기말고사 대체 개인 프로젝트 - 중간 #01

<스마트 주차 관리 시스템>

234042 황선하

① 프로젝트 주제

- 차량의 크기와 특성에 따라 주차 공간을 최적화하고, 주차장의 혼잡도를 관리할 수 있는 스마트 주차 시스템을 개발하는 프로젝트입니다. 이를 통해 사용자는 차량 유형에 맞는 자리를 추천받아 더욱 편리하게 주차할 수 있습니다.

② 프로그램 실행 방법

- 처음 출력된 주차장의 상태를 확인한다.
- 주차할 차량의 종류, 혹은 옵션을 입력한다.
- 추천된 자리 목록을 확인하고 주차할 행과 열을 입력한다. (ex. 1 2)
- 주차를 완료한 후 현 상태의 주차장을 확인한다.

③ 현재까지 구현한 바

- 시작 시 map 출력
- 사용자로부터 종류, 옵션 입력받기
- 대형차: 두 자리에 걸쳐, 한 대 주차 가능
- 그 외 차량: 한 칸 당 한 대 주차 가능
- 전기차: 충전 여부를 입력받고 이에 따라 자리 추천
- 추천 자리는 최대 5개까지

④ 코드 설명

```cpp
#include <iostream>
#include <string>
#include <vector>
using namespace std;

// 주차 공간의 종류 정의
const int EMPTY = 0;        // 비어있는 공간
const int MOTORBIKE = 1;    // 오토바이 공간
const int COMPACT = 2;      // 경차 공간
const int REGULAR = 3;      // 일반차 공간
const int LARGE = 4;        // 대형차 공간
const int ELECTRIC = 5;     // 전기차 공간
```

- **설명:**
  - 각 차량별로 상태를 표시, 숫자를 통해 상태를 표시한다.
  - 이는 변경되지 않는 사항이기 때문에 const로 선언한다.
- **사용된 개념과 주차:**
  - 2주차: constant(상수)

```cpp
// 주자창 크기 설정
const int rows = 4; // 주차장의 세로 길이
const int cols = 10;// 주차장의 가로 길이
```

- **설명:**
  - 주차장의 크기(칸 수)를 설정, 행과 열로 나타낸다.
  - 이는 변경되지 않는 사항이기 때문에 const로 선언한다.
- **사용된 개념과 주차:**
  - 2주차: constant(상수)

```cpp
// 2차원 벡터 선언(주차 상태 표시)
vector<vector<int>> parking;
```

- **설명:**
  - 주차장을 나타내기 위한 2차원 벡터 선언
- **사용된 개념과 주차:**
  - 10주차: vector(2차원 벡터)

```
char parkingSymbol (int type) {
    switch (type) {
        case MOTORBIKE:
            return 'M';
        case COMPACT:
            return 'C';
        case REGULAR:
            return 'R';
        case LARGE:
            return 'L';
        case ELECTRIC:
            return 'E';
        default:
            return ' ';     // 비어있는 공간은 공백으로 표시
    }
}
```

- **반환값:**
  - 각 차량별로 부여된 Symbol
  - 오토바이-M, 경차-C, 일반차-R, 대형차-L, 전기차-E, 빈자리-' '
- **설명:**
  - 차량별로 부여되었던 숫자를 통해 symbol 할당
  - EMPTY(0)의 경우 공백 symbol할당
- **사용된 개념과 주차:**
  - 4주차: Condition(switch)
  - 6주차/7주차: function(char 함수)

```
// 초기 주차 공간 세팅
void setParking() {
    // 왼쪽, 오른쪽 끝 자리를 경차 전용자리로
    for (int i = 0; i < rows; ++i) {
        parking[i][0] = COMPACT;
        parking[i][cols - 1] = COMPACT;
    }

    // 마지막 행의 2~5열(4칸)을 전기차 충전자리로
    for (int i = 1; i < 5; ++i) {
        parking[rows - 1][i] = ELECTRIC;
    }
}
```

