**Destroyer Ship Simulator**

**Introduction**

There is a Destroyer Ship named INS Kolkata, you are the operator of this ship and your task is to provide the operational picture of the battlefield. There can be 3 categories of targets (friendly and foe) namely air, surface and subsurface. Targets can have different identities namely friendly, hostile and neutral.

There are times when the sensors (radars and sonars) don’t work due to some technical difficulties, at such times making a strategic decision becomes very important. Although the ship sensors are down, you have access of secret intelligence of the positions of the targets, you as an operator must take control of situation and provide a combat view of battlefield to make strategic decisions.

You can get the secret intelligence from various sources, at times it may happen that the two (or more) targets reported by the sources can be the same, at such an event the operator has special function to correlate the targets, as they are one.

But over time the two targets which are thought to be same, prove to be not the same as per the reports. The situation where such an event can arise are:

1. The two targets have different category.
2. The two targets have different identity.
3. The two targets are distantly spaced from each other (above a threshold distance).

The ship has a specific range of view.

Targets can drift away from the field of view, and the targets inside the radius of view are of interest.

Extrapolate targets between 1 second interval, using formula distance = speed \* time.

You can perform the following actions:

1. Create a target: You can create air, surface and subsurface targets. The speed of air ranges from 50 – 200 KMPH, surface is 0 – 50 KMPH, subsurface is 0 – 30 KMPH.
2. Update a target: You can update the targets speed, identity and category as per the secret intelligence you get over time.
3. Delete a target: You can update the target, if the target is falsely created or is being destroyed in battle.
4. Correlate target’s: You can correlate(bind) the targets if found to be the same target reported from two different sources.
5. Give a summary: You can stop the simulation and give the summary of all the targets in the current combat view of the ship.

**Goals**

This lab covers the following C++ concepts:

* Smart Pointers
* Thread Synchronization
* Move Semantics
* Constructor Delegation
* Vector

**Documentation**

Refer the documentation [here](html/classes.html)

**Implementation**

The starter code is in a zip file here.

Here is a quick description of the provided src directory.

* CMakeLists.txt: CLion CMake project file.
* Simulator.h: Simulator header file.
* Simulator.cpp: Simulator source file.
* Track.h: Track header file.
* Track.cpp: Track source file.
* main.cpp: The main program.

If you are using CLion, you should maintain the src directory with all the files inside, except for CMakeLists.txt which should be at the top level. Right click on the src directory and select Mark Directory As -> Project Sources and Headers.

**Implementation Details**

* Do not change any of the provided files except for Simulator.cpp, Track.cpp and main.cpp.
* You will implement all the source files.
* There should be no dynamic memory directly allocated by your code.
* There should be no memory leak.

**Simulator**

Read over the documentation for this class and implement it accordingly. You must use the track\_store thread safely, as multiple threads are accessing it.

**Track**

Read over the documentation for this class and implement it accordingly.

**Main**

When a track is successfully created it should be added to the track\_store. When a track is successfully deleted it should be removed from the track\_store and the resource be released.

The Operator can create three type of tracks.

* Air – Airborne targets, can be either neutral/friendly/hostile.
* Surface – Surface targets, can be either neutral/friendly/hostile.
* Subsurface – Subsurface targets, can be either neutral/friendly/hostile.

Here is a full list of commands that the main loop must support.

help

* Toggle help of simulator.

create

* Create a track with specified category, identity and speed.

update

* Update a track with specified category, identity and speed.

delete

* Delete a track.

correlate

* Associate two specific tracks.

stop

* Stop the simulation

**Errors**

Your program should be able to handle the following errors:

* Create : When passed invalid parameters
  + The tracks will be created with the following valid parameters:
  + For Category: air/surface/subsurface
    - Sample command – create air friendly 100
  + For Identity: neutral/friendly/hostile
    - Sample command – update 1 surface hostile 33
  + Anything **except for the above keywords** should throw error string “Invalid category and/or identity type”.
* Update(When the track is not existing) : Track does not exist.
* Delete(When the track is not existing) : Track does not exist.
* Update : When passed invalid parameters
  + For air track the speed must be between 50 and 200 mph.
    - Anything **except for the above range** should throw error string “Invalid speed for air target”.
  + For surface track the speed must be between 0 and 50 mph.
    - Anything **except for the above range** should throw error string “Invalid speed for surface target”.
  + For subsurface track the speed must be between 0 and 30 mph.
    - Anything **except for the above range** should throw error string “Invalid speed for subsurface target”.
* Correlate: When passed with track id’s that do not share the same properties
  + Two tracks with different category.
  + Two tracks having different identities.
  + The following error string should be thrown “Cannot correlate as identity and/or category do not match.”

