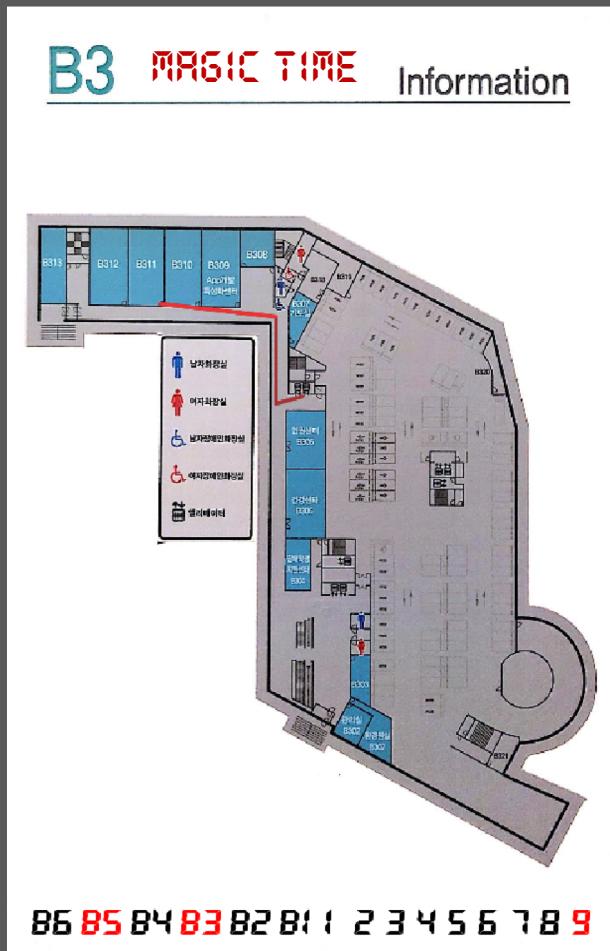
(시작)

- 시작 층 이미지 로드
- 경로 그리기
- 층간 이동 수단 표기
- 도착 층 이미지 로드
- 경로 그리기

(종료)

Implementation : Path-Visualization (VER.2) PROCESSING

• • •

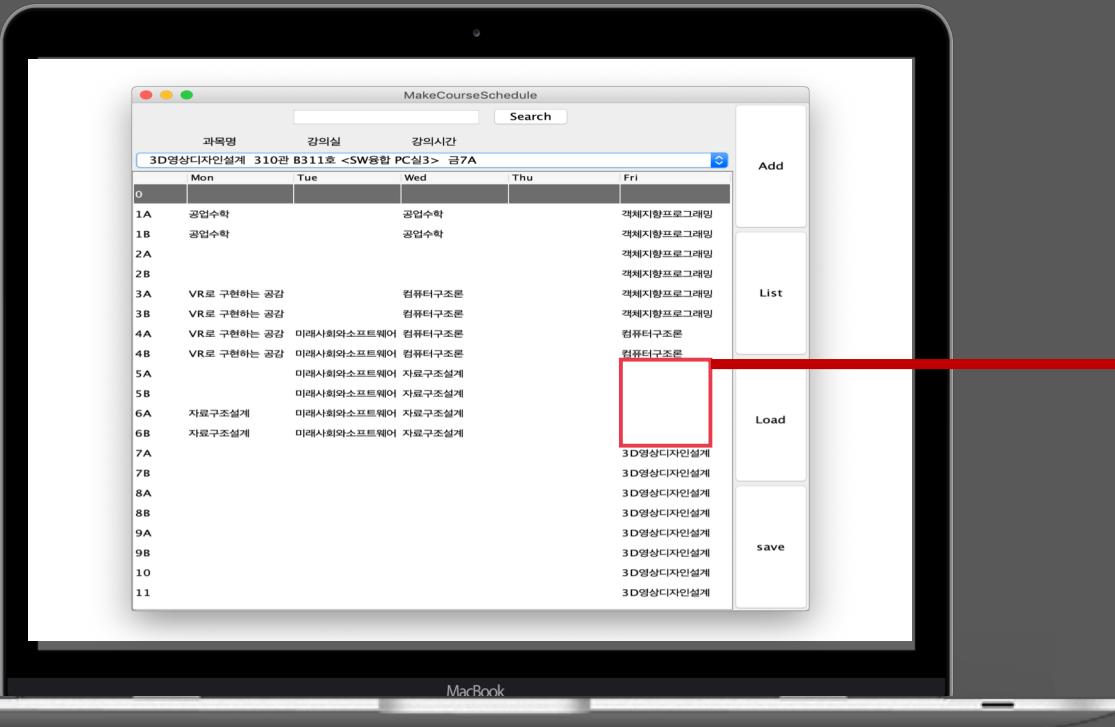


- MAGIC TIME 운행 여부 표기
- Shortest Path ALG. 결과 시각화
- 추천 Elevator의 운행 층 표기

Additional Function

ADD-ON

• • •



Q. 공강이 2시간 이상일 경우!?

A. 제 2공학관 6PC실 안내

[다음 수업의 출발지로 지정]

RUN [DEMO.EXE]

YC @310MAP

QnA

YC @310MAP