```
#include <iostream>
#include <vector>
#include <string>
#include <fstream>
#include <cassert>

using namespace std;


vector<string> getWordsFromLine(string line)
{
	vector<string> parts;

	int prevPos = 0;
	int spacePos = line.find(" ", 0);

	string word;

	while (spacePos != std::string::npos)
	{
		word = line.substr(prevPos, (spacePos-prevPos));

		if (!word.empty()) parts.push_back(word);

		prevPos = spacePos + 1;
		spacePos = line.find(" ", prevPos);
	}

	word = line.substr(prevPos, (spacePos-prevPos));

	if (!word.empty()) parts.push_back(word);

	return parts;
}


void ProcessInventoryAction(vector<string> & parts, vector<string> & playerInv,
vector<string> & roomInv)
{
	if (parts.size() < 2) return;

	string item = "";
	for (int i=1; i < parts.size(); i++)
	{
		item += parts[i];
		if (i < parts.size() - 1)
		{
			item += " ";
		}
	}

	vector<string>::iterator it;

	if (parts[0] == "create")
	{
		cout << "You magically create a " << item << "\n";
		roomInv.push_back(item);
	}

	if (parts[0] == "drop" || parts[0] == "d")
	{
		// cout << "dropping " << parts[1] << endl;
		it = find(playerInv.begin(), playerInv.end(), item);
		if (it == playerInv.end())
		{
			cout << "You don't have a " << item << "\n";
		}
		else
		{
			cout << "You drop the " << item << "\n";
			// remove from inventory
			playerInv.erase(it);

			// add to room
			roomInv.push_back(item);
		}
	}
	if (parts[0] == "take" || parts[0] == "t")
	{
		// cout << "taking " << parts[1] << endl;
		it = find(roomInv.begin(), roomInv.end(), item);
		if (it == roomInv.end())
		{
			cout << "There's no " << item << " here\n";

		}
		else
		{
			cout << "You pick up the " << item << "\n";
```

```
				roomInv.erase(it);
				playerInv.push_back(item);

			}
		}
}

void printVec(vector<string> & v)
{
	for (int i=0; i<v.size(); i++)
	{
		cout << v[i];
		if (i < v.size()-1) cout << ",";
	}
}

void InitRooms(vector<string> & names, vector<vector<string>> & contents)
{
	ifstream roomFile("rooms.txt");
	assert(roomFile);

	string line;
	while (getline(roomFile, line))
	{
		vector<string> parts = getWordsFromLine(line);

		if (parts[0] == "room:")
		{
			names.push_back(parts[1]);
		}
		else
		{
			contents.push_back(parts);
		}


	}

	roomFile.close();

}

int AttemptMove(int curr, vector<string> & parts, vector<string> & roomNames)
{
	for (int i=0; i < roomNames.size(); i++)
	{
		if (roomNames[i] == parts[0])
		{
			cout << "You move to the " << roomNames[i] << "\n";
			return i;
		}

	}

	return curr;
}

int main()
{
	bool IsActive = true;

	vector<string> inventory;
	vector<string> roomInv;

	vector<string> roomNames;
	vector<vector<string>> rooms;

	InitRooms(roomNames, rooms);

	string input;

	int currRoom = 0;

	cout << "Welcome to an adventure. There are places you can go: ";
	printVec(roomNames);
	cout << "\n";

	while (IsActive)
	{
		cout << "You are in the " << roomNames[currRoom] << "\n";
		if (roomInv.size() > 0)
		{
			cout << "There are some things here: ";
			printVec(rooms[currRoom]);
			cout << "\n";
		}
		if (inventory.size() > 0)
		{
			cout << "You have: ";
```

```
            printVec(inventory);
            cout << "\n";
        }

        // output the current room
        cout << "->";

        // accept an input
        getline(cin, input);

        vector<string> parts = getWordsFromLine(input);
        if (parts[0] == "quit" || parts[0] == "q")
        {
            IsActive = false;
        }

        currRoom = AttemptMove(currRoom, parts, roomNames);

        ProcessInventoryAction(parts, inventory, rooms[currRoom]);
    }



    return 0;
}
```