MULTIMEDIA UNIVERSITY

CCP6124:
OBJECT-ORIENTED PROGRAMMING AND DATA STRUCTURES

TRIMESTER:

LECTURE SECTION: TC1L

TUTORIAL SECTION: TT1L

GROUP NUMBER: G05

GROUP LEADER:
MARYAM BINTI NORAZMAN

TASK DISTRIBUTION

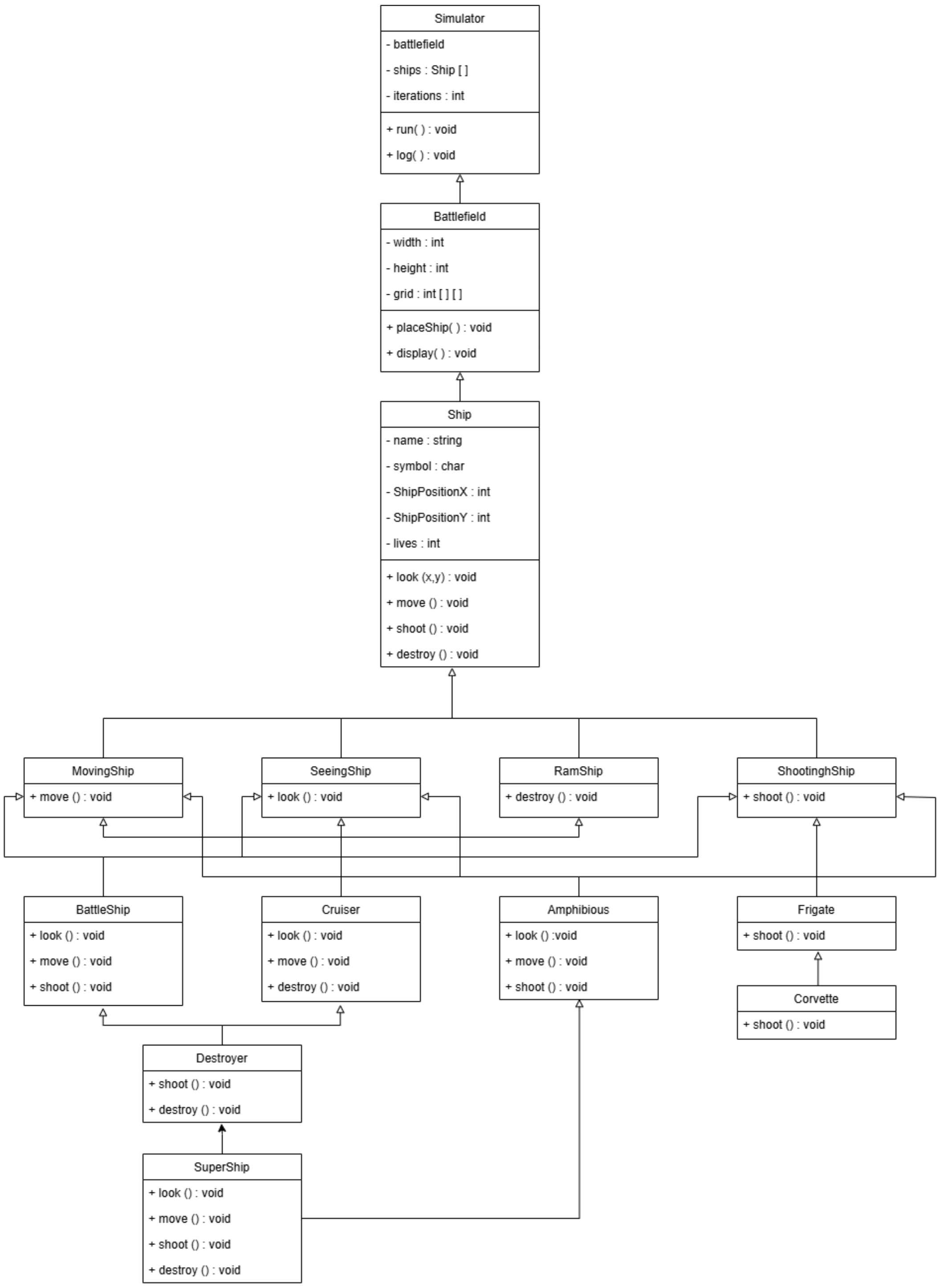
NO.	STUDENT ID	STUDENT NAME	TASK DESCRIPTION	(%)
1	1211111809	Maryam binti Norazman	Simulator, Battlefield Destroyer, MovingShip	
2	241UC240QH	Mohanad Hassan Fathy Abdelatty	Shootingship, Corvette, Supership, Documentation	
3	241UC2415F	Nur Rafiqah Qurratu’aini binti Mohd Seman	Seeingship, Amphibious & Frigate	
4	241UC2400K	Ayuub Yusuf Ibrahim	Battleship, Destroy, Ship, Cruiser	
			TOTAL	100%

TABLE OF CONTENTS

1.0 DESIGN DOCUMENTATION

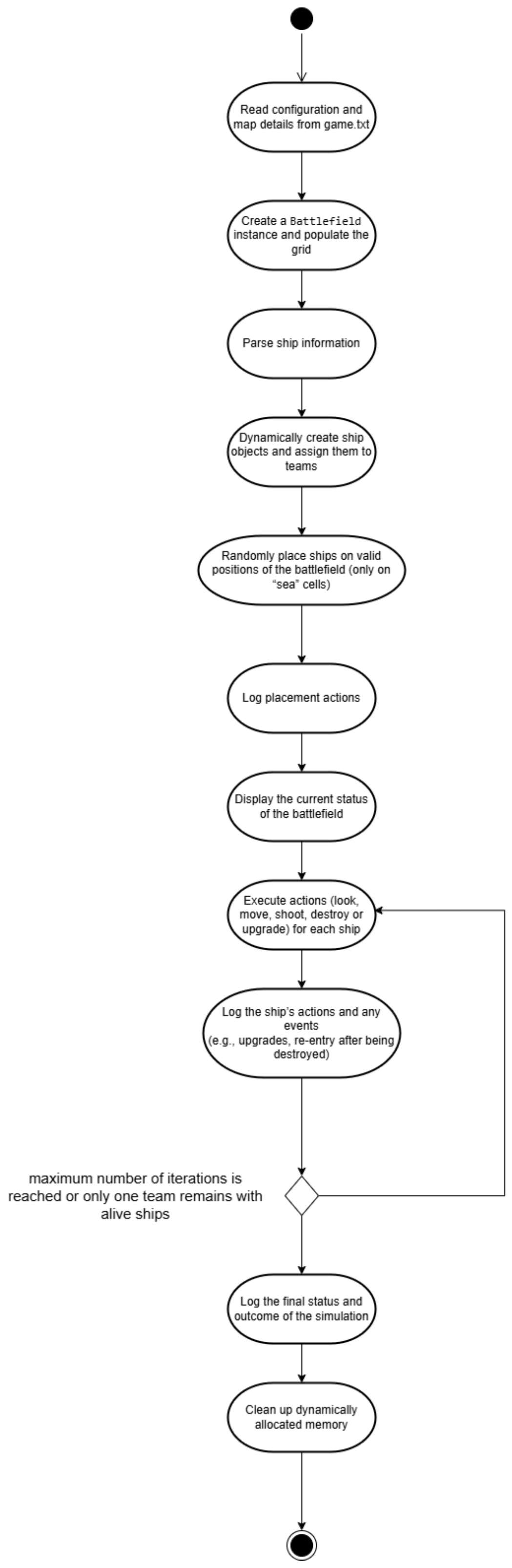
1.1 Class Diagram

Class diagram for the project (classes, attributes, relations)



1.2 Activity Diagram

General program workflow in the form of an activity diagram.


Read configuration and map details from game.txt
Create a Battlefield instance and populate the grid
Parse ship information
Dynamically create ship objects and assign them to teams
Randomly place ships on valid positions of the battlefield (only on "sea" cells)
Log placement actions
Display the current status of the battlefield
Execute actions (look, move, shoot, destroy or upgrade) for each ship
Log the ship's actions and any events (e.g., upgrades, re-entry after being destroyed)
maximum number of iterations is reached or only one team remains with alive ships
Log the final status and outcome of the simulation
Clean up dynamically allocated memory

2.0 INITIALIZATION OF A SIMULATION

2.1 File reading and initial simulation settings (how the file was read and how the initial set of ships was created).

Input File (game.txt):

• Contains the battlefield configuration (grid layout) and ship configuration (teams and ship types).

• The Battlefield constructor reads the file line-by-line, ignoring empty or separator lines.

• Ship data (e.g., team designation, ship type, symbol, count) is parsed in Simulator::loadShips().

Ship Creation and Team Assignment:

• The simulator uses a factory-like function (createShip) to instantiate the appropriate ship subclass based on the input.

• Each ship is assigned a unique symbol and team (e.g., "Team A" or "Team B").

2.2 How the concept of teams is implemented.

Simulation Iterations:

• A fixed number of iterations (e.g., 100 turns) is set.

Container Usage:

• A custom Vector holds the active ship pointers.

• A custom Queue manages ships that have been temporarily destroyed but may be reactivated.

Random Placement:

• Ships are placed on the battlefield using a randomized algorithm that ensures they are only positioned on valid (sea) cells.

3.0 DISPLAY AND LOGGING OF THE STATUS OF THE BATTLEFIELD AT EACH TURN

3.1 How the battlefield is implemented and linked to the ships.

Grid Representation:

The battlefield is implemented as a 2D grid using custom Vector containers. Each cell in the grid holds a value indicating its state: "0" for sea, "1" for island, or a ship's unique symbol if occupied.

Formatted Display:

The display() method prints the grid with uniform spacing (using setw), ensuring that the layout is clear and easy to read.

Turn-by-Turn Updates:

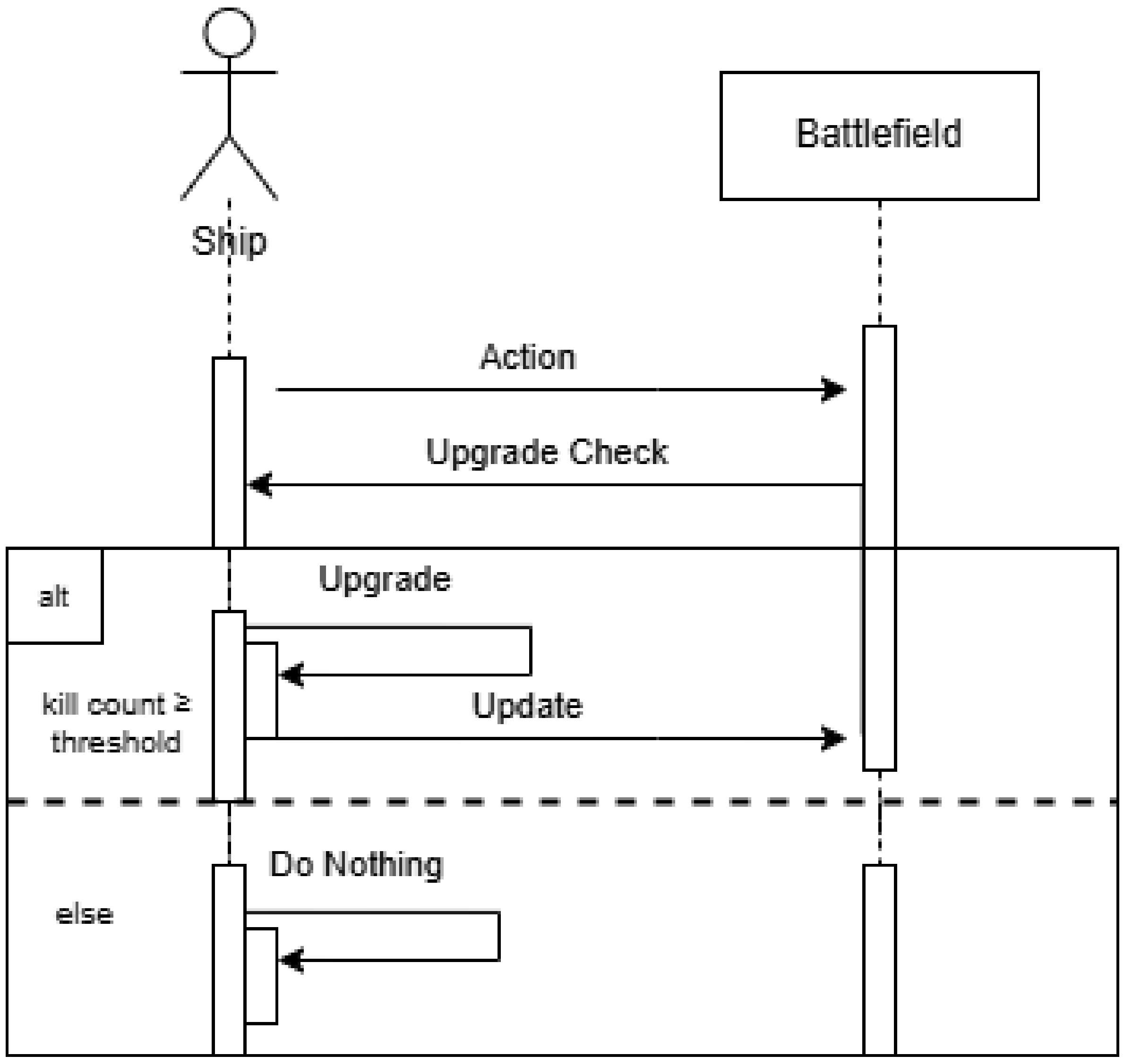
At the start of each turn, the simulation prints a turn header and then calls the display() method to show the current battlefield state. As ships act—moving, attacking, or being destroyed—they update the grid immediately through methods.

Logging Mechanism:

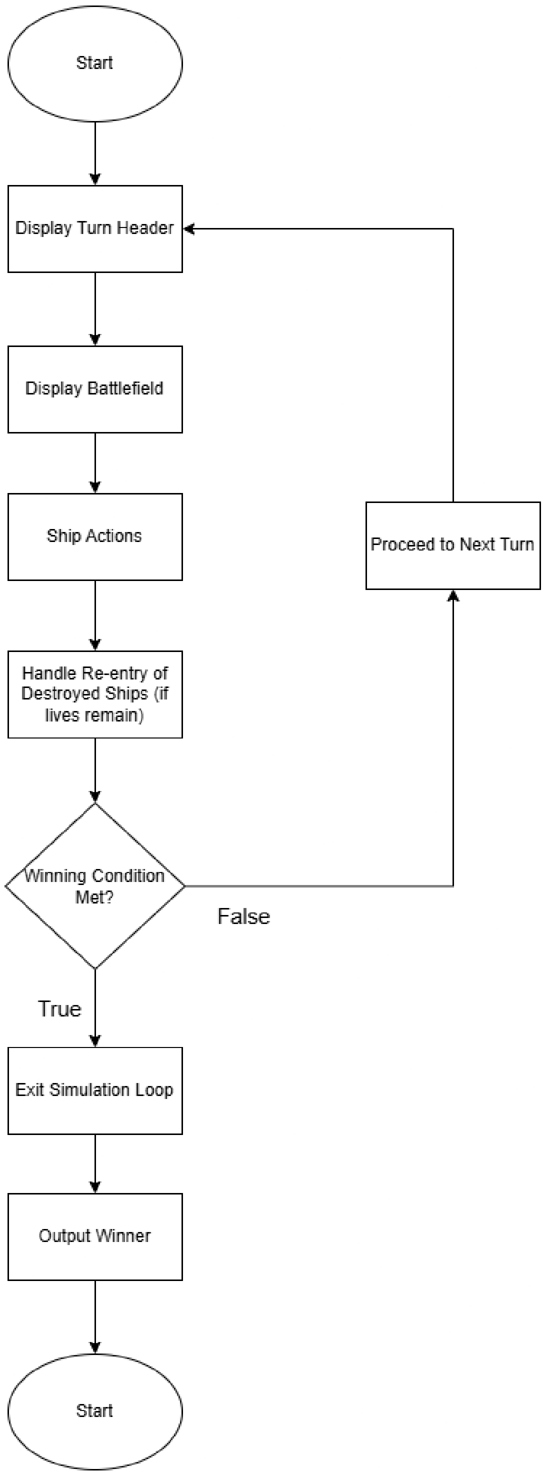
In parallel with the console display, all actions and the updated grid state are logged to an output file. This ensures that every turn is documented, aiding in traceability and debugging.

4.0 DISPLAY AND LOGGING OF THE ACTIONS AND THE STATUS OF EACH SHIP AT EACH TURN

4.1 How the upgrade is done (preferred sequence diagram)



4.2 How the main simulation loop is performed (activity diagram or flowchart).



4.3 Memory management used and how memory leaks are avoided.

Memory management in our program is architected with a focus on deterministic resource release and robust exception safety. We achieve this through a multifaceted strategy that combines the following key elements:

1. **RAII (Resource Acquisition Is Initialization):**
2. Our custom data structure specifically, the Vector and Queue classes—are designed using the RAII paradigm. This ensures that resources such as dynamically allocated memory are acquired during object initialization and automatically released in the destructors when the objecs go out of scope.
3. **Sophisticated Copy/Move Semantics:**
   Both the Vector and Queue classes implement carefully engineered copy and move constructors, as well as assignment operators. These mechanisms guarantee that resource ownership is transferred safely between objects and that any dynamically allocated memory is reliably reclaimed without duplication or leaks.
4. **Explicit Deallocation in the Simulator:**
   The Simulator class, which is central to our application, explicitly manages the lifecycle of ship objects and file streams. It stores ship pointers in our custom Vector and, upon simulation termination or in the event of an exception, iterates over these pointers to delete each ship object. Likewise, open file streams are closed appropriately, ensuring that all allocated resources are explicitly deallocated.
5. **Exception Handling:**
   Our program employs comprehensive exception handling to safeguard against unexpected failures. Even when exceptions occur during file I/O or dynamic memory operations, the destructors for our objects are automatically invoked, ensuring that all resources are systematically released and that memory leaks are avoided.

