/\*

\* Group Members: Hiroya Gojo, Zhongming Liao, Monty Choy Ziling Yu, Amy Petrine

\*/

// DO NOT CHANGE OR REMOVE THE FOLLOWING LINES

#ifndef \_\_DEFINE\_MAP\_FUNCTIONS\_CPP\_\_

#define \_\_DEFINE\_MAP\_FUNCTIONS\_CPP\_\_

#include <cstring>

#include <fstream>

#include <iostream>

using namespace std;

// DO NOT CHANGE OR REMOVE THE PRECEDING LINES

/\*

\* MAP CONSTANTS

\*/

const char MAP\_SQUARE\_CHASM = 'O';

const char MAP\_SQUARE\_EMPTY = ' ';

const char MAP\_SQUARE\_KEY = 'k';

const char MAP\_SQUARE\_LOCK = '@';

const char MAP\_SQUARE\_PEBBLE = '.';

const char MAP\_SQUARE\_PEBBLES = ':';

const char MAP\_SQUARE\_PLANK = '=';

const char MAP\_SQUARE\_PLANK\_SET = 'I';

const char MAP\_SQUARE\_ROCK = '#';

const char MAP\_SQUARE\_ROPE = '&';

const char MAP\_SQUARE\_ROPE\_TIED = '~';

const char MAP\_SQUARE\_SLINGSHOT = 'Y';

const char MAP\_SQUARE\_PLAYER\_START = 'S';

const int MAP\_MAX\_WIDTH\_HEIGHT = 255;

/\*

\* MAP STATE

\*/

string MAP;

int MAP\_WIDTH;

int MAP\_HEIGHT;

/\*

\* FILE CONSTANTS

\*/

const int FILE\_INVENTORY\_LENGTH = 10;

const char FILE\_INVENTORY\_ITEM\_EMPTY = '\0';

const int FILE\_INVENTORY\_VALUE\_EMPTY = 0;

/\*

\* FILE STATE

\*/

int FILE\_START\_LOCATION\_AS\_INDEX;

char FILE\_START\_LOOKING\_DIRECTION;

char FILE\_INVENTORY\_ITEMS[FILE\_INVENTORY\_LENGTH];

int FILE\_INVENTORY\_VALUES[FILE\_INVENTORY\_LENGTH];

/\*

\* FUNCTION PROTOTYPES

\*/

void printMapRow(const int, const int, const int, const int, const int, const char);

char getMapSquare(const int, const int);

bool setMapSquare(const int, const int, const char, const char);

bool clearMapSquare(const int, const int, const char);

bool isOpenSpace(const int, const int);

bool loadMapFile(string);

bool loadGame(string fileName);

bool saveGame(string fileName, const int, const int, const char, const char[], const int[], const int);

/\*

\* FUNCTION IMPLEMENTATIONS

\*/

char getMapSquare(const int x, const int y)

{

char result = MAP\_SQUARE\_ROCK;

if (x >= 0 && x < MAP\_WIDTH && y >= 0 && y < MAP\_HEIGHT)

{

result = MAP[x + y \* MAP\_WIDTH]; //Hiroya Gojo

}

return result;

}

bool setMapSquare(const int x, const int y, const char currentValue, const char newValue)

{

bool result = false;

if (x >= 0 && x < MAP\_WIDTH &&

y >= 0 && y < MAP\_HEIGHT &&

getMapSquare(x, y) == currentValue)

{

MAP[x + y \* MAP\_WIDTH] = newValue; //Hiroya Gojo

result = true;

}

return result;

}

bool clearMapSquare(const int x, const int y, const char currentValue)

{

return setMapSquare(x, y, currentValue, MAP\_SQUARE\_EMPTY);

}

bool isOpenSpace(const int x, const int y)

{

char mapSquare = getMapSquare(x, y);

return (mapSquare == MAP\_SQUARE\_EMPTY || mapSquare == MAP\_SQUARE\_PLANK\_SET || mapSquare == MAP\_SQUARE\_ROPE\_TIED); //Hiroya Gojo

}

void printMapRow(const int centerX, const int centerY, const int rowOffset, const int screenWidth, const int screenRadius, const char playerChar)

{

const int row = centerY - screenRadius + rowOffset;

for (int col = centerX - screenRadius; col <= centerX + screenRadius; col++)

{

if (row == centerY && col == centerX)

{

cout << playerChar;

}

else if (row >= 0 && row < MAP\_HEIGHT && col >= 0 && col < MAP\_WIDTH)

{

cout << MAP[row \* MAP\_WIDTH + col];