- **반환값:**
  - 주차자리 기준 별 C(경차자리) or E(전기차 충전 자리)로 채워진 주차장
- **설명:**
  - 4x10크기의 주자창에서 왼쪽, 오른쪽 끝 자리는 경차 전용자리로 설정
  - 4x10크기의 주자창에서 마지막 행의 2~5까지의 4칸을 전기차 충전자리로 설정
  - 벡터로 선언된 parking을 통해 각 특정 자리에 특징 추가(경차 or 전기차)
- **사용된 개념과 주차:**
  - 6/7주차: function(void 함수)

■ 10주차: vector(2차원 벡터의 인덱스 접근)

```
// 주차 상태 맵 출력 함수
void displayMap() {
    // 가로 경계선
    int width = cols * 4 + 1;
    string line;
    for (int i = 0; i < width; ++i) {
        line += '-';
    }

    // 각 자리에 대한 주차 자리(상태) 출력
    // '\'를 통해 구분
    for (int i = 0; i < rows; ++i) {
        cout << line << endl;
        for (int j = 0; j < cols; ++j) {
            // 공간에 맞는 symbol가져오기
            char symbol = parkingSymbol(parking[i][j]);
            cout << "| " << symbol << " ";
        }
        cout << "|" << endl;
    }
    cout << line << endl;
}
```

- **반환값:**
  - ■ 특수한 자리와 현재 주차된 상태가 나타난 주차장 출력
- 설명:
  - ■ '-'를 통해 가로로 행 분리
  - ■ '₩'를 통해 세로로 열 분리
  - ■ 행과 열만큼 반복하여 parking의 각 자리에 나타난 값(주차 자리 or 상태 표시)에 맞는 symbol 출력
- **사용한 개념과 주차:**
  - ■ 6/7주차: function(void 함수)
  - ■ 7주차: string(문자열)
  - ■ 5주차: Loop(for)(중첩 for)
  - ■ 10주차: vector(2차원 벡터의 인덱스 접근)

```cpp
// 차량 타입에 따른 주차 공간 추천 함수
vector<pair<int, int>> recommendSpots(int type, bool charging = false, int max = 5) {
    vector<pair<int, int>> recommend; // 추천되는 주차 공간을 저장할 벡터

    // 주차 자리를 순차적으로 훑으며 탐색
    // 일정 개수를 채울 경우 탐색 중단 후 반환
    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            // 전기차일 경우, 충전 여부에 따라 추천
            if (type == ELECTRIC) {
                if (charging && parking[i][j] == ELECTRIC) {
                    recommend.push_back({i, j});  // 충전이 필요한 경우 전기차 자리 추천
                }
                else if (!charging && parking[i][j] == EMPTY) {
                    recommend.push_back({i, j});  // 충전이 아닌 경우 일반 자리 추천
                }
            }
            // 일반 차량 주차 공간 추천
            else if (type == COMPACT && (parking[i][j] == COMPACT || parking[i][j] == EMPTY))
                recommend.push_back({i, j});
            else if (type == LARGE && (j < cols - 1 && (parking[i][j] == EMPTY && parking[i][j + 1] == EMPTY)))
                recommend.push_back({i, j});
            else if (type == REGULAR && parking[i][j] == EMPTY)
                recommend.push_back({i, j});
            else if (type == MOTORBIKE && parking[i][j] == EMPTY)
                recommend.push_back({i, j});

            // 추천해야할 최대 수를 찾은 경우 반환(그만큼을)
            if (recommend.size() >= max)
                return recommend;
        }
    }
    return recommend;
}
```

- **반환값:**
  - 추천자리로 채워진 벡터 반환
- **설명:**
  - 벡터 함수 안에 벡터를 생성하여 조건에 해당하는 주차 자리를 recommend 벡터에 push
  - 행과 열만큼 반복하여 각 차량 종류별로 조건을 확인한 후 그 자리를 추천 목록에 추가
  - 전기차: 전기차이며 + 충전을 원하고 + 자리가 전기차 전용인 경우
  - 전기차: 전기차이며 + 충전을 원하지 않고 + 자리가 비어있는 경우
  - 경차: 경차이며 + 자리가 경차 전용이고 + 자리가 비어있는 경우
  - 대형차: 대형차이며 + 열이 마지막 위치보다 1 왼쪽이고 + 자리와 그 옆자리가 비어있는 경우
  - 일반차: 일반차이며 + 자리가 비어있는 경우
  - 오토바이: 오토바이이며 + 자리가 비어있는 경우
  - 벡터에 값이 5개까지 쌓이면 탐색을 종료하고 이 자리 목록을 반환한다.
  - type외의 인자값을 미리 지정하여, 조건이 필요없는 차량의 사용을 용이화
- **사용한 개념과 주차:**
  - 5주차: Loop(중첩 for)

