```
/*
-----------------------------------------------------------------------------------
Laboratory  : labo_11
File        : map.h
Author(s)   : Yannick Schaufelberger et David Gallay
Date        : 18.01.2020

Goal        : Library defining the Map and functions to modify, get values or
 * display one
Remark(s)   :

                There is the github repository:
                https://github.com/dgheig/labo11

Compiler    : MinGW-g++ 6.3.0 and g++ 7.4.0
-------------------------------------------------------------------------------------*/

#ifndef MAP_H
#define MAP_H

#include <cstdlib> //size_t
#include <vector>

enum MapState {
    MS_OUT,
    MS_EARTH,
    MS_WATER,
    MS_TREASURE,
    MS_START
};

const int NUMBER_OF_LAKE = 3;

typedef std::vector<MapState> Axe;
typedef std::vector<Axe> Map;

/**
 * @param map
 * @return the height of the map aka the size of the vector<Axe>
 */
size_t getHeight(const Map& map);
/**
 * @param map
 * @return the width  of the map aka the size of the vector<MapState>
 */
size_t getWidth(const Map& map);

/**
 * @param height
 * @param width
 * @return an Map with the given sizes and filled with MS_EARTH
 */
Map getEmptyMap(size_t height, size_t width);

/**
 * @param map
 * @param x
 * @param y
 * @return the MapState at the given coordinates
 */
MapState getMapValue(const Map& map, size_t x, size_t y);
/**
 * @brief sets the MapState at the given coordinates to the given value
 * @param map
 * @param x
 * @param y
 * @param value
 * @return true if success, false if not
 */
bool setMapValue(Map& map, size_t x, size_t y, MapState value);

/**
 * @brief if the MapState at the given coordinates is MS_EARTA, sets it to MS_TREASURE
 * @param map
 * @param height
 * @param width
 * @return true if success, false if not
 */
bool addTreasure(Map& map, size_t height, size_t width);
/**
 * @brief sets a random valid cell to MS_TREASURE
 * @param map
 */
void addRandomTreasure(Map& map);

/**
 * @brief if the given coordinates and radius are valid, adds a lake to the map
 * @param map
 * @param originX
 * @param originY
 * @param radius
 * @return true if success, false if not
 */
bool addLake(Map& map, size_t originX, size_t originY, size_t radius);
/**
 * @brief calls addRandomLake with the same map and a radius set to the third
 * of the smallest size
 * @param map
 */
void addRandomLake(Map& map);
/**
 * @brief adds a lake at random coordinates with a random radius
 * @param map
 * @param maxRadius
 */
void addRandomLake(Map& map, size_t maxRadius);

/**
 * @brief if the MapState at the given coordinates is MS_EARTA, sets it to MS_START
 * @param map
 * @param height
 * @param width
 * @return true if success, false if not
 */
bool addStart(Map& map, size_t x, size_t y);
/**
 * @brief calls addRandomStart with an x and y parameter
 * @param map
 */
void addRandomStart(Map& map);
/**
 * @brief sets a random valid cell to MS_START.
 * @param map
 * @param x
 * @param y
 */
void addRandomStart(Map& map, size_t& x, size_t& y);

/**
 * @param height the height of the map
 * @param width the width of the map
 * @param x the x coordinate of the MS_START
 * @param y the y coordinate of the MS_START
 * @return a map with NUMBER_OF_LAKE lakes, one start and one treasure
 */
Map initWorld(size_t height, size_t width, size_t& startX, size_t& startY);

/**
 * @brief displays the map
 * @param map
 */
void displayWorld(const Map& map);

#endif // MAP_H
```