   This multi-layered approach to memory management not only prevents leaks but also contributes to the overall stability, performance, and scalability of the simulation. By ensuring that every dynamically allocated resource is promptly and predictably reclaimed—whether under normal execution or exceptional conditions—we maintain efficient resource usage throughout the program's lifecycle

5.0 IMPLEMENTATION OF THE REQUIRED SHIP CLASSES WITH OOP CONCEPTS

5.1 Polymorphism in the Assignment

Object-Oriented Programming (OOP) is characterized by polymorphism, which enables various classes to be regarded as instances of the same base class. This assignment uses virtual functions and overriding methods to achieve polymorphism, allowing for dynamic method dispatch at runtime. Examples of polymorphism used in the program are shown below.

Example 1: Virtual Function for Displaying Ship Information

The display function is declared as virtual in the base class Ship, allowing each derived class to override it and provide its own implementation.

```
class Ship {
public:
   virtual void display() const {
      cout << "Ship: " << ShipName_ << ", Symbol: " << symbol
          << ", Team: " << team
          << ", Position: (" << ShipPositionX << ", " << ShipPositionY << "), Lives: " <<
numOfLives_ << endl;
   }
};
```

Overridden in the Battleship class:

```
class Battleship : public SeeingShip, public MovingShip, public ShootingShip {
public:
   virtual void display() const override {
      cout << "Battleship " << ID << " at position (" << ShipPositionX << ", " << ShipPositionY
<< ")" << endl;
   }
};
```

This ensures that calling display() on a Ship* pointer invokes the correct function based on the actual object type

Example 2: Overriding actionMove for Different Ship Types

The actionMove function is defined in the MovingShip class and overridden in the Battleship class to modify movement behavior.

```
class MovingShip : virtual public Ship {
public:
    virtual bool actionMove(Battlefield &battlefield, const string &direction) {
        int newX = ShipPositionX, newY = ShipPositionY;
        if (direction == "up") newY--;
        else if (direction == "down") newY++;
        else if (direction == "left") newX--;
        else if (direction == "right") newX++;
        else return false;

        if (battlefield.isCellEmpty(newX, newY)) {
            battlefield.clearCell(ShipPositionX, ShipPositionY);
            if (battlefield.placeShip(newX, newY, symbol)) {
                ShipPositionX = newX;
                ShipPositionY = newY;
                return true;
            }
        }
        return false;
    }
};
```

Overridden in the Battleship class:

```
class Battleship : public MovingShip {
public:
    virtual bool actionMove(Battlefield &battlefield, const string &direction) override {
        if (numOfKills_ >= 4) {
            cout << "Upgraded Battleship moves aggressively!" << endl;
            return MovingShip::actionMove(battlefield, direction);
        } else {
            cout << "Battleship moving cautiously!" << endl;
            return MovingShip::actionMove(battlefield, direction);
        }
    }
};
```

This illustrates runtime polymorphism, ensuring different behaviour based on the ship type.

Example 3: Dynamic Dispatch Using dynamic_cast

```
for (auto ship : ships) {
    if (auto movingShip = dynamic_cast<MovingShip*>(ship)) {
        movingShip->actionMove(battlefield, "up");
    }
    if (auto shootingShip = dynamic_cast<ShootingShip*>(ship)) {
        shootingShip->actionShoot(battlefield, ships, 1, 0);
    }
}
```

This allows the program to handle different ship types dynamically at runtime.

Explain how polymorphism is used in the assignment (sample codes from the program - not to exceed 10 lines for each example)

6.0 THE ALGORITHMS USED TO OPTIMIZE THE ACTIONS OF THE SHIPS

How the Ships Act Smartly in Battle :
The ships in the game use a smart action plan for fighting and moving better. The actions are optimized using a set of rules and strategies. Explained in detail below is how ships make decisions and act.

1. How Ships Decide Their Actions

Every Ship follows a step-by-step, decision process according to its capabilities, ordered in the form of an action prioritization by which ships will always complete the most relevant action first.

Action Prioritization List (From the Most Important One to the Weakest One)

1. Look around → Here, the space Ship will explore its surroundings searching for an opponent.
2. Attack → With an opponent available, the attack is triggered from the ship.
- If it is a Shooting Ship, it shall shoot from a distance.
- If it is a Ram Ship, it shall try to collide with the enemy.
3. Move → If no attack is possible, the ship will move to a better position.

Explanation:
- Battleship will first look for enemies.
- If it detects one, then it will attack an enemy.
If no enemies around, then move to some other place.

2. Ships Intelligent Movement

Ships do not make random moves; they check before moving that a way ahead must be clear.

Rules of Movement
- Ships only move only if the new position is free.
- If their preferred direction is blocked, they will try another direction.
- In case they are unable to move, they are going to remain in place until the next move.

What If the Ship Cannot Attack?
- Ram Ships will then move randomly looking for an enemy to attack.
- The Shooting Ships would reposition ate themselves to have a better angle.

Example:
- If a Destroyer wants to go forward but another ship is there, it will try left or right instead.

3. How Ships Attack Their Enemies

Ships do not attack randomly. They follow a smart attack strategy.

Attack Strategy

1. Find the Nearest Enemy → The ship looks for the closest target.
2. Attack the Most Important Enemy First →

- If an enemy is right next to the ship, a Ram Ship will ram into it.
- If the enemy is far away, a Shooting Ship will shoot at it.

3. If No Enemies Are Nearby → The ship moves to a better position to attack.

Example:
- A Cruiser sees an opponent right next to it, immediately closes in for ramming, - an opposing ship that is close to its neighbours might just fire right into that direction instead of course and speed changing. The above rules come up again here

4. When the Game Ends (Winner Detection)

The game ends when there is one losing team

Team Loses its ships.

- The game checks if Team A has any ships left.
- Then it checks if Team B has any ships left.
- If only one team still has ships, that team wins the battle.

Example:
- If all of Team A's ships are destroyed, Team B wins automatically.

5. How Destroyed Ships Return to the Battle

To make the game fun and fair, some of the ships can respawn after being destroyed.

How Ships Re-enter the Game

- If a ship is destroyed but it still has lives left, it will respawn.
- The ship will reappear in a random location on the battlefield.
- This prevents one team from losing too quickly.

Example:

- Battleship goes down but has 1 extra life. It respawns in a new location and continues the battle.

6.1 Explain how inheritance is used in the assignment (sample codes from the program - not to exceed 10 lines for each example)

The game can create multiple types of ships with the help of inheritance. In addition to sharing characteristics, all ships have unique capabilities and originated from the core Ship class.

1. The Main Ship Class (Parent Class)

The Ship class is the parent class for all ships. It has basic features like:

- Position (X, Y)
- Team (Team A or Team B)
- Lives (How many times a ship can be destroyed)

Example of the Ship Class

```
class Ship {
protected:
    string symbol;
    int ShipPositionX, ShipPositionY;
    string ID, team;
    int numOfLives_ = 3;
public:
    Ship(string id = "", int x = 0, int y = 0) : ID(id), ShipPositionX(x), ShipPositionY(y) {}
    virtual ~Ship() = default; // Helps avoid errors in child classes
};
```

All ships have these common features.

Other ship types will inherit from this class.

2. Child Classes (Ships with Special Abilities)

Different types of ships inherit from the Ship class and get extra abilities.

A. Seeing Ship (Can Look Around)

This ship can scan the battlefield to find enemies.

```
class SeeingShip : virtual public Ship {
public:
    virtual void actionLook(const Battlefield &battlefield) const {
        cout << ShipName_ << " is looking around." << endl;
    }
};
```

Inherits from Ship → Still has position, team, and lives.

Adds a new function → actionLook() to scan for enemies.

B. Moving Ship (Can Move)

This ship can change its position on the battlefield.

```
class MovingShip : virtual public Ship {
public:
    virtual void actionMove(Battlefield &battlefield, string direction) {
        cout << ShipName_ << " is moving " << direction << "." << endl;
    }
};
```

Inherits from Ship → Still has position, team, and lives.

Adds a new function → actionMove() to move in different directions.

C. Shooting Ship (Can Shoot)

This ship can attack enemies from a distance.

```
class ShootingShip : virtual public Ship {
public:
    virtual void actionShoot(Battlefield &battlefield, int x, int y) {
        cout << ShipName_ << " is shooting at (" << x << ", " << y << ")." << endl;
    }
};
```

Inherits from Ship → Still has position, team, and lives.

Adds a new function → actionShoot() to attack enemies.

D. Ram Ship (Can Crash into Enemies)

This ship destroys enemies by ramming into them.

```
class RamShip : public MovingShip {
public:
    virtual void actionDestroy(Battlefield &battlefield) {
        cout << ShipName_ << " is ramming into an enemy!" << endl;
    }
};
```

Inherits from MovingShip → Still has movement ability.

Adds a new function → actionDestroy() to crash into enemies.

3. Combining Abilities (Multiple Inheritance)

Some ships need more than one ability. They inherit from multiple classes.

A. Battleship (Can Look, Move, and Shoot)

A battleship can look around, move, and shoot.

```
class Battleship : public SeeingShip, public MovingShip, public ShootingShip {
public:
   Battleship(string name, string symbol, string team, int lives = 3)
       : SeeingShip(name, symbol, team, lives),
         MovingShip(name, symbol, team, lives),
         ShootingShip(name, symbol, team, lives) {}
};
```

Inherits from three classes → SeeingShip, MovingShip, ShootingShip.

Can scan, move, and attack.

4. Avoiding Problems with Virtual Inheritance

When a ship inherits from multiple classes, the Ship class might get copied twice. To prevent this, the program uses virtual inheritance:

```
class SeeingShip : virtual public Ship {};
class MovingShip : virtual public Ship {};
class ShootingShip : virtual public Ship {};
```

Prevents duplicate copies of the Ship class.

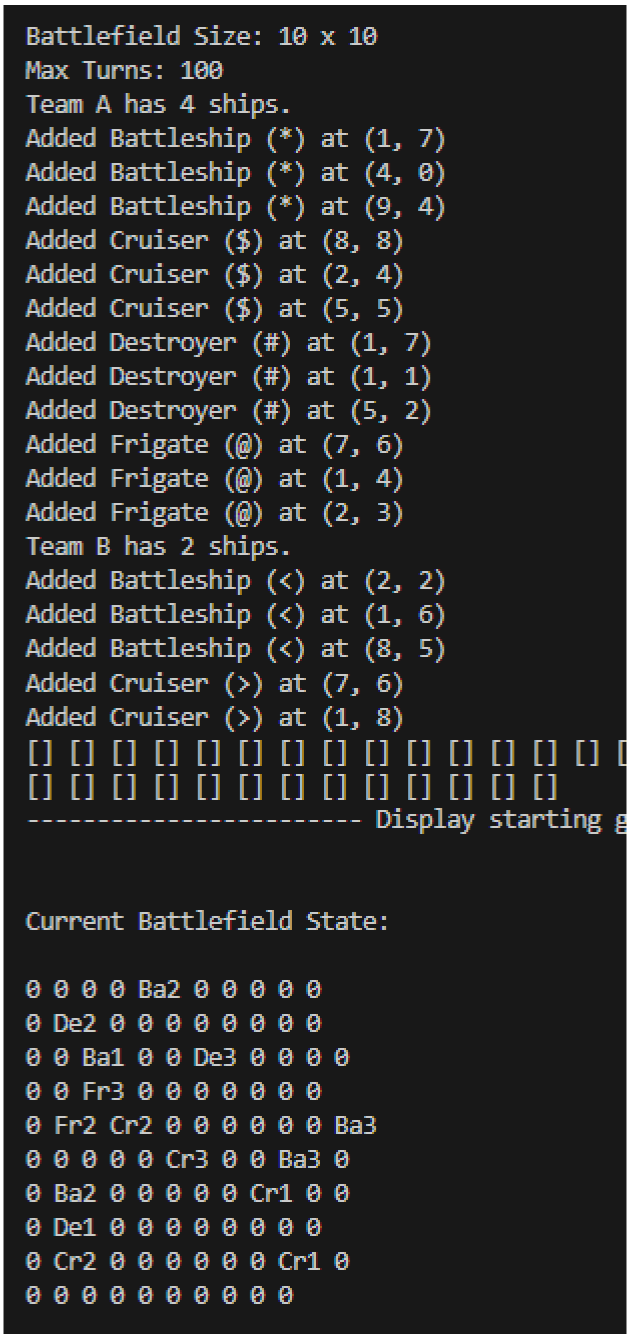
Keeps the program clean and organized.

7.0 DESIGN AND IMPLEMENTATION OF DATA STRUCTURES (LINKED LIST AND QUEUES)

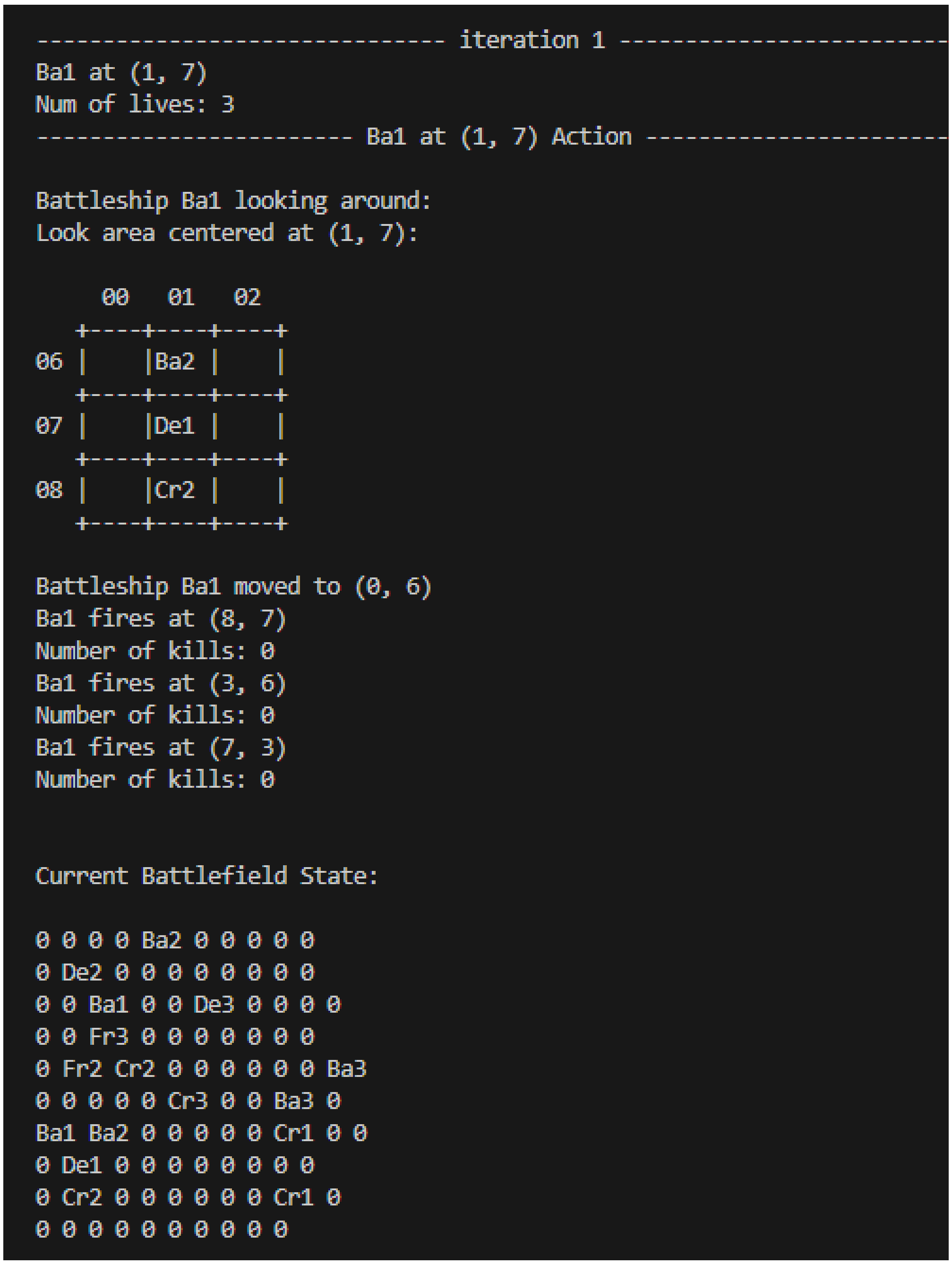
7.1

At least 3 sample ".txt" files for testing your assignment

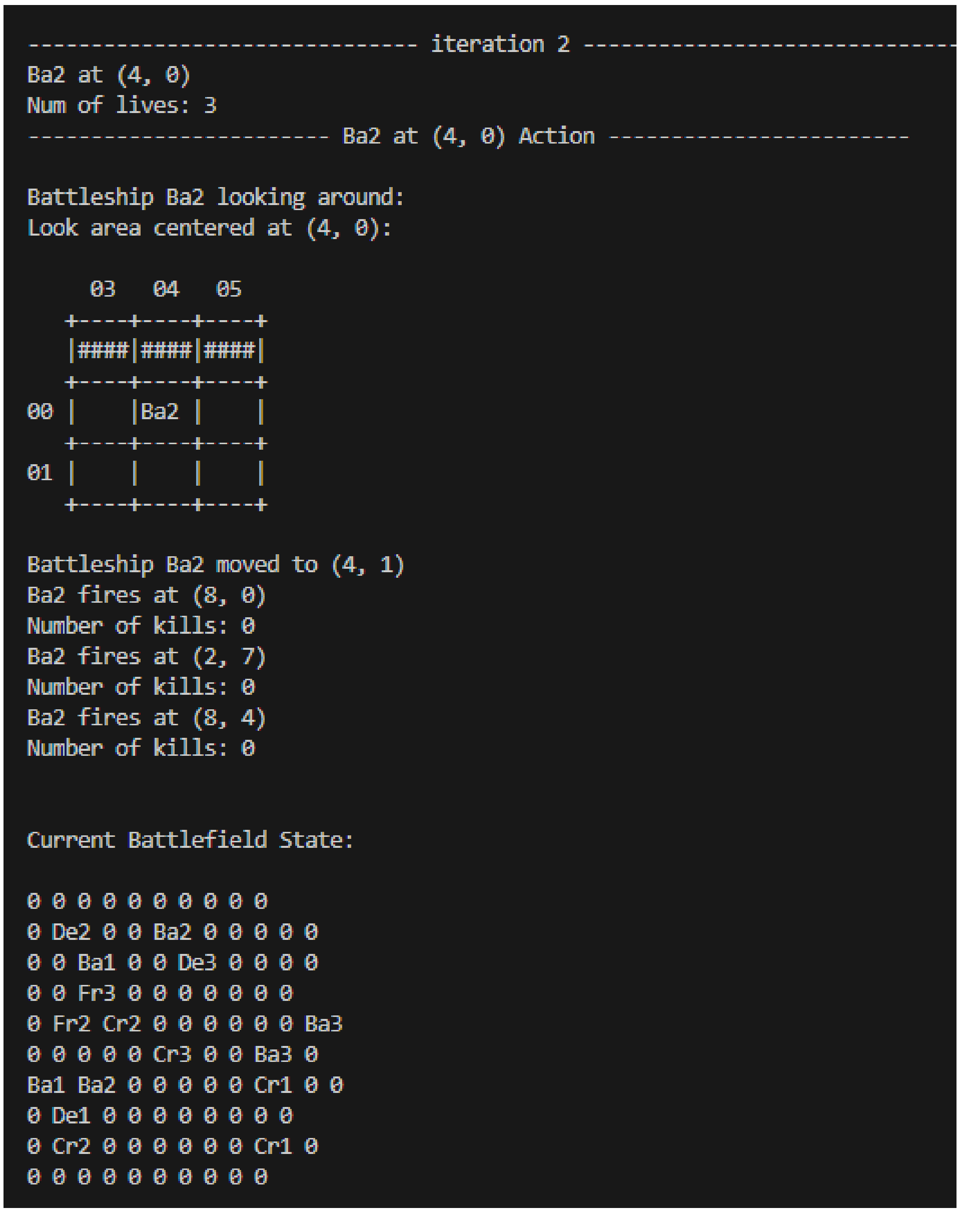
Screenshots of the first 10 turns of each run using the 3 test text files.
First_Test:



