Assignment III

(TAKE HOME part submission deadline: 28 Aug, 11:59pm, DomJudge)

Lab Submission Tasks (Friday 22 Aug, submission by 10:45 am)

1. Location class encapsulated x, y, z coordinates. Each of the coordinates can take integer values from 0 to 100. A location can have an 'alias', e.g. I am going 'home', here 'home' is the alias, which means the longitude latitude of your home location without explicitly mentioning them. Lastly, a location can be either free or occupied. One flag is used to denote the status.
    a. Location.cpp has some initial code. Complete the class logic. **Do not using any other library than iostream and C++ string.**
    b. You should be able to compile location.cpp using g++ -c without error.
    c. Divide the location class into location.h and location.cpp. Include location.h in location.cpp and in main.cpp. Create two location objects inside main():
        i. ***LocOne*** with 10, 20, 40 (all ints), alias 'intLoc'
        ii. ***LocAnother*** with "13, 14, 15" (full string), alias 'strLoc'
        iii. Your main program should validate the objects by calling their methods, in the following sequence, they should work:
            1. ***LocOne.***isItMe('intLoc')
            2. LocOne.isItMe('desert')
            3. LocAnother.getX()
            4. LocAnother.getY()
            5. LocAnother.getZ()
            6. LocOne.isOccupied()
            7. LocOne.setLoc("100, 2, 2")
            8. LocOne.getX()
            9. LocOne.getY()
            10. LocOne.getZ()
2. Take Drone description from Assignment I, and refer to drone.cpp for its coding
    a. Inlcude "location.h". Drone saves its location coordinates in a member of ***Location*** type (pointer, or reference or object – that is your choice as per your understanding of Assignment I). Same applies to last location.
    b. Complete drone.cpp,
    c. compile using g++ -c, drone.cpp should compile without error
3. Split drone.cpp into interface and implementation:
    a. Drone.h as interface, having only the class declaration
    b. Drone.cpp should have definitions of class methods
    c. #include "drone.h" in main.cpp. Now compile location.cpp, main.cpp and drone.cpp together, there should not be any error

d. In main() in main.cpp, create one drone “drone1234567” with location **LocOne**. Then call the following drone methods sequentially, they should work:
    i. hasID(“randomID0192”)
    ii. printID()
    iii. showBattery()
    iv. showLoc()

TAKE HOME assignment

4. Complete the remaining logic of Drone by taking code from your C assignment.
5. Create class DroneManager,
    a. who will have an array of 10 drones in max.
    b. In main(). Have only one instance of DroneManager class.
    c. Your command parsing logic should be inside the DroneManager class. E.g. SHOW_BATTERY <drone_ID>: the drone manager object should take the command string, parse it to get the command string and the parameter, and find out the drone object from its array, and call showBattery() on the drone object to show the battery.
    d. Similarly, all the existing drone commands should be processed through drone manager.
    e. Create .h and .cpp for DroneManager.

Your program should take command line arguments as earlier.

Note** Important update on the MOVE command:

Updated Command	**Output**	**Meaning**
MOVE <drone-ID> <x,y,z>	true	Was able to move, even if it was a different location, and avoided collision
	false	Either did not move because of insufficient charge, or moved and collided with other drone