}

else

{

cout << MAP\_SQUARE\_ROCK; //Hiroya Gojo

}

}

}

void openFile(ifstream& file, const char\* fileName)

{

//Monty Choy & Hiroya Gojo

file.open(fileName);

}

void openFile(ofstream& file, const char\* fileName)

{

//Monty Choy & Hiroya Gojo

file.open(fileName);

}

bool loadGame(string fileName)

{

//Monty Choy & Hiroya Gojo

bool success = false;

ifstream file;

openFile(file, fileName.c\_str());

if (!file.fail())

{

int mapWidth, mapHeight, playerLocationAsIndex;

char lookingDirection;

file >> mapWidth >> mapHeight >> playerLocationAsIndex >> lookingDirection;

file.ignore(200, '\n');

if (mapWidth > 0 && mapHeight > 0)

{

const int charsToWriteIncludingNullTerminator = mapWidth + 1;

char buffer[MAP\_MAX\_WIDTH\_HEIGHT + 1];

string mapText = "";

for (int i = 0; i < MAP\_MAX\_WIDTH\_HEIGHT + 1; i++)

{

buffer[i] = '\0';

}

//load map in file into mapText

for (int i = 0; i < mapWidth \* mapHeight; i++)

{

file.get(buffer[i]);

if (buffer[i] == '\n')

{

i--;

continue;

}

if (file.fail())

{

break;

}

else if (buffer[0] == '\0')

{

i--;

continue;

}

mapText = buffer;

}

//Debugging

//cout << mapWidth << endl << mapHeight << endl << playerLocationAsIndex << endl << lookingDirection << endl << buffer;

if (mapText.length() == mapWidth \* mapHeight)

{

MAP = mapText;

MAP\_WIDTH = mapWidth;

MAP\_HEIGHT = mapHeight;

FILE\_START\_LOCATION\_AS\_INDEX = playerLocationAsIndex;

FILE\_START\_LOOKING\_DIRECTION = lookingDirection;

if (!file.fail())

{

char item;

int value;

for (int i = 0; i < FILE\_INVENTORY\_LENGTH; i++)

{

if (file.fail())

{

FILE\_INVENTORY\_ITEMS[i] = FILE\_INVENTORY\_ITEM\_EMPTY;

FILE\_INVENTORY\_VALUES[i] = FILE\_INVENTORY\_VALUE\_EMPTY;

continue;

}

file >> item;

if (file.fail())

{

i--;

continue;

}

file >> value;

if (file.fail())

{

i--;

continue;

}

FILE\_INVENTORY\_ITEMS[i] = item;

FILE\_INVENTORY\_VALUES[i] = value;

}

}

success = true;

} //if

} //if

} //if !file.fail()

file.close();

return success;

}

bool saveGame(string fileName, const int playerX, const int playerY, const char lookingDirection, const char inventoryItems[], const int inventoryValues[], const int inventoryLength)

{

bool success = false;

ofstream file;

openFile(file, fileName.c\_str());

if (!file.fail())

{

// write map dimensions and player location

int playerSymbolIndex = playerX + playerY \* MAP\_WIDTH; //Hiroya Gojo

file << MAP\_WIDTH << ' ' << MAP\_HEIGHT << ' ' << playerSymbolIndex << ' ' << lookingDirection << endl;

// write map

for (int i = 0; i < MAP\_HEIGHT; i++)

{

for (int j = 0; j < MAP\_WIDTH; j++)

{

file << MAP[j + i \* MAP\_WIDTH]; //Hiroya Gojo

}

file << endl;

}

// write inventory

for (int i = 0; i < inventoryLength; i++)

{

file << inventoryItems[i] << ' ' << inventoryValues[i] << endl;

}

success = !file.fail();

}

file.close();

return success;

}

void loadDefaultMap()

{

//Hiroya Gojo

for (int i = 0; i < FILE\_INVENTORY\_LENGTH; i++)

{

FILE\_INVENTORY\_ITEMS[0] = FILE\_INVENTORY\_ITEM\_EMPTY;

FILE\_INVENTORY\_VALUES[0] = FILE\_INVENTORY\_VALUE\_EMPTY;

}

MAP = "&.@:= OYO k";

MAP\_WIDTH = 5;

MAP\_HEIGHT = 3;

FILE\_START\_LOCATION\_AS\_INDEX = 7;

FILE\_START\_LOOKING\_DIRECTION = '>';

}

// DO NOT CHANGE OR REMOVE THE FOLLOWING LINE

#endif

// DO NOT CHANGE OR REMOVE THE PRECEDING LINE