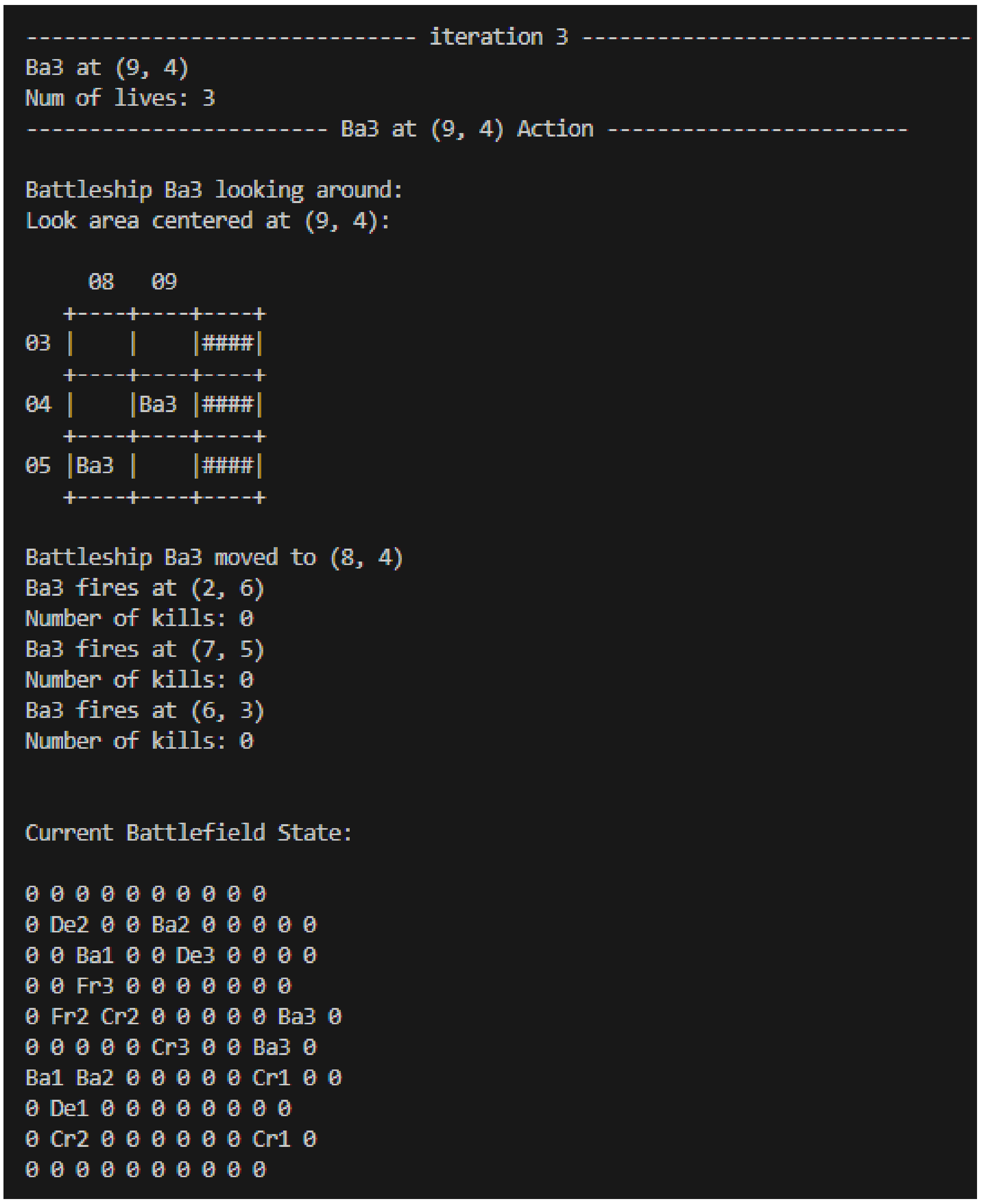
```
Battlefield Size: 10 x 10
Max Turns: 100
Team A has 4 ships.
Added Battleship (*) at (1, 7)
Added Battleship (*) at (4, 0)
Added Battleship (*) at (9, 4)
Added Cruiser ($) at (8, 8)
Added Cruiser ($) at (2, 4)
Added Cruiser ($) at (5, 5)
Added Destroyer (#) at (1, 7)
Added Destroyer (#) at (1, 1)
Added Destroyer (#) at (5, 2)
Added Frigate (@) at (7, 6)
Added Frigate (@) at (1, 4)
Added Frigate (@) at (2, 3)
Team B has 2 ships.
Added Battleship (<) at (2, 2)
Added Battleship (<) at (1, 6)
Added Battleship (<) at (8, 5)
Added Cruiser (>) at (7, 6)
Added Cruiser (>) at (1, 8)
[] [] [] [] [] [] [] [] [] [] [] [] [] [] [
[] [] [] [] [] [] [] [] [] [] [] [] []
------------------------ Display starting g


Current Battlefield State:

0 0 0 0 Ba2 0 0 0 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 0 Ba3
0 0 0 0 0 Cr3 0 0 Ba3 0
0 Ba2 0 0 0 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 0 Cr1 0
0 0 0 0 0 0 0 0 0 0
```



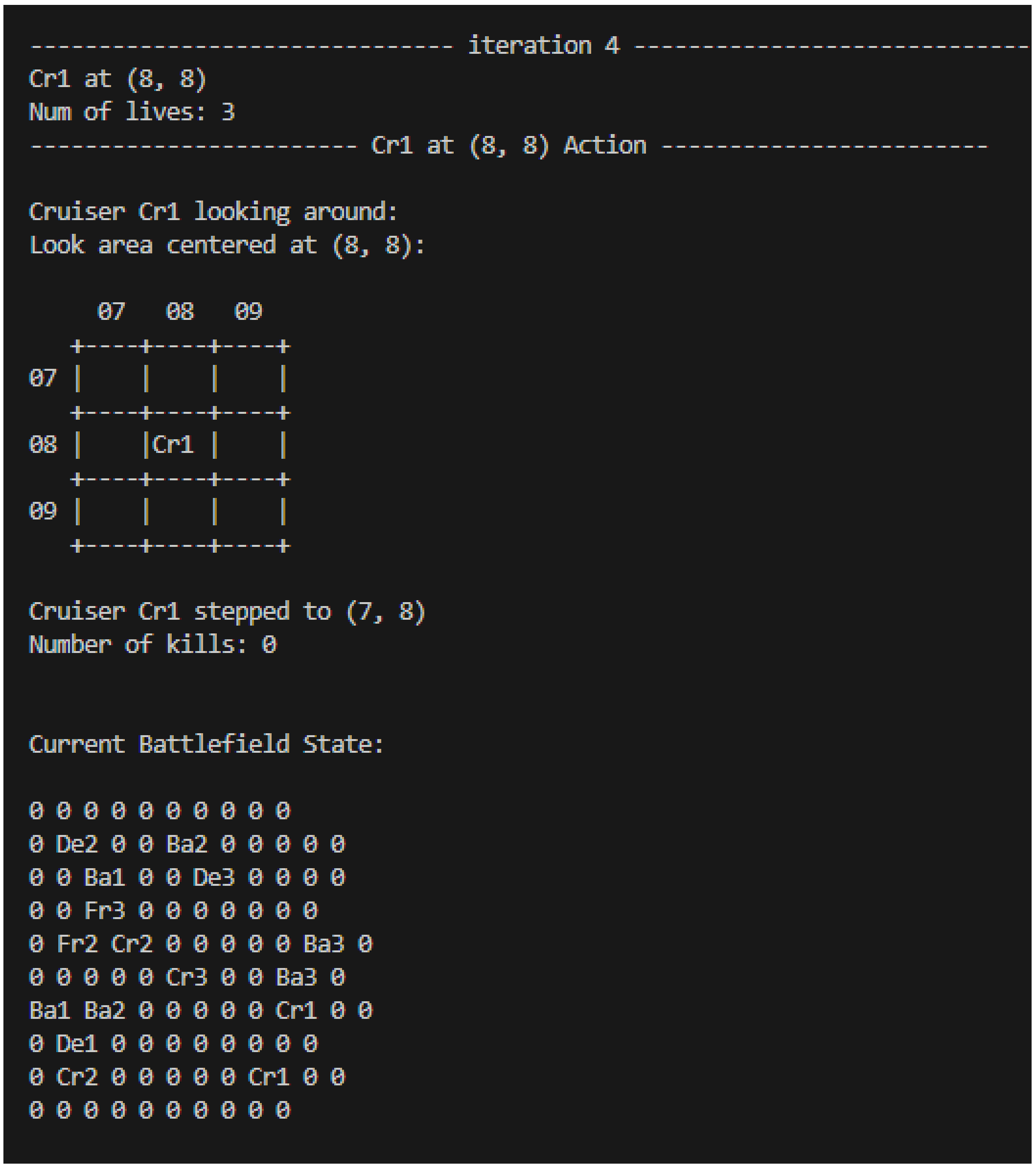
```
 ------------------------------- iteration 1 -------------------------
 Ba1 at (1, 7)
 Num of lives: 3
 ------------------------ Ba1 at (1, 7) Action -----------------------

 Battleship Ba1 looking around:
 Look area centered at (1, 7):

      00   01   02
    +----+----+----+
 06 |    |Ba2 |    |
    +----+----+----+
 07 |    |De1 |    |
    +----+----+----+
 08 |    |Cr2 |    |
    +----+----+----+

 Battleship Ba1 moved to (0, 6)
 Ba1 fires at (8, 7)
 Number of kills: 0
 Ba1 fires at (3, 6)
 Number of kills: 0
 Ba1 fires at (7, 3)
 Number of kills: 0


 Current Battlefield State:

 0 0 0 0 Ba2 0 0 0 0 0
 0 De2 0 0 0 0 0 0 0 0
 0 0 Ba1 0 0 De3 0 0 0 0
 0 0 Fr3 0 0 0 0 0 0 0
 0 Fr2 Cr2 0 0 0 0 0 0 Ba3
 0 0 0 0 0 Cr3 0 0 Ba3 0
 Ba1 Ba2 0 0 0 0 0 Cr1 0 0
 0 De1 0 0 0 0 0 0 0 0
 0 Cr2 0 0 0 0 0 0 Cr1 0
 0 0 0 0 0 0 0 0 0 0
```



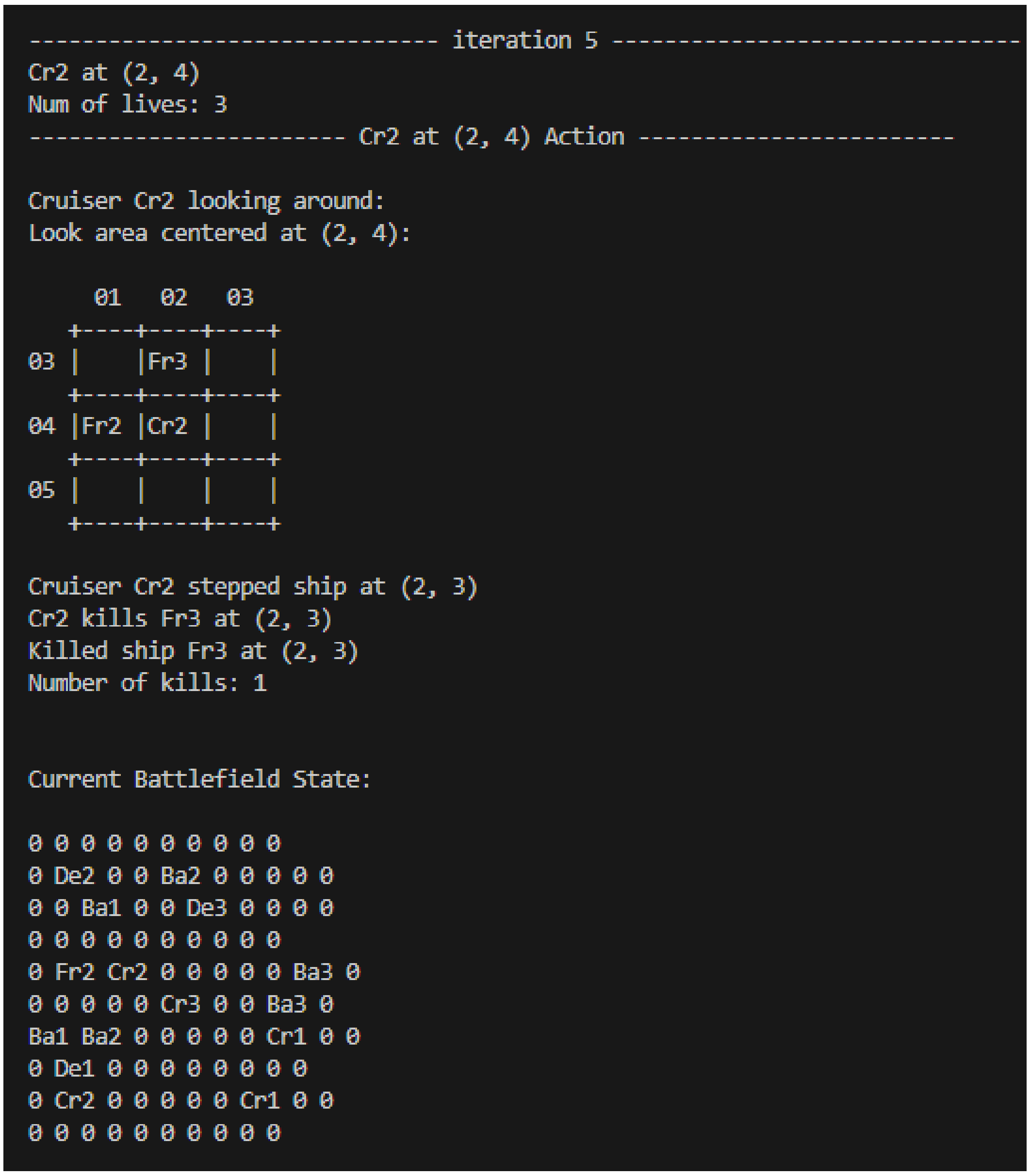
```
------------------------------- iteration 2 -----------------------------
Ba2 at (4, 0)
Num of lives: 3
------------------------ Ba2 at (4, 0) Action ------------------------

Battleship Ba2 looking around:
Look area centered at (4, 0):

     03   04   05
   +----+----+----+
   |####|####|####|
   +----+----+----+
00 |    |Ba2 |    |
   +----+----+----+
01 |    |    |    |
   +----+----+----+

Battleship Ba2 moved to (4, 1)
Ba2 fires at (8, 0)
Number of kills: 0
Ba2 fires at (2, 7)
Number of kills: 0
Ba2 fires at (8, 4)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 0 Ba3
0 0 0 0 0 Cr3 0 0 Ba3 0
Ba1 Ba2 0 0 0 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 0 Cr1 0
0 0 0 0 0 0 0 0 0 0
```



