**Please note if your codes don’t compile, they will be given 0 mark.**

You will create an application that simulates a game where the player will use canon to destroy enemies’ ships.

1. We will still use a two-dimension array to represent the map, where the size can be set by the user, details will be given later

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

1. Please create an **abstract class WarShip**, which has the following attributes and method: (**3 marks**)

**position**: the ship current position on the graph (two-dimension matrix), **int x, y;**

**status**: the ship status (alive or dead).

**size:** the size of the ship.

**shield**: the shield of the ship, which is the same as size if the shield reduced to 0, the ship will be sunk (dead).

**Deploy** (): it is an abstract method to describe how the ship will be deployed.

**Hint: You need to design the method signature, which is very important.**

1. Please create an interface **Play**, it is used to define the players’ behavior, it has the following abstract method: (**2 marks**)

**CalDistance**() : this method is used to calculate the minimum distance from the spot player’s canon shot and the enemy’s ships.

**Fire**(): this method is used to decide how the players will attack the other’s warship.

1. Please extend the **WarShip** class to create a **Carrier** class by implementing the Deploy() method. (**3 marks**)

**Deploy** (): The Carrier will be deployed at certain position (occupying certain grids on the map).

1. Please extend the **WarShip** class to create a **Submarine** class by implementing the Deploy () method. (**1 marks**)

**Deploy** (): The Submarine will be deployed at certain position (occupying certain grids on the map).

1. Please extend the **WarShip** class to create a **Cruiser** class by implementing the Deploy () method. (**1 marks**)

**Deploy** (): The Submarine will be deployed at certain position (occupying certain grids on the map).

Here is the table summarize the attributes of those war ships:

Table 1. WarShip attributes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Position | Size | Shield | Status |
| Carrier | Random | 3 | 3 | Alive (true) |
| Submarine | Random | 2 | 2 | Alive |
| Cruiser | Random | 2 | 2 | Alive |

1. Please create two other classes **HumanPlayer and CompPlayer** which will implement the Play interface. They both have those attributes (at least) (**25 marks**)

**name**: HumanPlayer’s name will be “Commander” and CompPlayer’s name will be “Comp”;

**score**: when 1 ship is hit, score will be increased by one.

**achievements**: when 1 ship is sunk, the type of the ship will be added here.

**Fire()**: this method will be different for HumanPlayer and CompPlayer.

**For HumanPlayer**: this method will ask for users’ input about the coordinates to

shoot the canon.

**For CompPlayer: this method will randomly generate the coordinates to shoot the canon, however, you need to handle the duplicated coordinates (same x, same y), two attacks should shoot at different spots.**

**CalDistance**(): this method will be different for HumanPlayer and CompPlayer.

**For HumanPlayer**: **this method will calculate the min distance from the shooting spot player selected and the enemy’s ships. You can use the difference in terms of col indices.**

**For CompPlayer:** doing nothing.

1. Please create a Game class which contains the main() method. You might want to create some methods to modularize your codes.
   1. Ask user’s input for the size of the map, min is 6. Use this size to generate a two-dimensional array, int, char or any other type you feel easier.

Create HumanPlayer and CompPlayer objects.

Create two arrays war ships, one for player and one for computer. The size of which is half of the map size (you might need to cast it to int), 1/3 of them are carriers, 1/3 of them are submarines and 1/3 of them are cruisers. Initialize them using Table 1. (**10 marks**)

* 1. Display the map; then ask player the coordinates launch the canon. A message should be displayed to inform the player if or not the canon hit the target and assert if any of the targets are sunk. If the player missed the target please display the shortest distance by invoking the ClacDistance() method. If the player hit any target the corresponding position on the map will be replaced by “H”; if it is a miss, it will be replaced by “M”. (**10 marks**)
  2. The computer player will then randomly attack the human player’s ships. A message should be displayed to inform the player under attack and what the damage is. (**5 marks**)
  3. Cheat mode, after 1 round of the above steps (b and c), the player will be asked to continue or quit, if the user input 999, it will be the cheat mode, where the position of the enemy’s ships will be displayed on the map by changing the \* to “T” except those spots have been changed to “H”;
  4. Simulate the game procedure, repeat step **b to d** until: (**5 marks**)
     1. The player wants to stop; or
     2. The player’s ships were all destroyed; or
     3. All enemy’s war ships were all destroyed.
  5. When the game finishes, please write the summarized results in to the result.txt file, including the number of rounds played, the number of warship sunk, the number of hits and the damage to the player’s ships, you can decide the format. (**5 marks**)

**You can decide any unmentioned details by yourself including but not limited to adding any necessary data attributes or methods, etc.**

Here is the snapshot of two rounds of the game:

Hi, commander, please let me know the size of the map:

10

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

Hi, commander, which position you want to shoot [x,y]:

2,4

Hi, commander, we missed the target. The min distance is 0!

Hi, commander, we are under attack! Our carrier is hit!

Hi, commander, do you want to continue [C] or quit[Q]?

C

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* M \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \*

Hi, commander, which position you want to shoot [x,y]:

0,1

Hi, commander, we hit the target!

Hi, commander, we are under attack! They missed, whoops!

Hi, commander, do you want to continue [C] or quit[Q]?

999

\* H \* \* \* \* \* \* \* \*

\* T \* \* \* \* \* \* \* \*

\* T \* \* M \* \* \* \* \*

\* \* \* \* T \* \* \* \* \*

\* \* \* \* T \* T \* \* \*

\* \* \* \* \* \* T \* \* \*

\* \* \* \* \* \* \* \* \* T

\* \* \* \* \* \* \* \* T T

\* \* \* \* \* \* \* \* T \*

\* \* \* \* \* \* \* \* \* \*