**Sample Runs**

Here are some sample runs for you to help verify things.

$ ./main

run\_01

Constructing simulator

Enter command:

create air friendly 100

Constructing Track

Track 1 created.

with attributes:

Track: 1

category: air

identity: friendly

speed: 100

distance: 0

Enter command:

create surface neutral 44

Constructing Track

Track 2 created.

with attributes:

Track: 2

category: surface

identity: neutral

speed: 44

distance: 0

Enter command:

create subsurface hostile 22

Constructing Track

Track 3 created.

with attributes:

Track: 3

category: subsurface

identity: hostile

speed: 22

distance: 0

Enter command:

create air friend 111

Invalid category and identity type.

Enter command:

create water neutral 11

Invalid category and identity type.

Enter command:

stop

---Summary---

Tracks in the field of view are:

Track: 1

category: air

identity: friendly

speed: 100

distance: 75.0278

Track: 2

category: surface

identity: neutral

speed: 44

distance: 24.64

Track: 3

category: subsurface

identity: hostile

speed: 22

distance: 6.89333

run\_02

Constructing simulator

Enter command:

create air friendly 100

Constructing Track

Track 1 created.

with attributes:

Track: 1

category: air

identity: friendly

speed: 100

distance: 0

Enter command:

create air friendly 112

Constructing Track

Track 2 created.

with attributes:

Track: 2

category: air

identity: friendly

speed: 112

distance: 0

Enter command:

correlate 1 2

Track: 1 correlated with: Track: 2

category: air

identity: friendly

speed: 112

distance: 0.466667

Enter command:

stop

---Summary---

Tracks in the field of view are:

Track: 1

category: air

identity: friendly

speed: 100

distance: 5.27778

run\_03

Constructing simulator

Enter command:

create surface neutral 32

Constructing Track

Track 1 created.

with attributes:

Track: 1

category: surface

identity: neutral

speed: 32

distance: 0

Enter command:

create surface hostile 23

Constructing Track

Track 2 created.

with attributes:

Track: 2

category: surface

identity: hostile

speed: 23

distance: 0

Enter command:

correlate 1 2

Cannot correlate as identity and/or category do not match.

Enter command:

stop

---Summary---

Tracks in the field of view are:

Track: 1

category: surface

identity: neutral

speed: 32

distance: 2.05333

Track: 2

category: surface

identity: hostile

speed: 23

distance: 0.421667

run\_04

Constructing simulator

Enter command:

create air friendly 180

Constructing Track

Track 1 created.

with attributes:

Track: 1

category: air

identity: friendly

speed: 180

distance: 0

Enter command:

create air friendly 120

Constructing Track

Track 2 created.

with attributes:

Track: 2

category: air

identity: friendly

speed: 120

distance: 0

Enter command:

correlate 1 2

Track: 1 correlated with: Track: 2

category: air

identity: friendly

speed: 120

distance: 7

Enter command:

Constructing Track

Track 2 created.

with attributes:

Track: 2

category: air

identity: friendly

speed: 120

distance: 44.2

Track: 1

category: air

identity: friendly

speed: 180

distance: 94.55

is decorrelated with Track: 2

category: air

identity: friendly

speed: 120

distance: 44.2

stop

---Summary---

Tracks in the field of view are:

Track: 1

category: air

identity: friendly

speed: 180

distance: 124.25

Track: 2

category: air

identity: friendly

speed: 120

distance: 46.0333

**Valgrind**

You should use valgrind to verify your program is properly managing memory. It is ok if bytes are still reachable in the heap when the program exits - this is expected. You should have no memory errors and your leak summary should look like this:

$ valgrind --leak-check=full ./main

...

==31715== LEAK SUMMARY:

==31715== definitely lost: 0 bytes in 0 blocks

==31715== indirectly lost: 0 bytes in 0 blocks

==31715== possibly lost: 0 bytes in 0 blocks

==31715== still reachable: 72,704 bytes in 1 blocks

==31715== suppressed: 0 bytes in 0 blocks