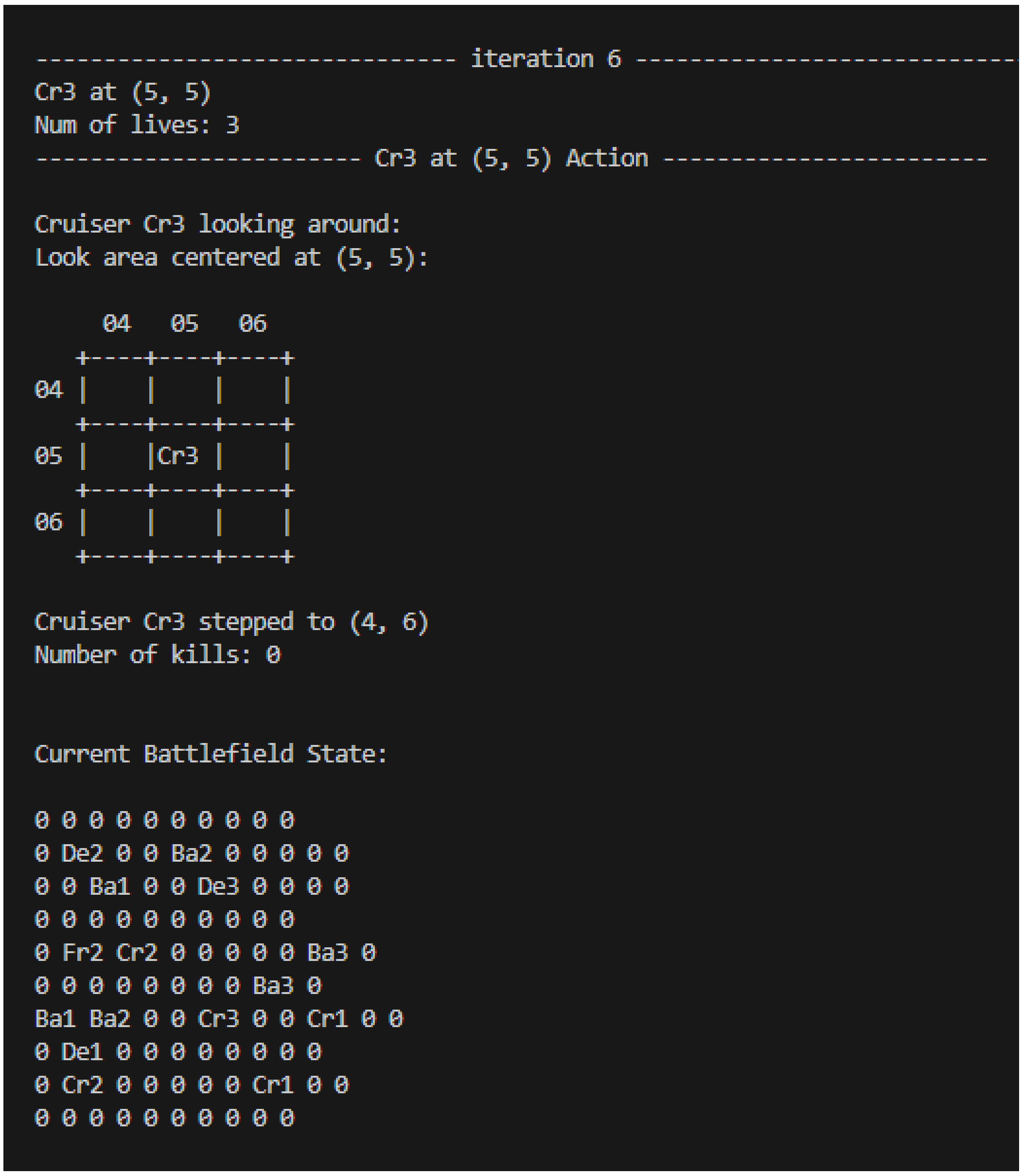
```
------------------------------- iteration 3 ------------------------------
Ba3 at (9, 4)
Num of lives: 3
------------------------ Ba3 at (9, 4) Action ------------------------

Battleship Ba3 looking around:
Look area centered at (9, 4):

     08   09
   +----+----+----+
03 |    |    |####|
   +----+----+----+
04 |    |Ba3 |####|
   +----+----+----+
05 |Ba3 |    |####|
   +----+----+----+

Battleship Ba3 moved to (8, 4)
Ba3 fires at (2, 6)
Number of kills: 0
Ba3 fires at (7, 5)
Number of kills: 0
Ba3 fires at (6, 3)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 Ba3 0
0 0 0 0 0 Cr3 0 0 Ba3 0
Ba1 Ba2 0 0 0 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 0 Cr1 0
0 0 0 0 0 0 0 0 0 0
```



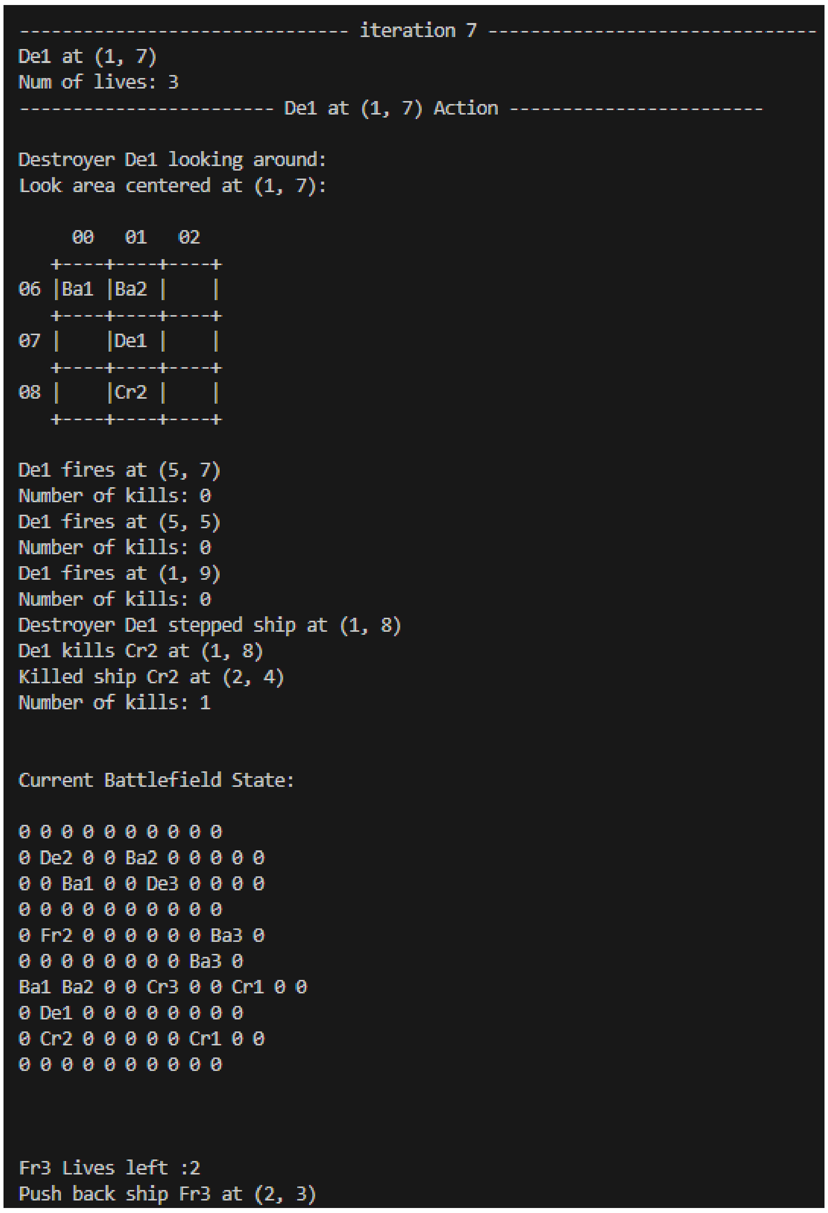
```
------------------------------- iteration 4 ------------------------------
Cr1 at (8, 8)
Num of lives: 3
------------------------ Cr1 at (8, 8) Action ------------------------

Cruiser Cr1 looking around:
Look area centered at (8, 8):

     07   08   09
   +----+----+----+
07 |    |    |    |
   +----+----+----+
08 |    |Cr1 |    |
   +----+----+----+
09 |    |    |    |
   +----+----+----+

Cruiser Cr1 stepped to (7, 8)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 Ba3 0
0 0 0 0 0 Cr3 0 0 Ba3 0
Ba1 Ba2 0 0 0 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0
```



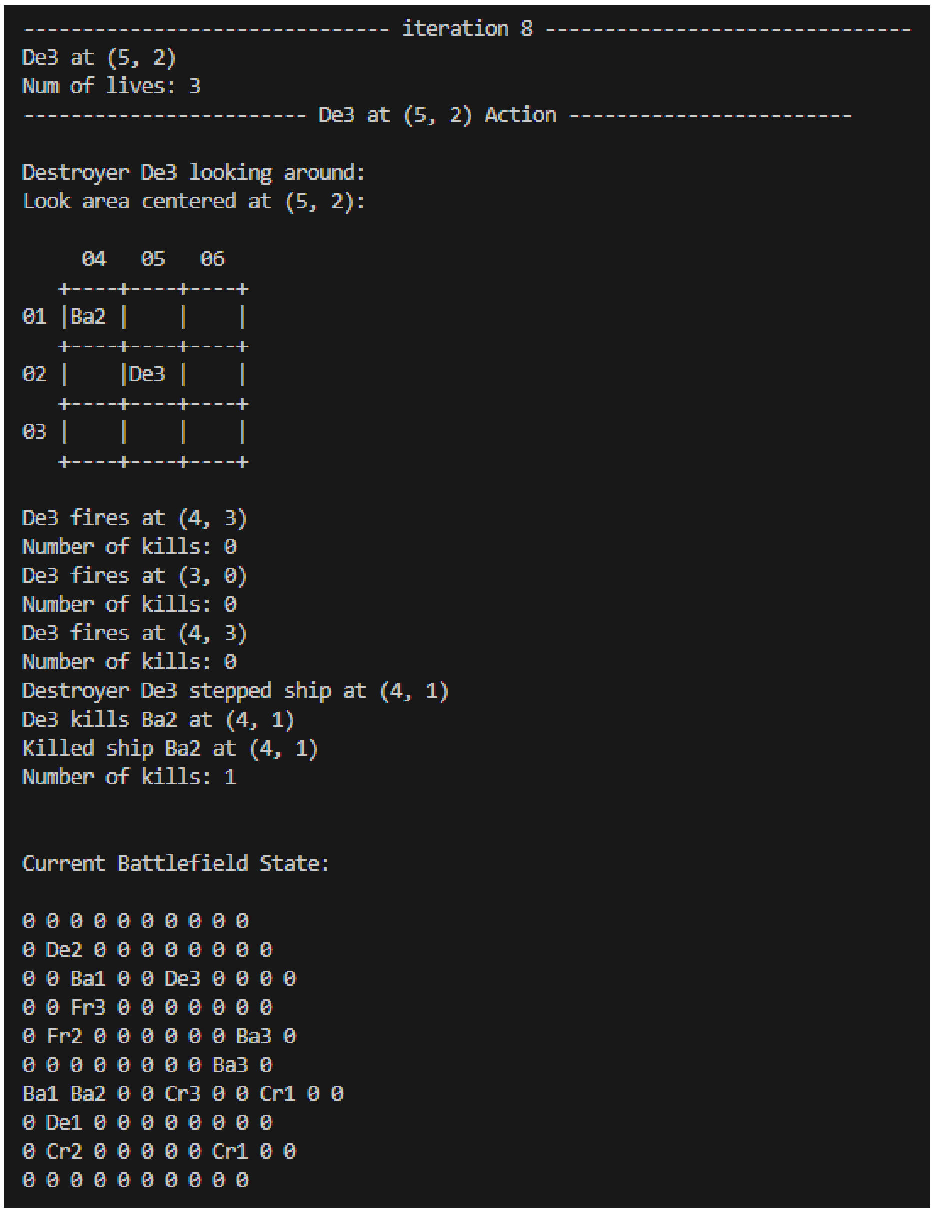
```
------------------------------- iteration 5 -------------------------------
Cr2 at (2, 4)
Num of lives: 3
------------------------ Cr2 at (2, 4) Action ------------------------

Cruiser Cr2 looking around:
Look area centered at (2, 4):

     01   02   03
   +----+----+----+
03 |    |Fr3 |    |
   +----+----+----+
04 |Fr2 |Cr2 |    |
   +----+----+----+
05 |    |    |    |
   +----+----+----+

Cruiser Cr2 stepped ship at (2, 3)
Cr2 kills Fr3 at (2, 3)
Killed ship Fr3 at (2, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 Ba3 0
0 0 0 0 0 Cr3 0 0 Ba3 0
Ba1 Ba2 0 0 0 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0
```



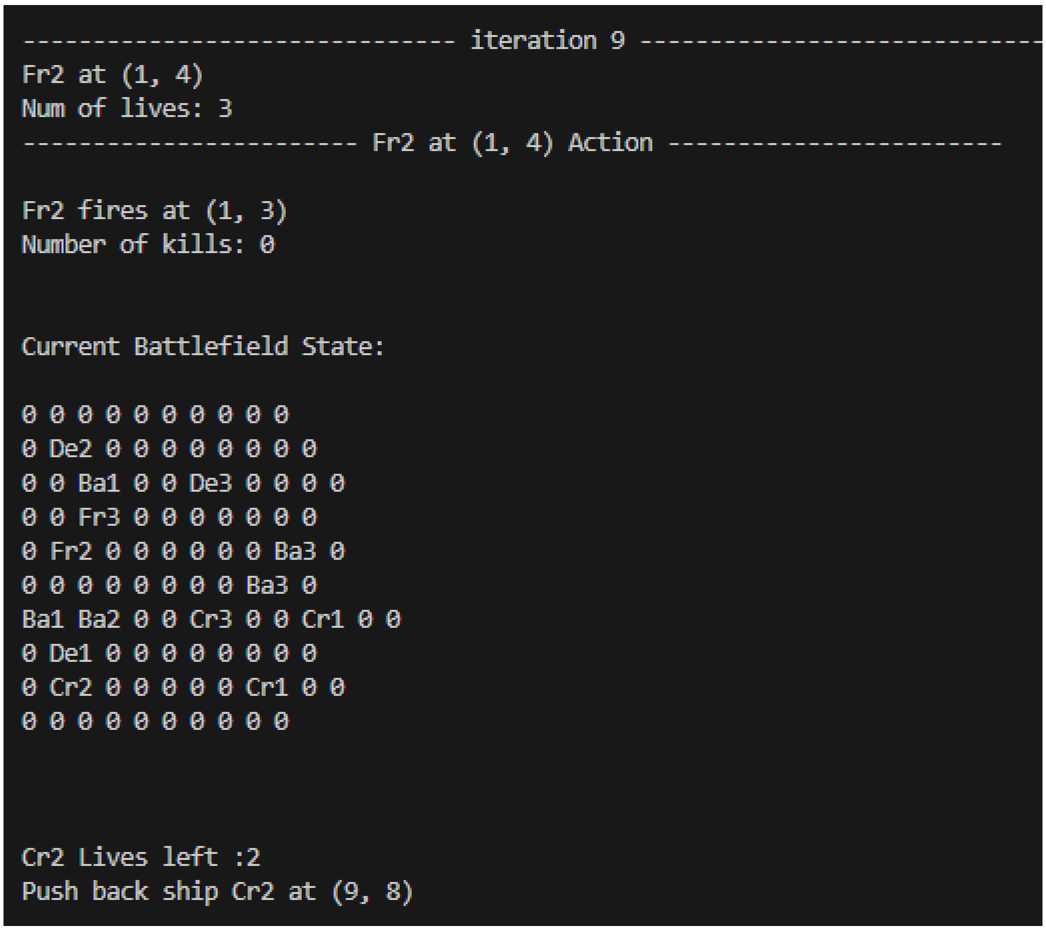
```
------------------------------- iteration 6 ----------------------------
Cr3 at (5, 5)
Num of lives: 3
------------------------ Cr3 at (5, 5) Action ------------------------

Cruiser Cr3 looking around:
Look area centered at (5, 5):

     04   05   06
   +----+----+----+
04 |    |    |    |
   +----+----+----+
05 |    |Cr3 |    |
   +----+----+----+
06 |    |    |    |
   +----+----+----+

Cruiser Cr3 stepped to (4, 6)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 Fr2 Cr2 0 0 0 0 0 Ba3 0
0 0 0 0 0 0 0 0 Ba3 0
Ba1 Ba2 0 0 Cr3 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0
```



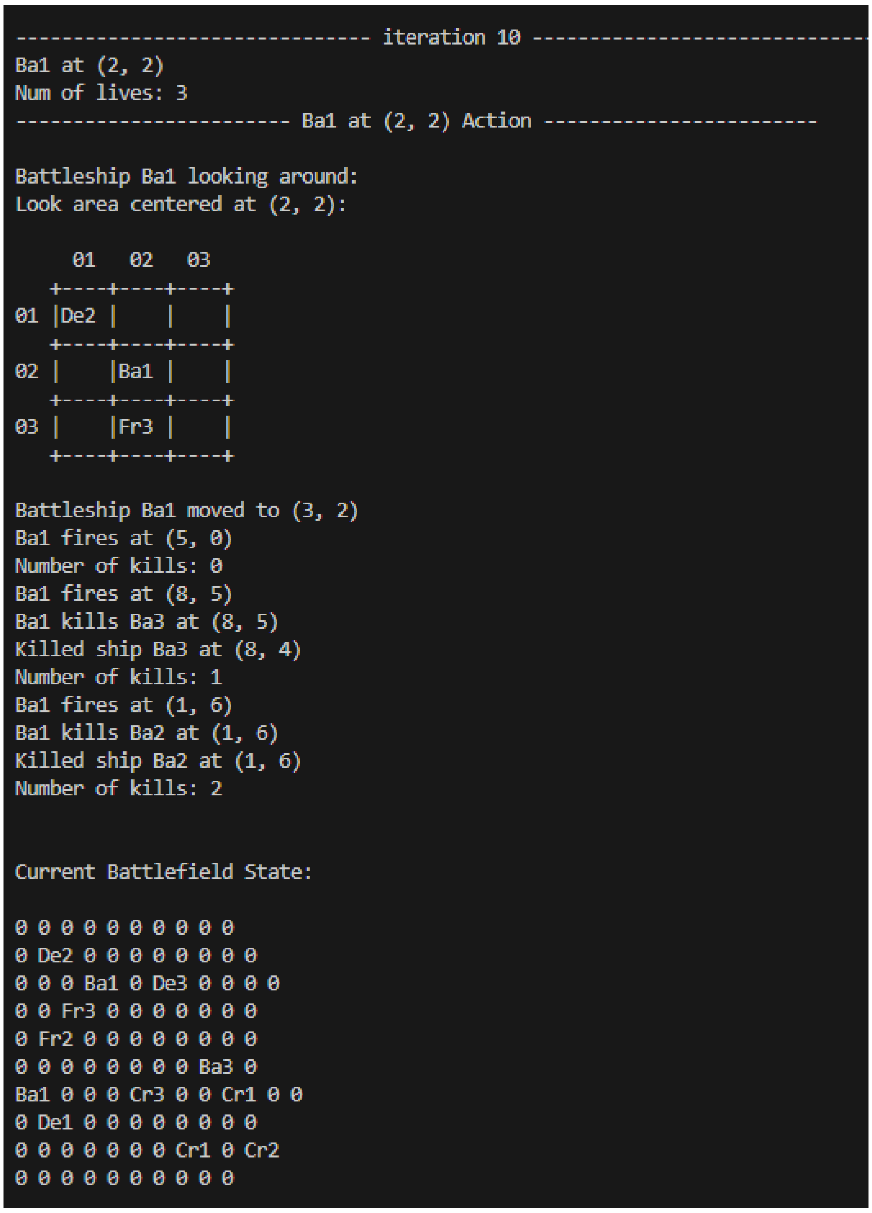
```
------------------------------- iteration 7 ------------------------------
De1 at (1, 7)
Num of lives: 3
------------------------ De1 at (1, 7) Action ------------------------

Destroyer De1 looking around:
Look area centered at (1, 7):

     00   01   02
   +----+----+----+
06 |Ba1 |Ba2 |    |
   +----+----+----+
07 |    |De1 |    |
   +----+----+----+
08 |    |Cr2 |    |
   +----+----+----+

De1 fires at (5, 7)
Number of kills: 0
De1 fires at (5, 5)
Number of kills: 0
De1 fires at (1, 9)
Number of kills: 0
Destroyer De1 stepped ship at (1, 8)
De1 kills Cr2 at (1, 8)
Killed ship Cr2 at (2, 4)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 Ba2 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 Ba3 0
0 0 0 0 0 0 0 0 Ba3 0
Ba1 Ba2 0 0 Cr3 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0



Fr3 Lives left :2
Push back ship Fr3 at (2, 3)
```



```
------------------------------- iteration 8 -------------------------------
De3 at (5, 2)
Num of lives: 3
------------------------ De3 at (5, 2) Action ------------------------

Destroyer De3 looking around:
Look area centered at (5, 2):

     04   05   06
   +----+----+----+
01 |Ba2 |    |    |
   +----+----+----+
02 |    |De3 |    |
   +----+----+----+
03 |    |    |    |
   +----+----+----+

De3 fires at (4, 3)
Number of kills: 0
De3 fires at (3, 0)
Number of kills: 0
De3 fires at (4, 3)
Number of kills: 0
Destroyer De3 stepped ship at (4, 1)
De3 kills Ba2 at (4, 1)
Killed ship Ba2 at (4, 1)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 Ba3 0
0 0 0 0 0 0 0 0 Ba3 0
Ba1 Ba2 0 0 Cr3 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0
```