- 10주차: vector(2차원 벡터의 인덱스 접근)
- 10주차: vector(2차원 벡터)
- 10주차: vector(2차원 벡터의 값 선언(push_back))
- 6/7주차: function(함수)
- 4주차: Condition(if, else if)

```
// 차량 주차 함수
void park(int row, int col, int type) {
    // 대형차의 경우 두 칸 차지
    if (type == LARGE) {
        parking[row][col] = LARGE;
        parking[row][col + 1] = LARGE;
    }
    // 그외의 경우 그 자리만
    else {
        parking[row][col] = type;
    }
    // 주차 후에는 현재 맵 출력
    displayMap();
}
```

- **반환값:**
  - Parking 벡터에서 해당하는 행과 열에 해당하는 차량 저장
  - 차량 대입 후 현재 맵 출력
- **설명:**
  - 대형차(LARGE)의 경우 입력된 칸 옆까지 2자리를 차지
  - 그 외의 경우 입력된 행과 열에 해당하는 칸만 차량 symbol값 대입
  - 대입이 완료된 후에는 주차장의 모습 출력
- **사용된 개념과 주차:**
  - 6/7주차: function(void 함수)
  - 4주차: Condition(if, else)

----------**main문 시작**----------

```
int main() {
    // 상태 대입
    // 행의 크기만큼 열을 제작하여 push
    // 비어있는 상태(EMPTY)로 초기화
    for (int i = 0; i < rows; ++i) {
        vector<int> row(cols, EMPTY);
        parking.push_back(row);
    }
    setParking();   // 초기 설정
    displayMap();   // 맵 출력
```

-
- **설명:**

- 반복문을 통해 주차장의 상태를 나타내는 parking 초기화
- 행만큼 반복하여 EMPTY(symbol)로 채워진 열을 반환
- 이를 행 하나하나에 push하여 parking 채우기
- 초반 설정으로서 모든 칸은 EMPTY가 된다.
- 이후 setParking함수를 통해 특수한 자리 설정
- 완성된 parking에 따른 주차장의 모습 출력

- **사용된 개념과 주차:**
  - 10주차: vector(2차원 벡터)
  - 5주차: Loop(for)
  - 10주차: vector(2차원 벡터의 값 선언(push_back))

```
// mud game처럼 종료신호를 받기 전까지 무한 루프
while(true){
    // #1. 명령어 입력받기
    string command;
    cout << "차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : ";
    cin >> command;
```

- **설명:**
  - 무한루프를 도는 while을 통해 프로그램 동작
  - 프로그램 종료는 break 사용
  - String command를 통해 사용자의 차량, 옵션 입력 받고 변수 저장
- **사용한 개념과 주차:**
  - 5주차: Loop(while)
  - 7주차: 문자열(string)

```
// #2. exit 명령어가 들어온 경우
if (command == "exit")
    break;  // 프로그램 종료
// #3. map 명령어가 들어온 경우
else if (command == "map") {
    displayMap(); // 현재 상태의 map 출력
    continue;   // 다음 명령어 받기
}
```

- **설명:**
  - 입력된 명령어가 exit인 경우 break를 통해 while문 탈출 후 프로그램 종료
  - 입력된 명령어가 map인 경우 현재 주차장의 모습을 출력하는 displayMap 출력 후 while문의 위로(다시 명령어 받기)(continue)
- **사용한 개념과 주차:**
  - 4주차: Condition(if, else if)

```
// #4. 전기차의 충전 선택 확인
bool charging = false;  // 기본 상태는 충전X
char answer;
if (command == "electric") {
    cout << "충전을 원하십니까? (y/n) : ";
    cin >> answer;
    if (answer == 'y') {
        charging = true;    // 충전을 원할 경우 TRUE
    }
}
```

