TPP2022-HW1-40947023s 蘇子權 **（請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。）**

```cpp
/* ----------------------------------------------------------------
// main.cpp
// --------------------------------------------------------------- */
#include <iostream>
#include <string>
#include <vector>
#include "Maze.h"
#include "Robot.h"
int main()
{
    int direction=0; // direction 0,1,2,3 -> up, right, down, left
    std::pair<int,int> targetPosition;
    const int width = []() -> int{
        int tmp;
        std::cin >> tmp;
        return tmp;
    }();
    const int height = []() -> int{
        int tmp;
        std::cin >> tmp;
        return tmp;
    }();
    const long long totalStep = []() -> long long{
        long long tmp;
        std::cin >> tmp;
        return tmp;
    }();
    Maze _maze(width,height);
    for (int i = 0; i < height; i++) {
        std::string lineOfMap;
        std::cin >> lineOfMap;
        _maze.generateMazeMap(lineOfMap);
    }
```

TPP2022-HW1-40947023s 蘇子權 **(請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。)**

```
std::pair<int,int> nowRobotPosition=_maze.findRobotLocation();
    Robot _robot(nowRobotPosition, direction);
    for(long long i=0;i<totalStep;i++){
        std::pair<int,int> nextPos=_robot.nextPosition();
        while(!_maze.isRobotInMaze(nextPos)
               || _maze.isObstacle(nextPos)){
            _robot.turnRight();
            direction=_robot.getRobotToward();
            nextPos=_robot.nextPosition();
        }
        if(_robot.isVisitedAndSameToward(
           nowRobotPosition.first,
           nowRobotPosition.second,
           direction)){
            _robot.findFinalState(totalStep,i);
            break;
        }
        else{
            _robot.storeVisited(nowRobotPosition);
            _robot.walkNextPosition();
            nowRobotPosition=_robot.getPosition();
        }
    }
    targetPosition=_robot.getPosition();
    std::cout << targetPosition.first << " "
              << targetPosition.second <<  std::endl;
}
```

TPP2022-HW1-40947023s 蘇子權 **（請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。）**

```
/* ————————————————————————————————————————————————
// Maze.h
// ———————————————————————————————————————————————— */
#ifndef MAZE_H
#define MAZE_H
#include <iostream>
#include <string>
#include <vector>
class Maze{
    private:
        const int width, height;
        std::vector<std::string> mazeMap;
    public:
        Maze(const int _width, const int _height);
        std::pair<const int,const int> findRobotLocation(void)const;
        void generateMazeMap(const std::string str);
        bool isObstacle(const std::pair<int,int> robotPosition)const;
        bool isRobotInMaze(
                        const std::pair<int,int> robotPosition)const;
};
#endif
```

TPP2022-HW1-40947023s 蘇子權 **(請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。)**

```
/* ------------------------------------------------------------------
// Maze.cpp
// ---------------------------------------------------------------- */
#include "Maze.h"
Maze::Maze(const int _width, const int _height):width(_width),
height(_height){}
std::pair<const int,const int> Maze::findRobotLocation(void)const{
    for(int y=0;y<height;y++){
        for(int x=0;x<width;x++){
            if(mazeMap[y][x]=='O'){
                return std::make_pair(x,y);
            }
        }
    }
    return std::make_pair(0,0);
}
void Maze::generateMazeMap(const std::string str){
    mazeMap.emplace_back(str);
}
bool Maze::isObstacle(const std::pair<int,int> robotPosition)const{
    return mazeMap[robotPosition.second][robotPosition.first]=='#';
}
bool Maze::isRobotInMaze(const std::pair<int,int> robotPosition)const{
    return (robotPosition.first>=0
              && robotPosition.first<width
              && robotPosition.second>=0
              && robotPosition.second<height
          );
}
```

```
/* ────────────────────────────────────────────────────────────────
// Robot.h
// ────────────────────────────────────────────────────────────── */
#ifndef ROBOT_H
#define ROBOT_H
#include <iostream>
#include <vector>
class Robot{
    private:
        int x,y,Face; // robot position
        const int xQuadrant=0, yQuadrant=1;
        const int faceToward[4][2]={{0,-1},{1,0},{0,1},{-1,0}};
        std::vector<std::pair<std::pair<int,int>,int>> visited;
        /* first.first=>x, first.second=>y, second=>face direction
           which have visited */
    public:
        Robot(const std::pair<const int,const int>
              pos, const int face);
        bool isVisitedAndSameToward(
                 const int _x,
                 const int _y,
                 const int _Face
              )const;
        std::pair<const int,const int> getPosition(void)const;
        std::pair<const int,const int> nextPosition(void)const;
        void walkNextPosition(void);
        void turnRight(void);
        int getRobotToward(void)const;
        void storeVisited(const std::pair<const int,const int>
                          robotPosition);
        void findFinalState(
                 const long long n,
                 const long long current_n
              );
};
#endif
```

TPP2022-HW1-40947023s 蘇子權 **（請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。）**

```
/* ─────────────────────────────────────────────────────
// Robot.cpp
// ───────────────────────────────────────────────────── */
#include "Robot.h"
Robot::Robot(const std::pair<const int,const int> pos, const int
face):x(pos.first), y(pos.second), Face(face){}
bool Robot::isVisitedAndSameToward(const int _x, const int _y, const
int _face)const{
    for(auto i:visited){
        if(i.first.first==_x && i.first.second==_y && i.second==_face){
            return true;
        }
    }
    return false;
}
std::pair<const int,const int> Robot::getPosition(void)const{
    return std::make_pair(x,y);
}
std::pair<const int,const int> Robot::nextPosition(void)const{
    return std::make_pair(
                    x+faceToward[Face][xQuadrant],
                    y+faceToward[Face][yQuadrant]
                 );
}
void Robot::walkNextPosition(void){
    x+=faceToward[Face][xQuadrant];
    y+=faceToward[Face][yQuadrant];
}
void Robot::turnRight(void){
    Face=(Face+1)%4;
}
int Robot::getRobotToward(void)const{
    return Face;
}
void Robot::storeVisited(const std::pair<const int,const int>
                         robotPosition){
    visited.emplace_back(std::make_pair(robotPosition,Face));
}
```

TPP2022-HW1-40947023s 蘇子權 **（請記得修改這裡！未填學號與姓名將會扣很大！若為多人作業，請列出所有成員。）**

```
void Robot::findFinalState(
                  const long long totalStep,
                  const long long currentTotalStep
              ){
    long long i,loopSize=1,remainTotalStep=totalStep-currentTotalStep;
    const long long visited_length=visited.size();
    for(i=visited_length-1;i>=0;i--){
        if(visited[i].first.first==x
           && visited[i].first.second==y
           && visited[i].second==Face){
            break;
        }
        loopSize++;
    }
    long long finalState=(remainTotalStep%loopSize)+i;
    x=visited[finalState].first.first;
    y=visited[finalState].first.second;
    Face=visited[finalState].second;
}



/* ------------------------------------------------------------------
// makefile
// ---------------------------------------------------------------- */
CC= g++
CFLAGS= -Wall -Wextra -std=c++14 -O2
LDLIBS= -lm
NAMEONE= main
ONE= Main.o Maze.o Robot.o

.PHONY: all clean

all: main

main: Main.o Maze.o Robot.o
	$(CC) $(CFLAGS) -o $(NAMEONE) $(ONE)  $(LDLIBS)
clean:
	-rm -rf $(NAMEONE) $(ONE)
```