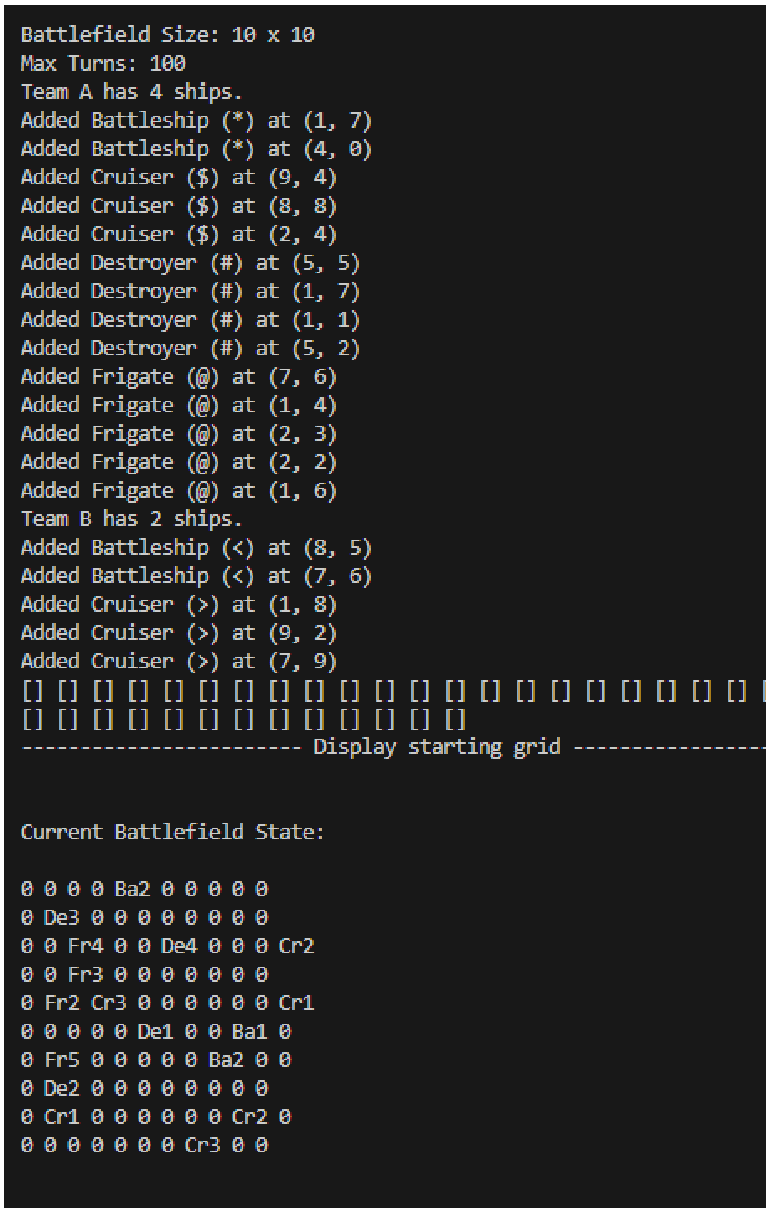


```
------------------------------- iteration 9 -----------------------------
Fr2 at (1, 4)
Num of lives: 3
------------------------ Fr2 at (1, 4) Action ------------------------

Fr2 fires at (1, 3)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 Ba1 0 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 Ba3 0
0 0 0 0 0 0 0 0 Ba3 0
Ba1 Ba2 0 0 Cr3 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 Cr2 0 0 0 0 0 Cr1 0 0
0 0 0 0 0 0 0 0 0 0



Cr2 Lives left :2
Push back ship Cr2 at (9, 8)
```

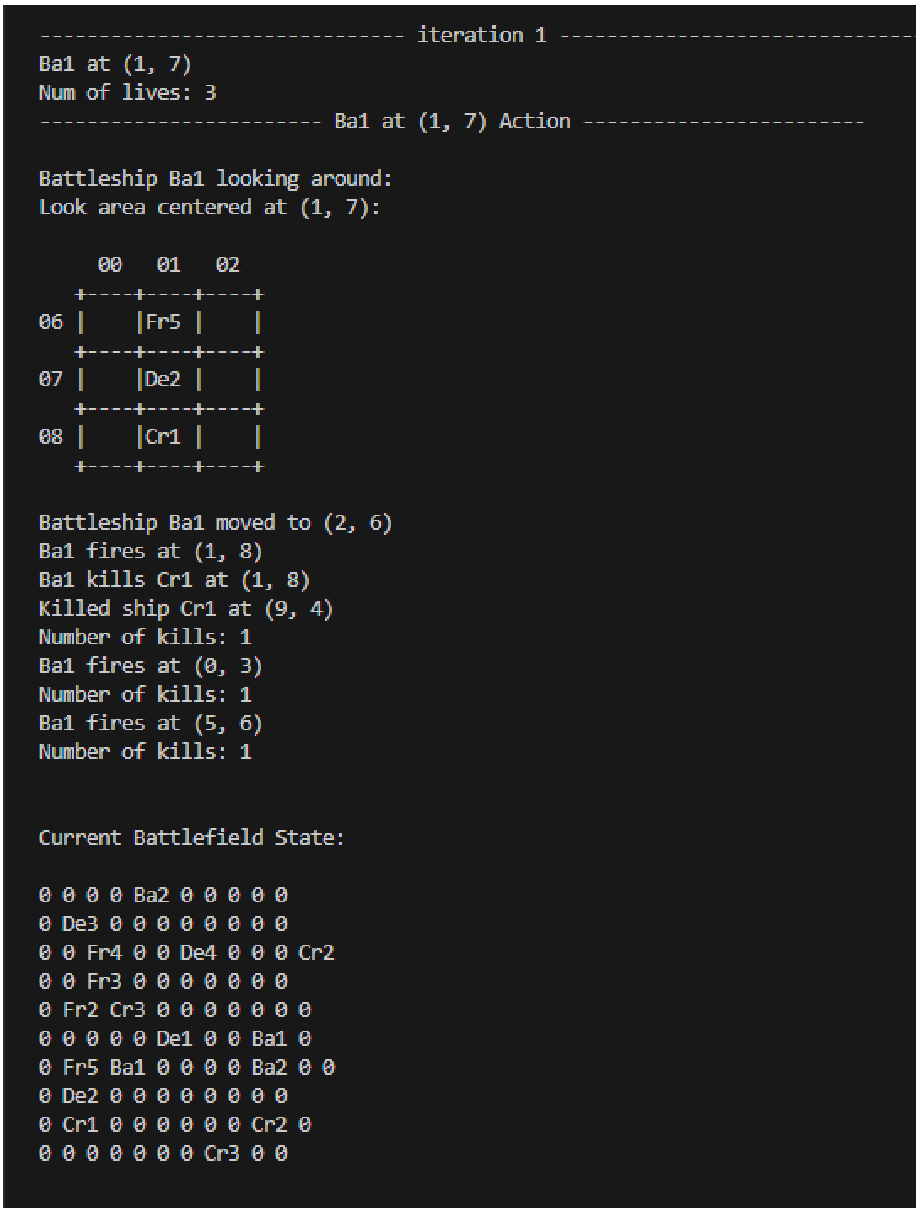


```
------------------------------- iteration 10 ------------------------------
Ba1 at (2, 2)
Num of lives: 3
------------------------ Ba1 at (2, 2) Action ------------------------

Battleship Ba1 looking around:
Look area centered at (2, 2):

     01   02   03
   +----+----+----+
01 |De2 |    |    |
   +----+----+----+
02 |    |Ba1 |    |
   +----+----+----+
03 |    |Fr3 |    |
   +----+----+----+

Battleship Ba1 moved to (3, 2)
Ba1 fires at (5, 0)
Number of kills: 0
Ba1 fires at (8, 5)
Ba1 kills Ba3 at (8, 5)
Killed ship Ba3 at (8, 4)
Number of kills: 1
Ba1 fires at (1, 6)
Ba1 kills Ba2 at (1, 6)
Killed ship Ba2 at (1, 6)
Number of kills: 2


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 0 Ba1 0 De3 0 0 0 0
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba3 0
Ba1 0 0 0 Cr3 0 0 Cr1 0 0
0 De1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 Cr1 0 Cr2
0 0 0 0 0 0 0 0 0 0
```

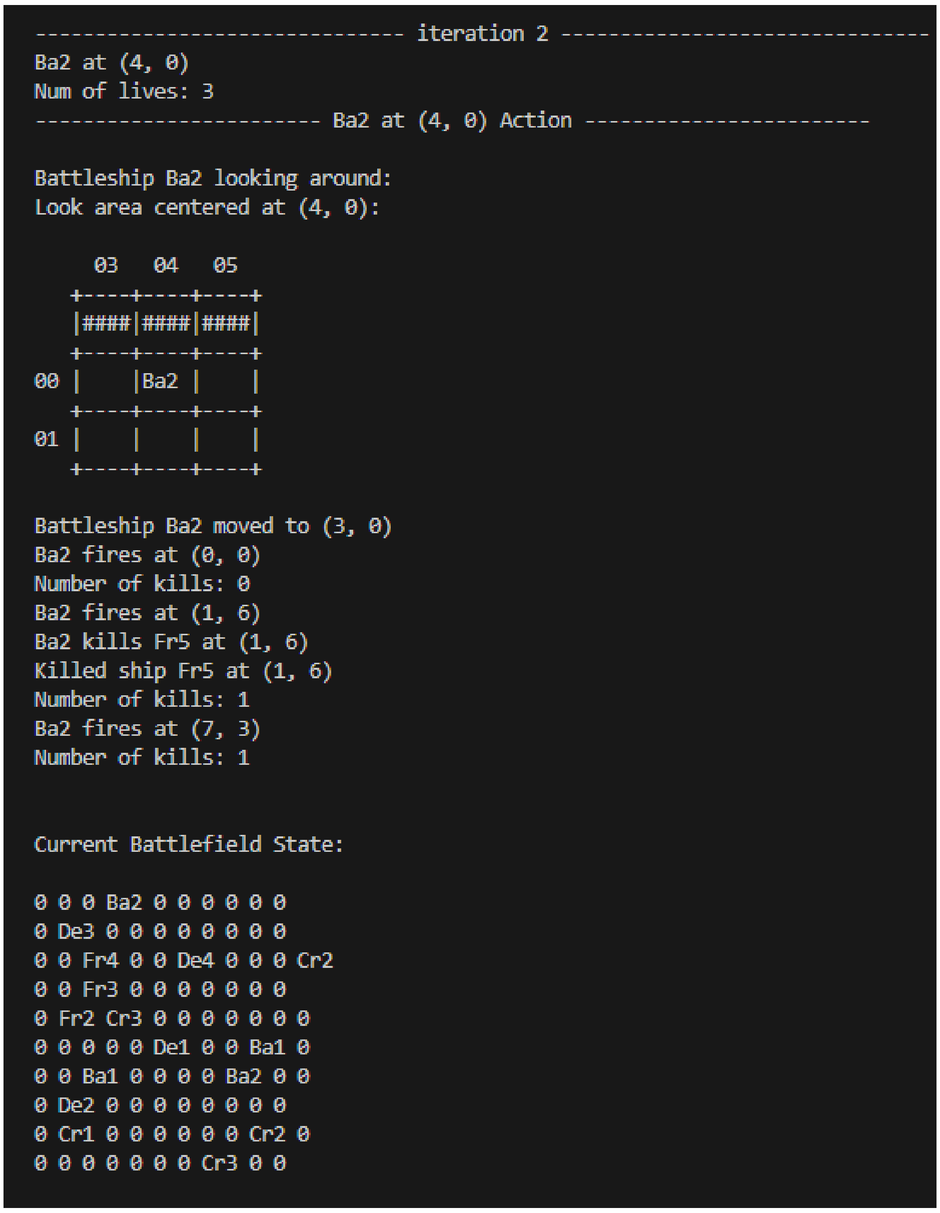
Second_Test:



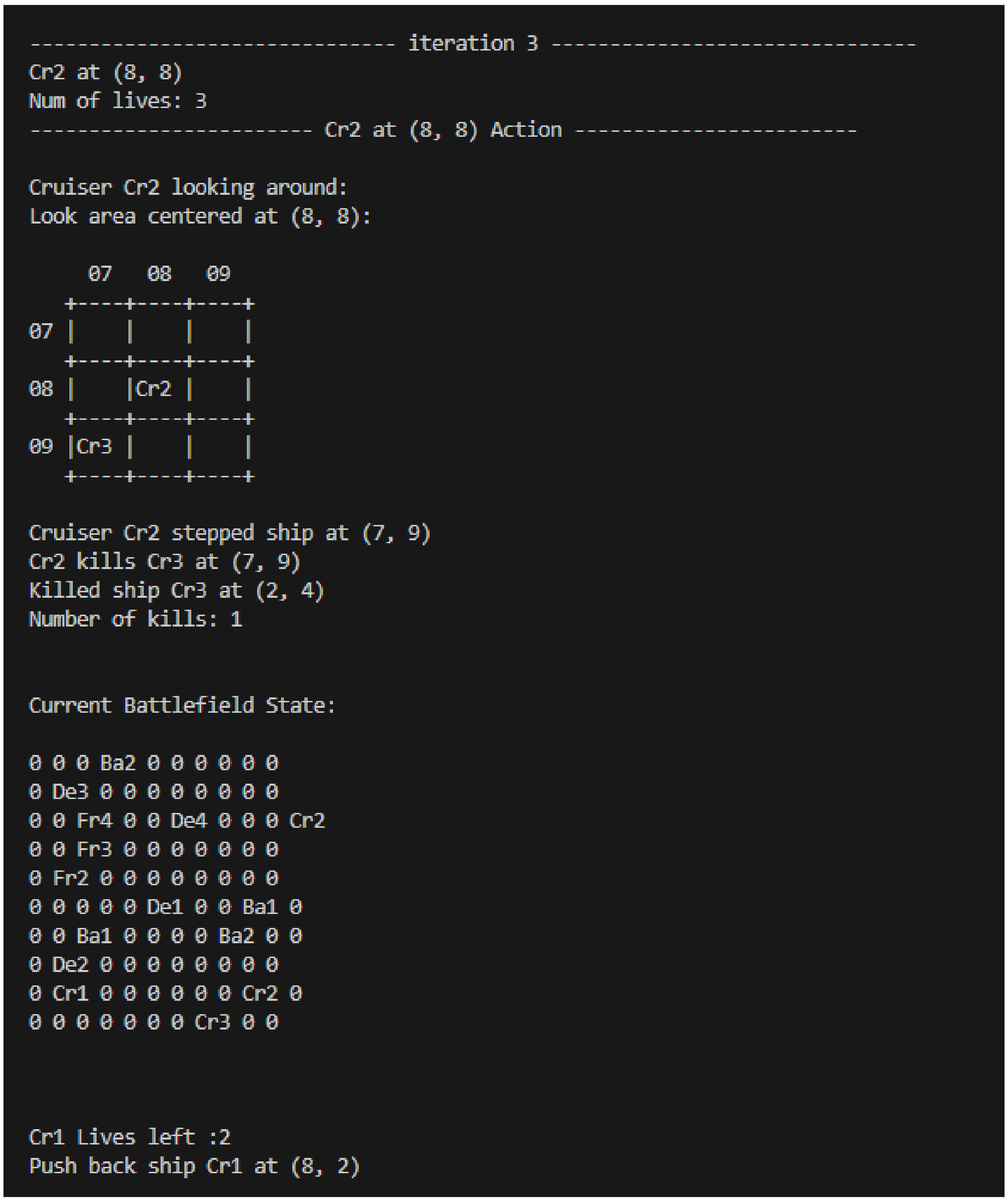
```
Battlefield Size: 10 x 10
Max Turns: 100
Team A has 4 ships.
Added Battleship (*) at (1, 7)
Added Battleship (*) at (4, 0)
Added Cruiser ($) at (9, 4)
Added Cruiser ($) at (8, 8)
Added Cruiser ($) at (2, 4)
Added Destroyer (#) at (5, 5)
Added Destroyer (#) at (1, 7)
Added Destroyer (#) at (1, 1)
Added Destroyer (#) at (5, 2)
Added Frigate (@) at (7, 6)
Added Frigate (@) at (1, 4)
Added Frigate (@) at (2, 3)
Added Frigate (@) at (2, 2)
Added Frigate (@) at (1, 6)
Team B has 2 ships.
Added Battleship (<) at (8, 5)
Added Battleship (<) at (7, 6)
Added Cruiser (>) at (1, 8)
Added Cruiser (>) at (9, 2)
Added Cruiser (>) at (7, 9)
[] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [
[] [] [] [] [] [] [] [] [] [] [] [] []
------------------------ Display starting grid ----------------


Current Battlefield State:

0 0 0 0 Ba2 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 0 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr3 0 0 0 0 0 0 Cr1
0 0 0 0 0 De1 0 0 Ba1 0
0 Fr5 0 0 0 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 Cr1 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0
```