- **설명:**
  - 전기차의 경우 충전 확인을 위한 추가 질문이 필요
  - Charging을 통해 충전여부 확인(t-충전, f-충전x)
  - 사용자의 대답이 y인 경우 true로, n인 경우 false로 설정
- **사용한 개념과 주차:**
  - 4주차: Condition(중첩 if)

```
// #5. 차량 타입에 따른 주차 공간 추천
vector<pair<int, int>> recommendedSpots;    // 추천된 주차 공간(주소)를 저장할 벡터

// 각 차량 타입을 확인하고 이에 따라 자리 추천
if (command == "motorbike") {
    recommendedSpots = recommendSpots(MOTORBIKE);
}
else if (command == "compact") {
    recommendedSpots = recommendSpots(COMPACT);
}
else if (command == "regular") {
    recommendedSpots = recommendSpots(REGULAR);
}
else if (command == "large") {
    recommendedSpots = recommendSpots(LARGE);
}
// 충전이 필요한 전기차의 경우
else if (command == "electric" && charging) {
    recommendedSpots = recommendSpots(ELECTRIC, true);
}
// 충전이 필요없는 전기차의 경우
else if (command == "electric") {
    recommendedSpots = recommendSpots(ELECTRIC);
}
```

- **설명:**
  - 각 차량종류 별 추천받은 자리를 저장하기 위한 벡터 선언
  - 입력받은 명령어를 확인하여 이에 따른 recommendSpots함수의 추천 자리를 받고 이를 벡터에 저장
  - 전기차의 경우 충전여부(charging)를 확인하여 필요한 경우 charging 파라미

터에 true 조건을 추가하여 함수 실행

- **사용한 개념과 주차:**
  - 10주차: vector(2차원 벡터)
  - 4주차: Condition(if, else if)

```
// #6. 추천된 자리 출력 및 주차
if (recommendedSpots.empty() == 0) {
    cout << "추천 주차 자리 : ";
    for (auto spot : recommendedSpots) {
        cout << "(" << spot.first << ", " << spot.second << ") ";
    }
    cout << endl;

    // 자리 선택
    int selectR, selectC;
    cout << "주차할 자리를 선택하세요 (행, 열) : ";
    cin >> selectR >> selectC;

    // 선택된 자리가 칸을 넘어가지 않는 지 확인 후 추자 처리
    // 각 차량별로 맞는 Symbol사용
    if (selectR >= 0 && selectR <= rows && selectC >= 0 && selectC <= cols) {
        // 차량의 종류에 맞게 주차 처리
        if (command == "motorbike") {
            park(selectR, selectC, MOTORBIKE);
        }
        else if (command == "compact") {
            park(selectR, selectC, COMPACT);
        }
        else if (command == "regular") {
            park(selectR, selectC, REGULAR);
        }
        else if (command == "large") {
            park(selectR, selectC, LARGE);
        }
        else if (command == "electric") {
            park(selectR, selectC, ELECTRIC);
        }
    }
}
```

- **설명:**
  - 각 차량종류별 추천받은 자리를 담은 벡터(1개라도 채워진)를 출력
  - 이에 따른 사용자의 선택을 입력받기(selectR, selectC)
  - 입력받은 칸이 주차장의 공간을 넘어서지 않는 경우 입력된 command를 확인하고 이에 따른 symbol로 parking에 값 초기화
  - park함수를 통해 해당 행과 열의 parking symbol 변경(' ' -> 차량)
- **사용한 개념과 주차:**
  - 4주차: Condition(중첩 if, else if)