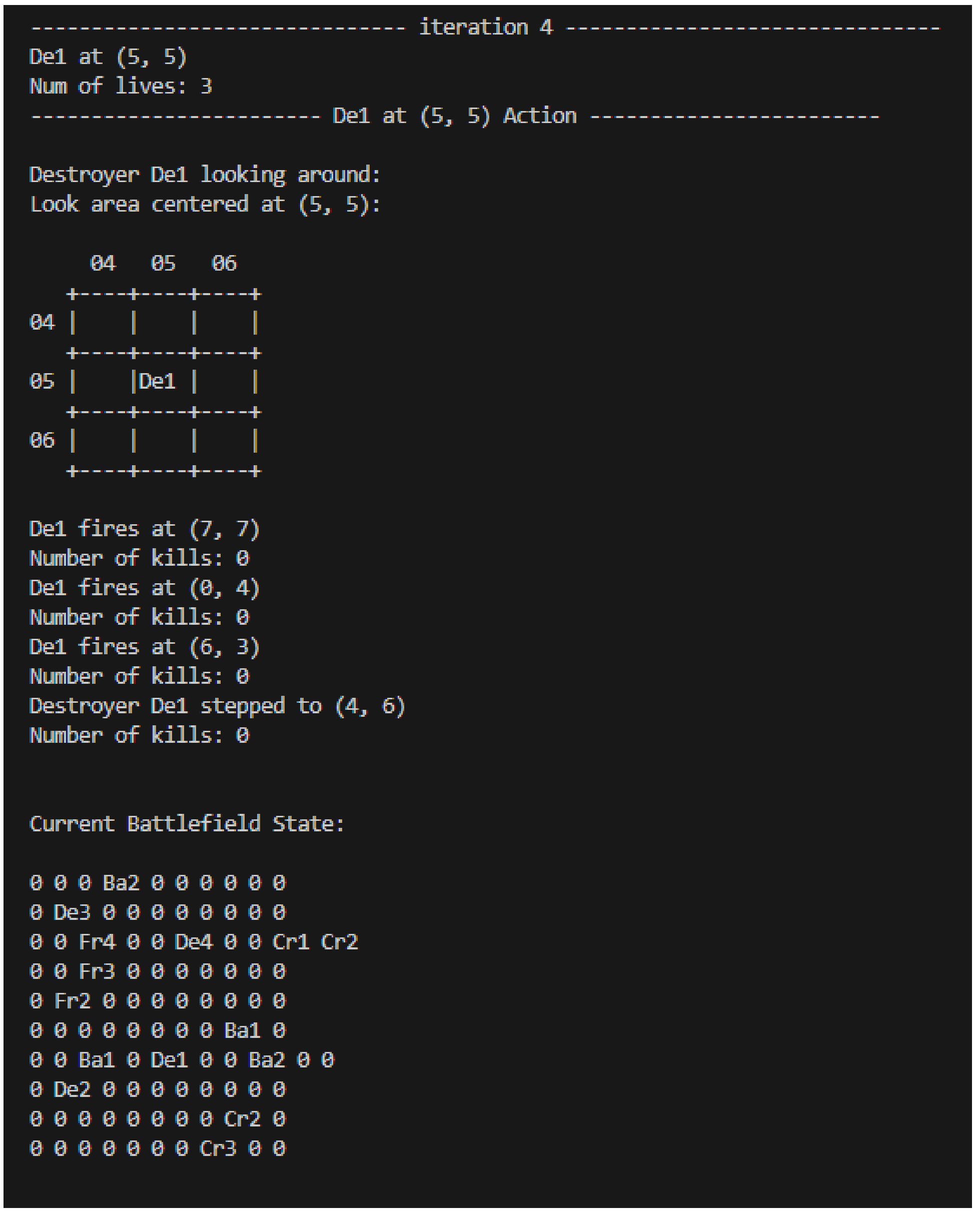
```
------------------------------- iteration 1 ------------------------------
Ba1 at (1, 7)
Num of lives: 3
------------------------ Ba1 at (1, 7) Action ------------------------

Battleship Ba1 looking around:
Look area centered at (1, 7):

     00   01   02
   +----+----+----+
06 |    |Fr5 |    |
   +----+----+----+
07 |    |De2 |    |
   +----+----+----+
08 |    |Cr1 |    |
   +----+----+----+

Battleship Ba1 moved to (2, 6)
Ba1 fires at (1, 8)
Ba1 kills Cr1 at (1, 8)
Killed ship Cr1 at (9, 4)
Number of kills: 1
Ba1 fires at (0, 3)
Number of kills: 1
Ba1 fires at (5, 6)
Number of kills: 1


Current Battlefield State:

0 0 0 0 Ba2 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 0 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr3 0 0 0 0 0 0 0
0 0 0 0 0 De1 0 0 Ba1 0
0 Fr5 Ba1 0 0 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 Cr1 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0
```



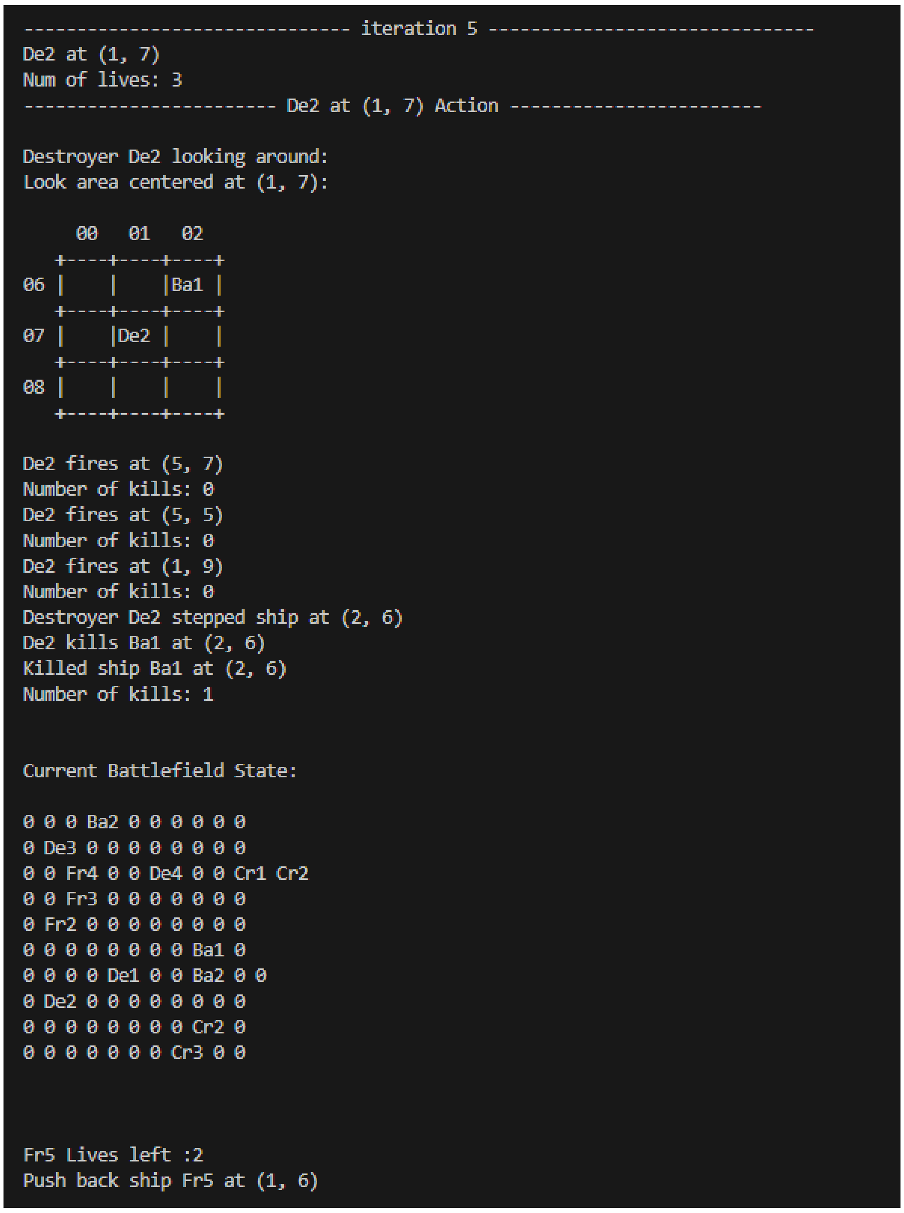
```
------------------------------- iteration 2 -------------------------------
Ba2 at (4, 0)
Num of lives: 3
------------------------ Ba2 at (4, 0) Action ------------------------

Battleship Ba2 looking around:
Look area centered at (4, 0):

     03   04   05
   +----+----+----+
   |####|####|####|
   +----+----+----+
00 |    |Ba2 |    |
   +----+----+----+
01 |    |    |    |
   +----+----+----+

Battleship Ba2 moved to (3, 0)
Ba2 fires at (0, 0)
Number of kills: 0
Ba2 fires at (1, 6)
Ba2 kills Fr5 at (1, 6)
Killed ship Fr5 at (1, 6)
Number of kills: 1
Ba2 fires at (7, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 0 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 Cr3 0 0 0 0 0 0 0
0 0 0 0 0 De1 0 0 Ba1 0
0 0 Ba1 0 0 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 Cr1 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0
```



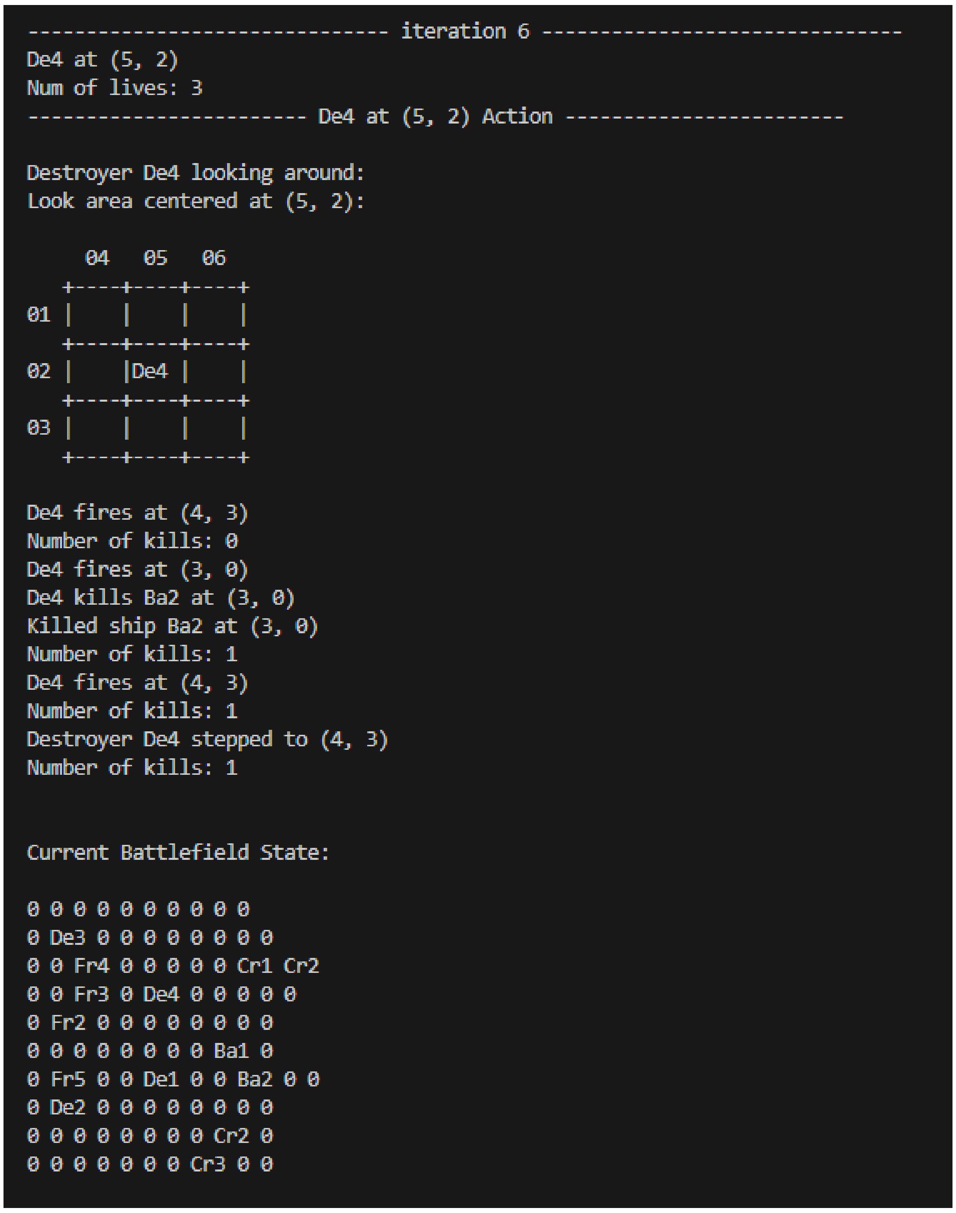
```
------------------------------- iteration 3 -------------------------------
Cr2 at (8, 8)
Num of lives: 3
------------------------ Cr2 at (8, 8) Action ------------------------

Cruiser Cr2 looking around:
Look area centered at (8, 8):

     07   08   09
   +----+----+----+
07 |    |    |    |
   +----+----+----+
08 |    |Cr2 |    |
   +----+----+----+
09 |Cr3 |    |    |
   +----+----+----+

Cruiser Cr2 stepped ship at (7, 9)
Cr2 kills Cr3 at (7, 9)
Killed ship Cr3 at (2, 4)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 0 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 De1 0 0 Ba1 0
0 0 Ba1 0 0 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 Cr1 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0



Cr1 Lives left :2
Push back ship Cr1 at (8, 2)
```



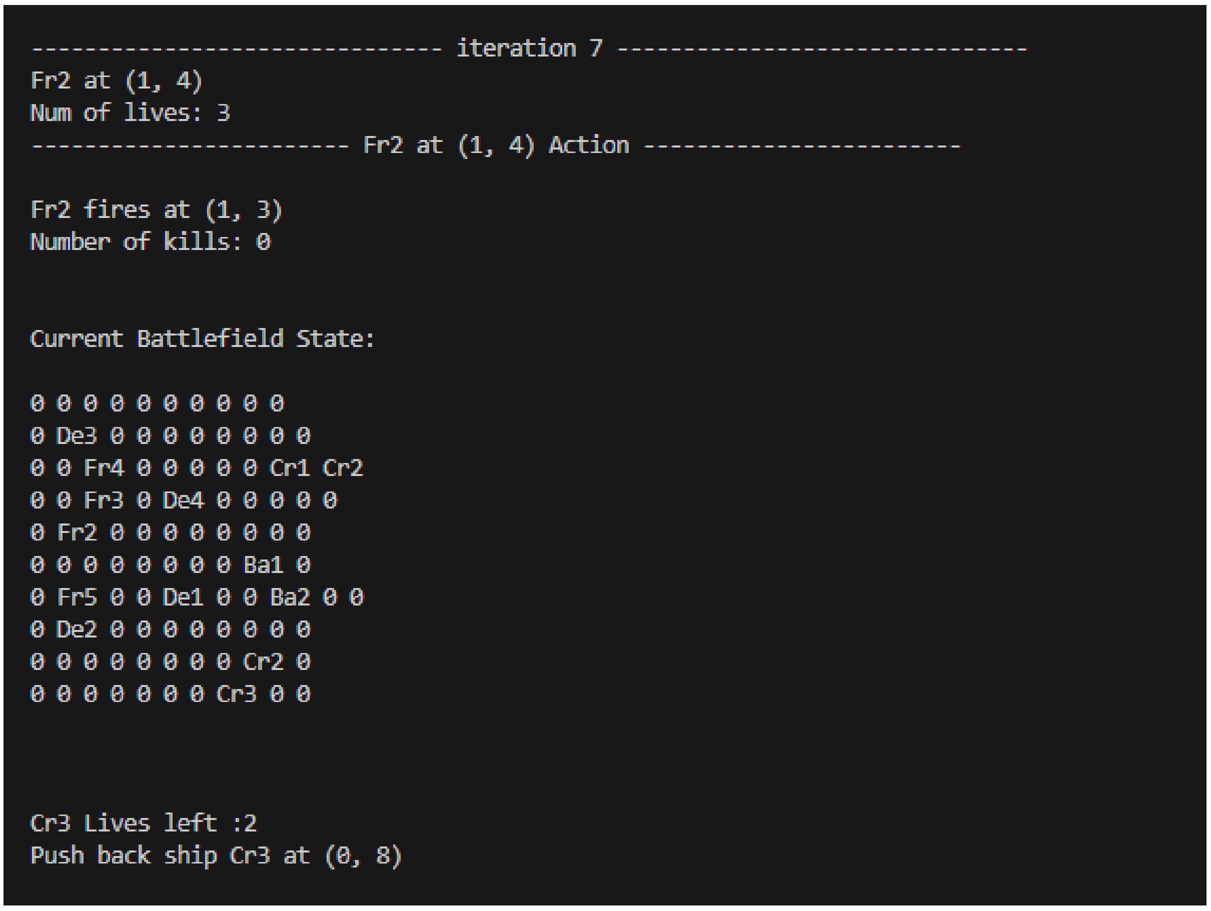
```
------------------------------- iteration 4 -------------------------------
De1 at (5, 5)
Num of lives: 3
------------------------ De1 at (5, 5) Action ------------------------

Destroyer De1 looking around:
Look area centered at (5, 5):

     04   05   06
   +----+----+----+
04 |    |    |    |
   +----+----+----+
05 |    |De1 |    |
   +----+----+----+
06 |    |    |    |
   +----+----+----+

De1 fires at (7, 7)
Number of kills: 0
De1 fires at (0, 4)
Number of kills: 0
De1 fires at (6, 3)
Number of kills: 0
Destroyer De1 stepped to (4, 6)
Number of kills: 0


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 Cr1 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba1 0
0 0 Ba1 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0
```