⑤ 실행 결과

```
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : motorbike
추천 주차 자리 : (0, 1) (0, 2) (0, 3) (0, 4) (0, 5)
주차할 자리를 선택하세요 (행, 열) : 0 1
----------------------------------------
| C | M |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : compact
추천 주차 자리 : (0, 0) (0, 2) (0, 3) (0, 4) (0, 5)
주차할 자리를 선택하세요 (행, 열) : 0 0
----------------------------------------
| C | M |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : regular
추천 주차 자리 : (0, 2) (0, 3) (0, 4) (0, 5) (0, 6)
주차할 자리를 선택하세요 (행, 열) : 0 2
----------------------------------------
| C | M | R |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
```

```
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : large
추천 주차 자리 : (0, 3) (0, 4) (0, 5) (0, 6) (0, 7)
주차할 자리를 선택하세요 (행, 열) : 0 3
----------------------------------------
| C | M | R | L | L |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : electric
충전을 원하십니까? (y/n) : y
추천 주차 자리 : (3, 1) (3, 2) (3, 3) (3, 4)
주차할 자리를 선택하세요 (행, 열) : 3 1
----------------------------------------
| C | M | R | L | L |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : electric
충전을 원하십니까? (y/n) : n
추천 주차 자리 : (0, 5) (0, 6) (0, 7) (0, 8) (1, 1)
주차할 자리를 선택하세요 (행, 열) : 0 5
----------------------------------------
| C | M | R | L | L | E |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : map
----------------------------------------
| C | M | R | L | L | E |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : exit
```

⑥ 코드 문제점

- 1. C와 E의 경우 이미 주차되어 있는 자리임에도 표시가 동일하여 중복되어 추천된다.

```
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : compact
추천 주차 자리 : (0, 0) (0, 1) (0, 2) (0, 3) (0, 4)
주차할 자리를 선택하세요 (행, 열) : 0 0
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : compact
추천 주차 자리 : (0, 0) (0, 1) (0, 2) (0, 3) (0, 4)
주차할 자리를 선택하세요 (행, 열) :
```

- 2. 이미 채워진 칸도 주차가 가능하다

```
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : regular
추천 주차 자리 : (0, 1) (0, 2) (0, 3) (0, 4) (0, 5)
주차할 자리를 선택하세요 (행, 열) : 0 2
----------------------------------------
| C |   | R |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : regular
추천 주차 자리 : (0, 1) (0, 3) (0, 4) (0, 5) (0, 6)
주차할 자리를 선택하세요 (행, 열) : 0 2
----------------------------------------
| C |   | R |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C |   |   |   |   |   |   |   | C |
----------------------------------------
| C | E | E | E | E |   |   |   | C |
----------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) :
```

- 3. 경차를 입력 시 경차 자리만 추천되지 않는다.

```
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C | E | E | E | E |   |   |   | C |
-------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : compact
추천 주차 자리 : (0, 0) (0, 1) (0, 2) (0, 3) (0, 4)
주차할 자리를 선택하세요 (행, 열) :
```

- 4. 추천 이외의 자리를 아무 경고문 없이 주차 가능하다.

```
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C | E | E | E | E |   |   |   | C |
-------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : compact
추천 주차 자리 : (0, 0) (0, 1) (0, 2) (0, 3) (0, 4)
주차할 자리를 선택하세요 (행, 열) : 3 4
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C | E | E | E | C |   |   |   | C |
-------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) :
```

- 5. 잘못된 입력이 들어왔을 때 아무 경고가 없다.

```
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C |   |   |   |   |   |   |   | C |
-------------------------------------
| C | E | E | E | E |   |   |   | C |
-------------------------------------
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) : eelectric
차량의 종류나 옵션을 입력해주세요 (motorbike, compact, regular, large, electric, map, exit) :
```

⑦ 개선할 점

- 오토바이는 한 자리에 2대까지 주차할 수 있다.
- 이미 오토바이가 한 칸에 한 대가 차 있는 경우 나중 오토바이가 입력되었을 때

그 자리가 우선시된다.

- 경차자리가 충분할 경우 5개의 추천 목록은 모두 경차자리로 채운다.
- 이미 채워진 자리는 다른 차량으로 다시 채워지지 않는다.
- 자리(특성) 표시와 주차 표시는 구분하여 나타낸다.
- 추천 외의 자리를 선택할 경우 이에 대한 안내 문구를 출력한다
- 다른 차는 전용자리를 선택할 수 없도록 한다.
- 최대한 대형차의 자리가 많이 남도록 자리를 추천한다.