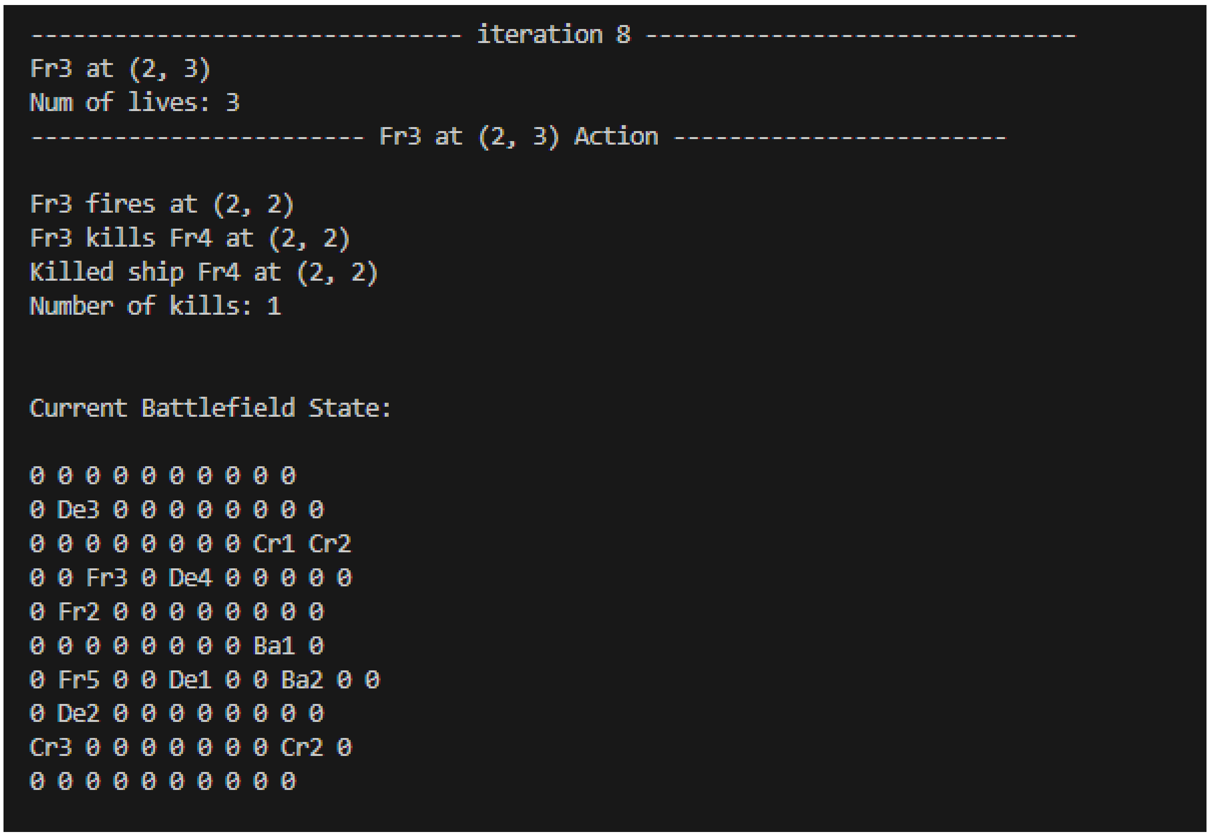


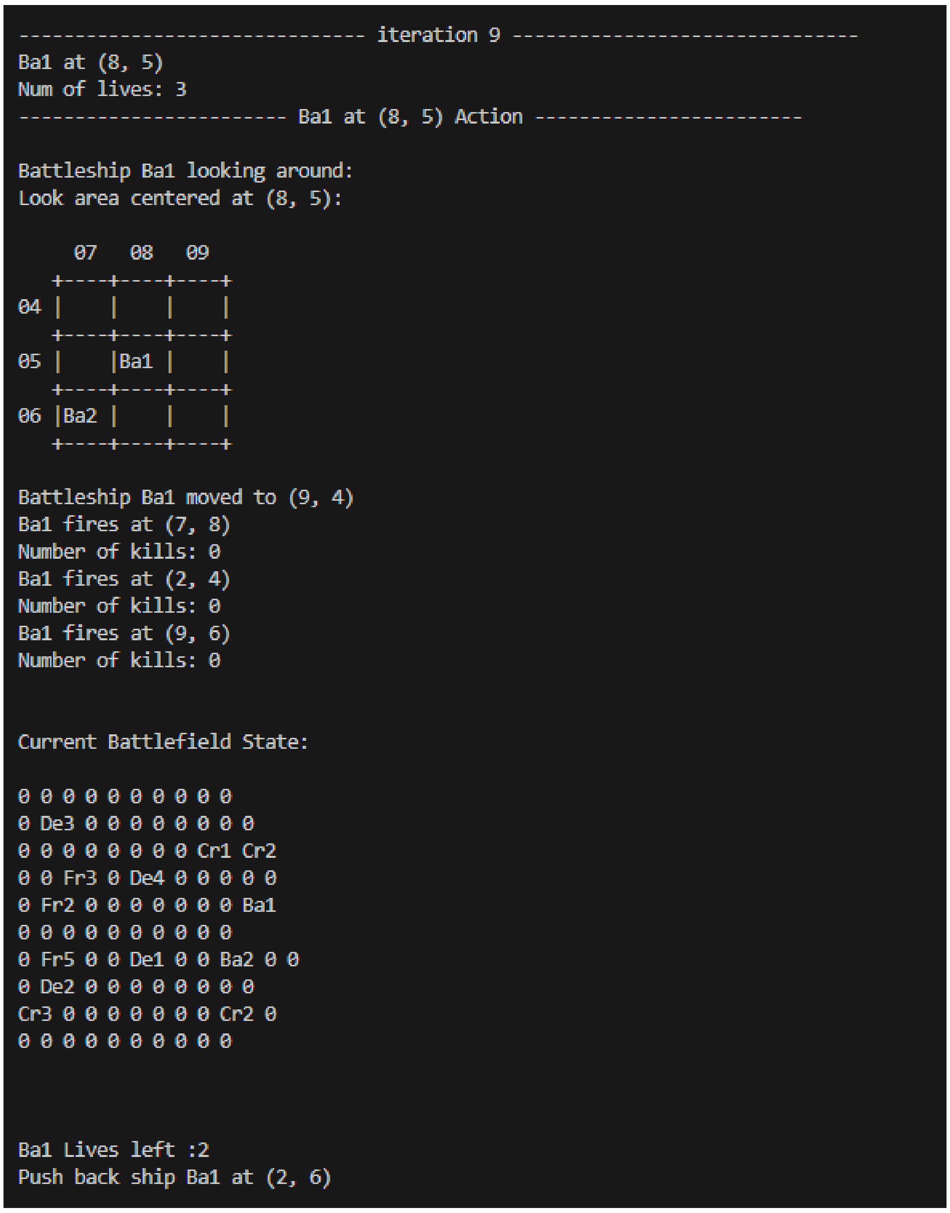
```
------------------------------ iteration 5 ------------------------------
De2 at (1, 7)
Num of lives: 3
----------------------- De2 at (1, 7) Action -----------------------

Destroyer De2 looking around:
Look area centered at (1, 7):

     00   01   02
   +----+----+----+
06 |    |    |Ba1 |
   +----+----+----+
07 |    |De2 |    |
   +----+----+----+
08 |    |    |    |
   +----+----+----+

De2 fires at (5, 7)
Number of kills: 0
De2 fires at (5, 5)
Number of kills: 0
De2 fires at (1, 9)
Number of kills: 0
Destroyer De2 stepped ship at (2, 6)
De2 kills Ba1 at (2, 6)
Killed ship Ba1 at (2, 6)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 De4 0 0 Cr1 Cr2
0 0 Fr3 0 0 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba1 0
0 0 0 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0



Fr5 Lives left :2
Push back ship Fr5 at (1, 6)
```



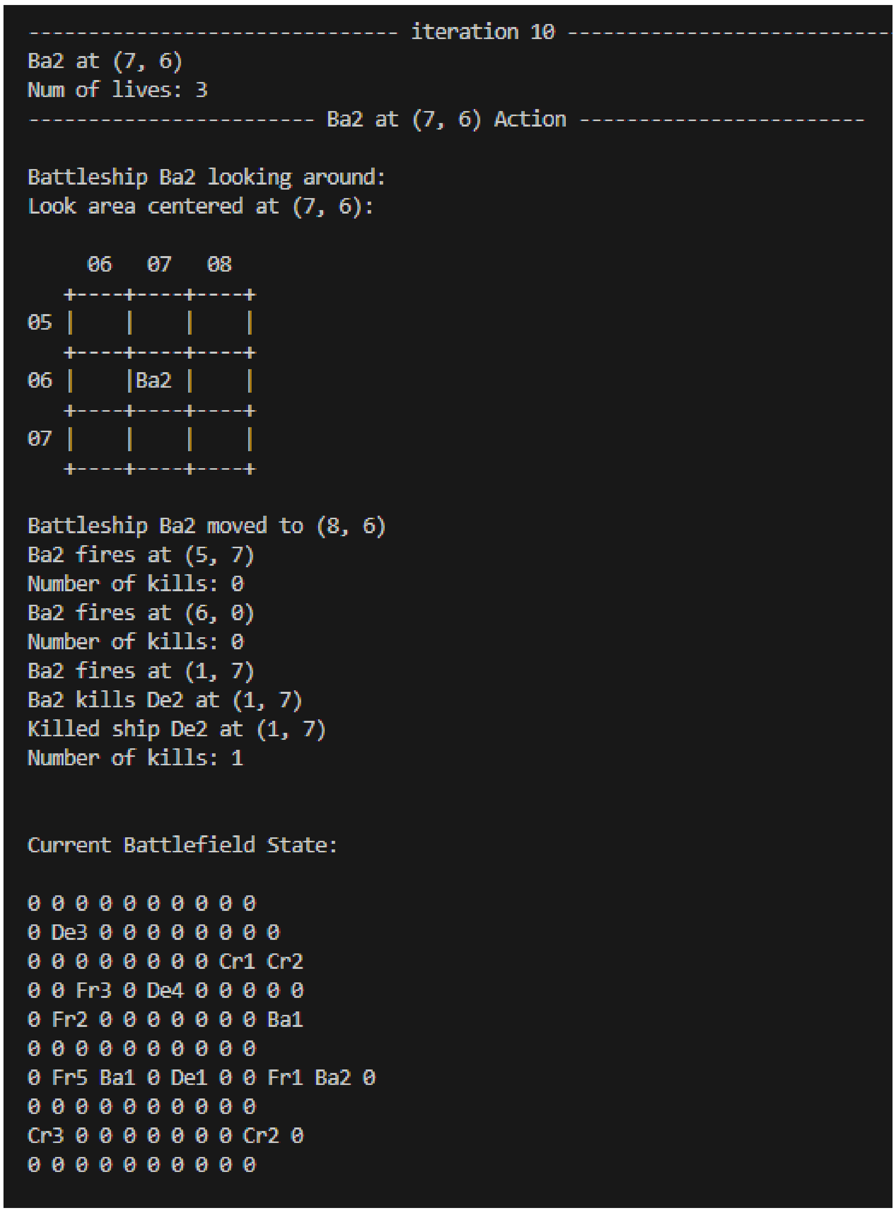
```
------------------------------- iteration 6 -------------------------------
De4 at (5, 2)
Num of lives: 3
------------------------ De4 at (5, 2) Action ------------------------

Destroyer De4 looking around:
Look area centered at (5, 2):

     04   05   06
   +----+----+----+
01 |    |    |    |
   +----+----+----+
02 |    |De4 |    |
   +----+----+----+
03 |    |    |    |
   +----+----+----+

De4 fires at (4, 3)
Number of kills: 0
De4 fires at (3, 0)
De4 kills Ba2 at (3, 0)
Killed ship Ba2 at (3, 0)
Number of kills: 1
De4 fires at (4, 3)
Number of kills: 1
Destroyer De4 stepped to (4, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 0 0 0 Cr1 Cr2
0 0 Fr3 0 De4 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba1 0
0 Fr5 0 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0
```


```
------------------------------ iteration 7 ------------------------------
Fr2 at (1, 4)
Num of lives: 3
------------------------ Fr2 at (1, 4) Action ------------------------

Fr2 fires at (1, 3)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 Fr4 0 0 0 0 0 Cr1 Cr2
0 0 Fr3 0 De4 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba1 0
0 Fr5 0 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 Cr3 0 0



Cr3 Lives left :2
Push back ship Cr3 at (0, 8)
```


```
------------------------------- iteration 8 -------------------------------
Fr3 at (2, 3)
Num of lives: 3
------------------------ Fr3 at (2, 3) Action ------------------------

Fr3 fires at (2, 2)
Fr3 kills Fr4 at (2, 2)
Killed ship Fr4 at (2, 2)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr1 Cr2
0 0 Fr3 0 De4 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Ba1 0
0 Fr5 0 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
Cr3 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 0 0 0
```

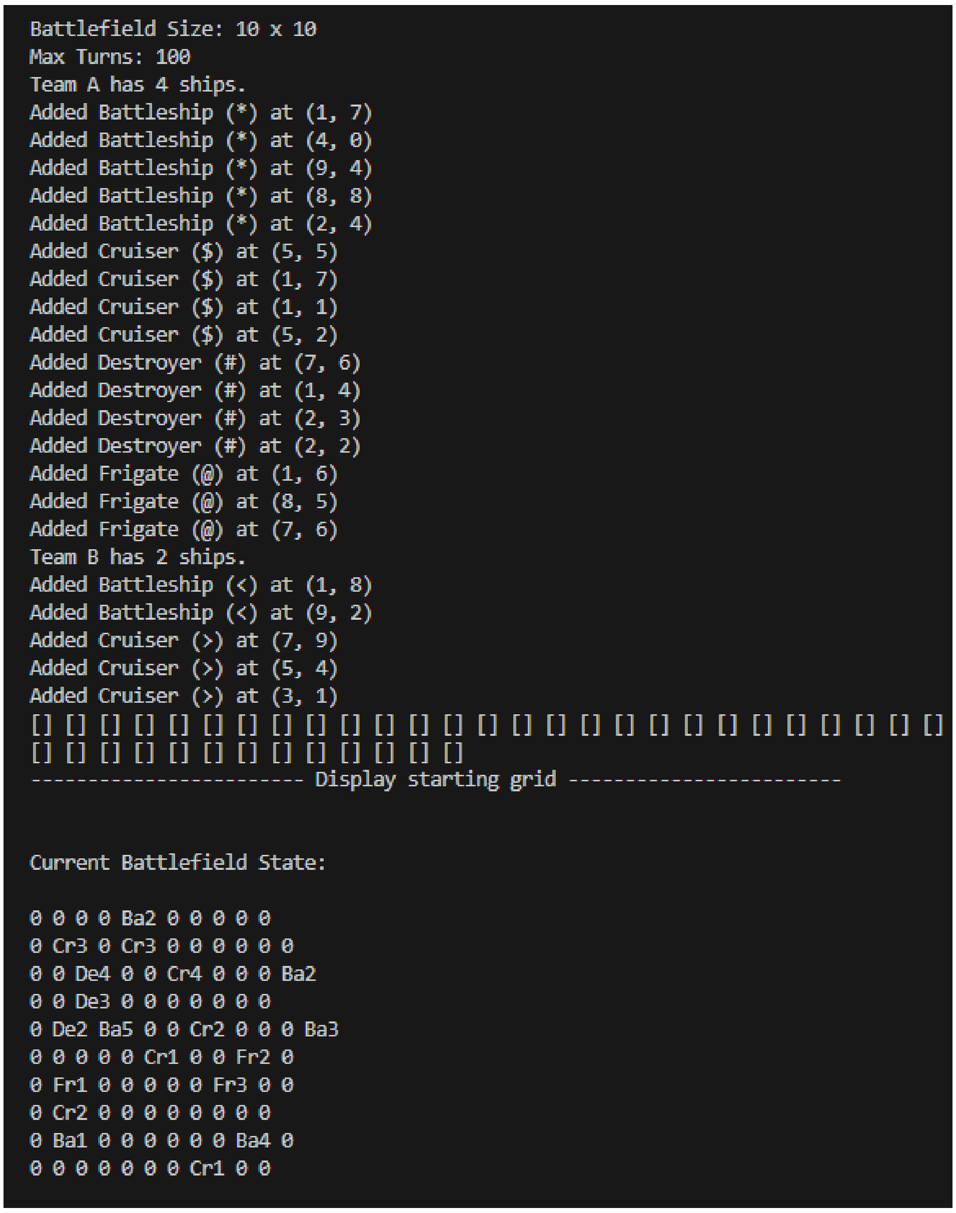


```
------------------------------ iteration 9 ------------------------------
Ba1 at (8, 5)
Num of lives: 3
----------------------- Ba1 at (8, 5) Action -----------------------

Battleship Ba1 looking around:
Look area centered at (8, 5):

     07   08   09
   +----+----+----+
04 |    |    |    |
   +----+----+----+
05 |    |Ba1 |    |
   +----+----+----+
06 |Ba2 |    |    |
   +----+----+----+

Battleship Ba1 moved to (9, 4)
Ba1 fires at (7, 8)
Number of kills: 0
Ba1 fires at (2, 4)
Number of kills: 0
Ba1 fires at (9, 6)
Number of kills: 0


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr1 Cr2
0 0 Fr3 0 De4 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 Ba1
0 0 0 0 0 0 0 0 0 0
0 Fr5 0 0 De1 0 0 Ba2 0 0
0 De2 0 0 0 0 0 0 0 0
Cr3 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 0 0 0



Ba1 Lives left :2
Push back ship Ba1 at (2, 6)
```

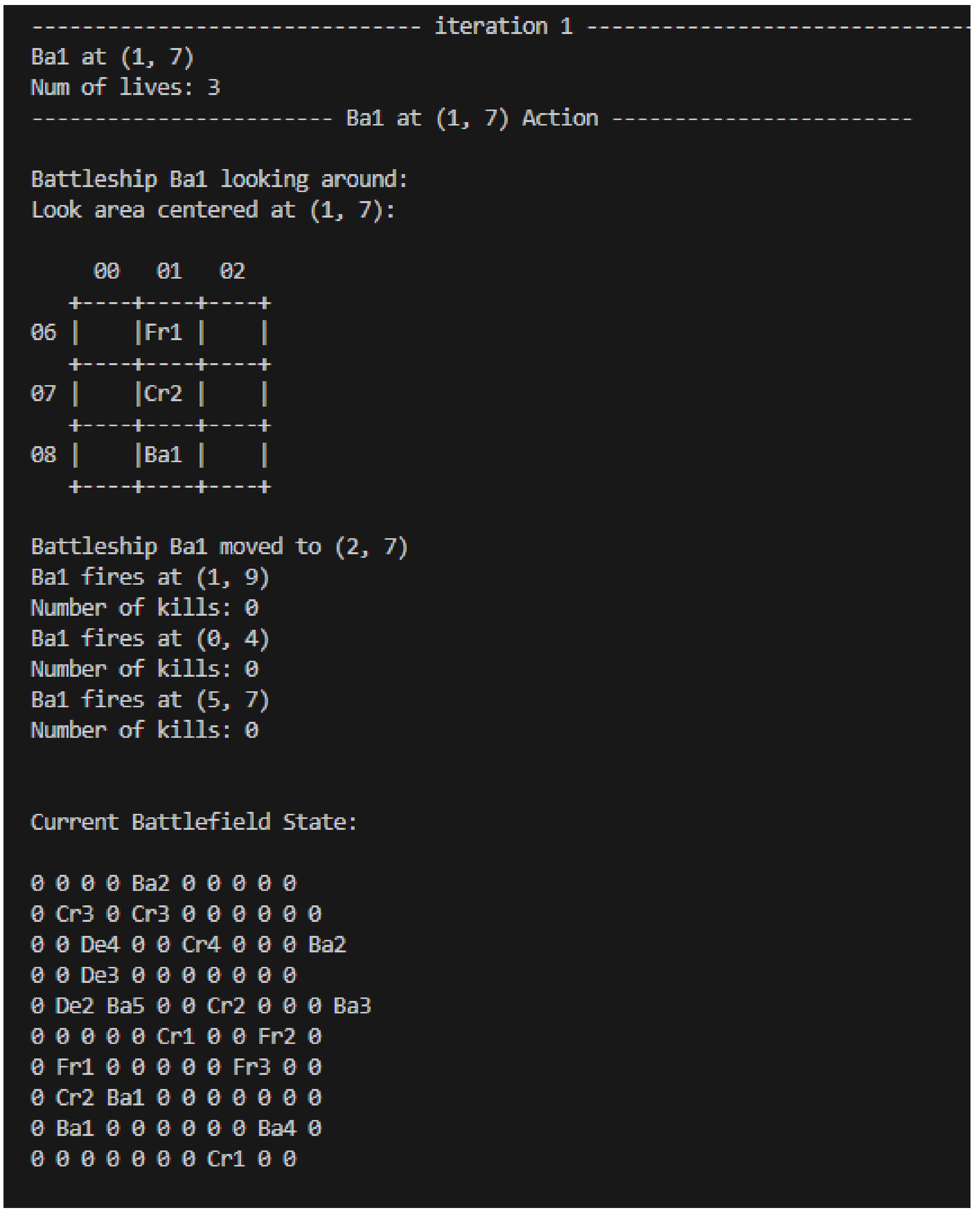


```
------------------------------- iteration 10 ----------------------------
Ba2 at (7, 6)
Num of lives: 3
------------------------ Ba2 at (7, 6) Action ------------------------

Battleship Ba2 looking around:
Look area centered at (7, 6):

     06   07   08
   +----+----+----+
05 |    |    |    |
   +----+----+----+
06 |    |Ba2 |    |
   +----+----+----+
07 |    |    |    |
   +----+----+----+

Battleship Ba2 moved to (8, 6)
Ba2 fires at (5, 7)
Number of kills: 0
Ba2 fires at (6, 0)
Number of kills: 0
Ba2 fires at (1, 7)
Ba2 kills De2 at (1, 7)
Killed ship De2 at (1, 7)
Number of kills: 1


Current Battlefield State:

0 0 0 0 0 0 0 0 0 0
0 De3 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 Cr1 Cr2
0 0 Fr3 0 De4 0 0 0 0 0
0 Fr2 0 0 0 0 0 0 0 Ba1
0 0 0 0 0 0 0 0 0 0
0 Fr5 Ba1 0 De1 0 0 Fr1 Ba2 0
0 0 0 0 0 0 0 0 0 0
Cr3 0 0 0 0 0 0 0 Cr2 0
0 0 0 0 0 0 0 0 0 0
```

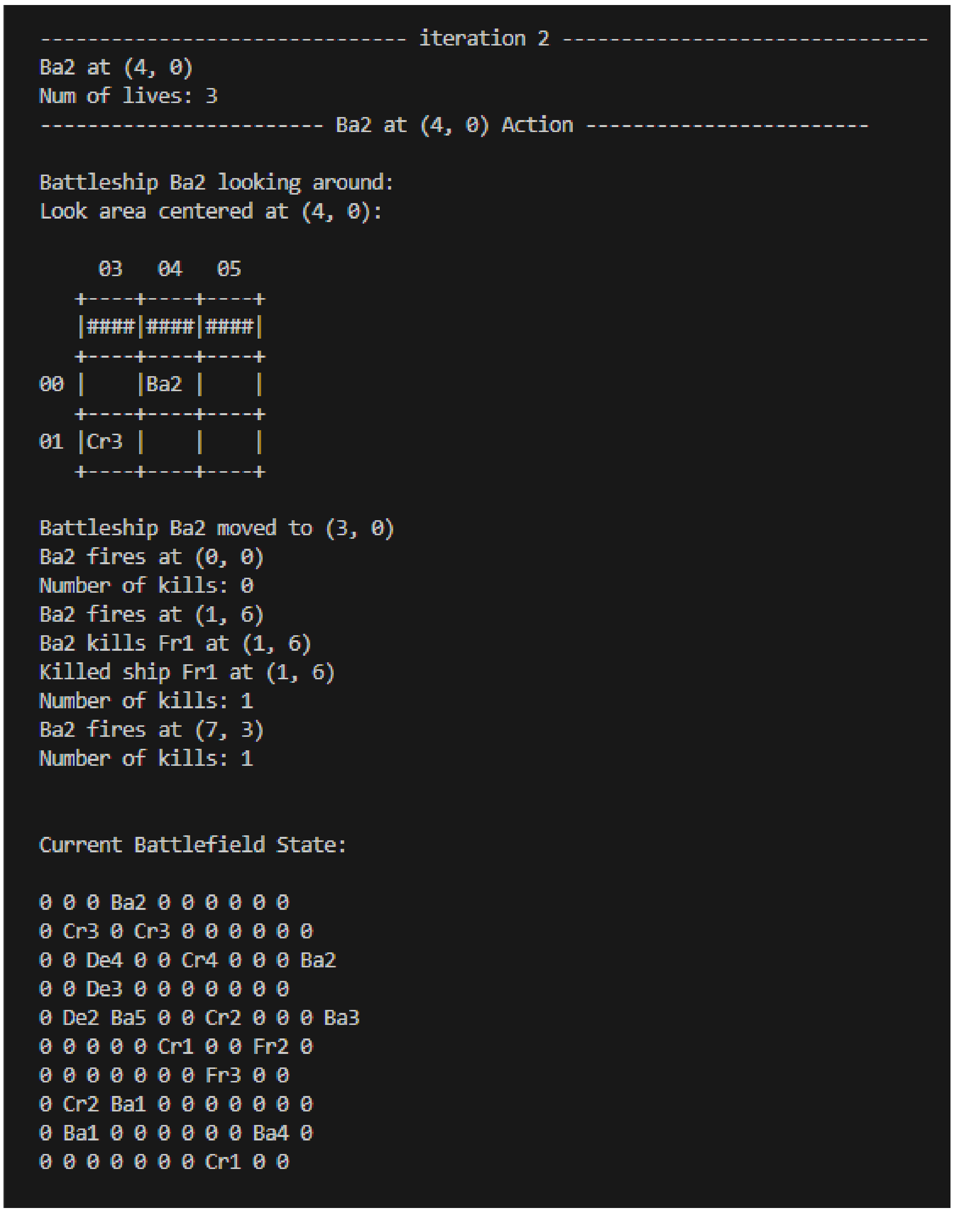
Third_Test:



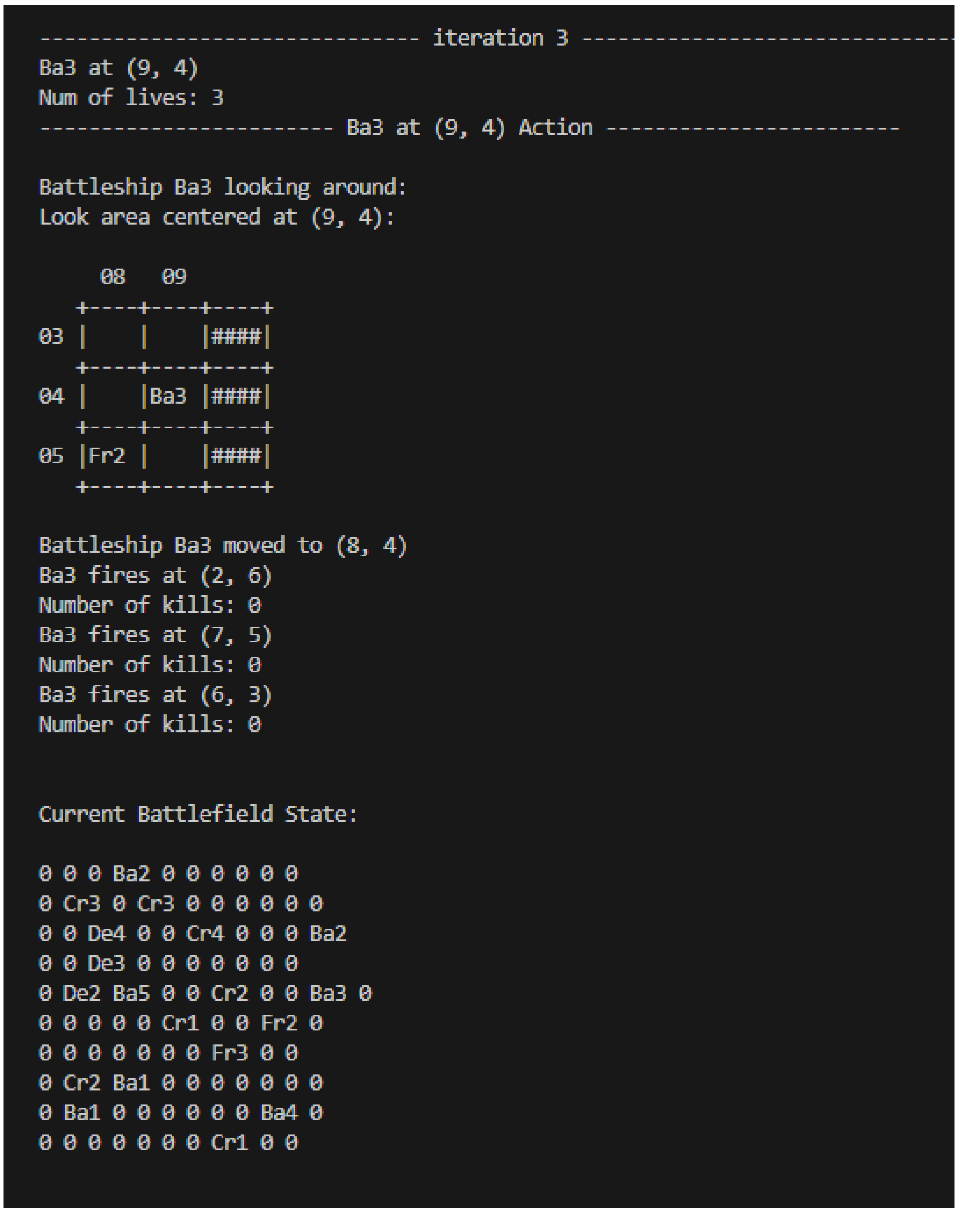
```
Battlefield Size: 10 x 10
Max Turns: 100
Team A has 4 ships.
Added Battleship (*) at (1, 7)
Added Battleship (*) at (4, 0)
Added Battleship (*) at (9, 4)
Added Battleship (*) at (8, 8)
Added Battleship (*) at (2, 4)
Added Cruiser ($) at (5, 5)
Added Cruiser ($) at (1, 7)
Added Cruiser ($) at (1, 1)
Added Cruiser ($) at (5, 2)
Added Destroyer (#) at (7, 6)
Added Destroyer (#) at (1, 4)
Added Destroyer (#) at (2, 3)
Added Destroyer (#) at (2, 2)
Added Frigate (@) at (1, 6)
Added Frigate (@) at (8, 5)
Added Frigate (@) at (7, 6)
Team B has 2 ships.
Added Battleship (<) at (1, 8)
Added Battleship (<) at (9, 2)
Added Cruiser (>) at (7, 9)
Added Cruiser (>) at (5, 4)
Added Cruiser (>) at (3, 1)
[] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
[] [] [] [] [] [] [] [] [] [] [] [] []
------------------------ Display starting grid ------------------------


Current Battlefield State:

0 0 0 0 Ba2 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 0 0 0 0 0 0 0
0 De2 Ba5 0 0 Cr2 0 0 0 Ba3
0 0 0 0 0 Cr1 0 0 Fr2 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 Cr2 0 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 0 Ba4 0
0 0 0 0 0 0 0 Cr1 0 0
```



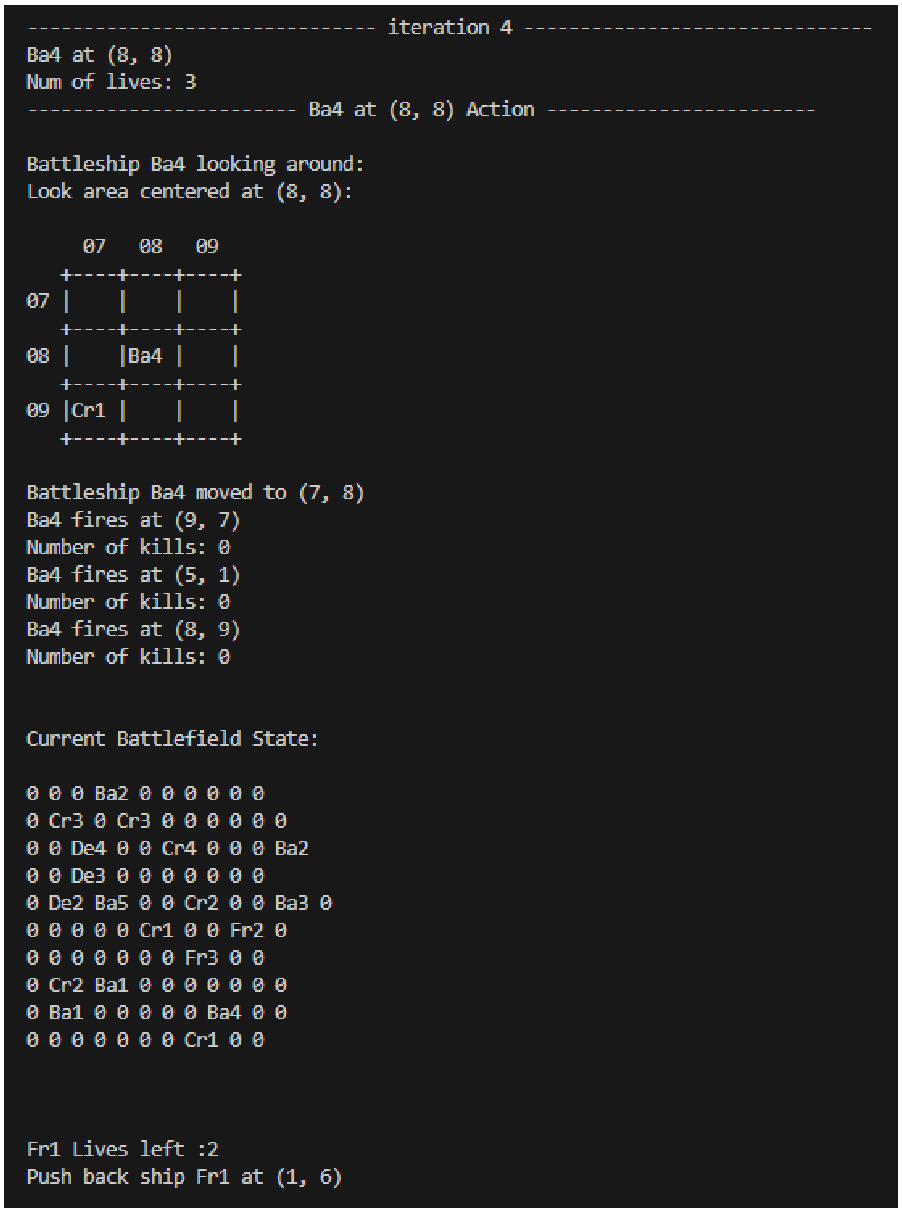
```
------------------------------- iteration 1 ------------------------------
Ba1 at (1, 7)
Num of lives: 3
----------------------- Ba1 at (1, 7) Action -----------------------

Battleship Ba1 looking around:
Look area centered at (1, 7):

     00   01   02
   +----+----+----+
06 |    |Fr1 |    |
   +----+----+----+
07 |    |Cr2 |    |
   +----+----+----+
08 |    |Ba1 |    |
   +----+----+----+

Battleship Ba1 moved to (2, 7)
Ba1 fires at (1, 9)
Number of kills: 0
Ba1 fires at (0, 4)
Number of kills: 0
Ba1 fires at (5, 7)
Number of kills: 0


Current Battlefield State:

0 0 0 0 Ba2 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 0 0 0 0 0 0 0
0 De2 Ba5 0 0 Cr2 0 0 0 Ba3
0 0 0 0 0 Cr1 0 0 Fr2 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 Cr2 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 0 Ba4 0
0 0 0 0 0 0 0 Cr1 0 0
```



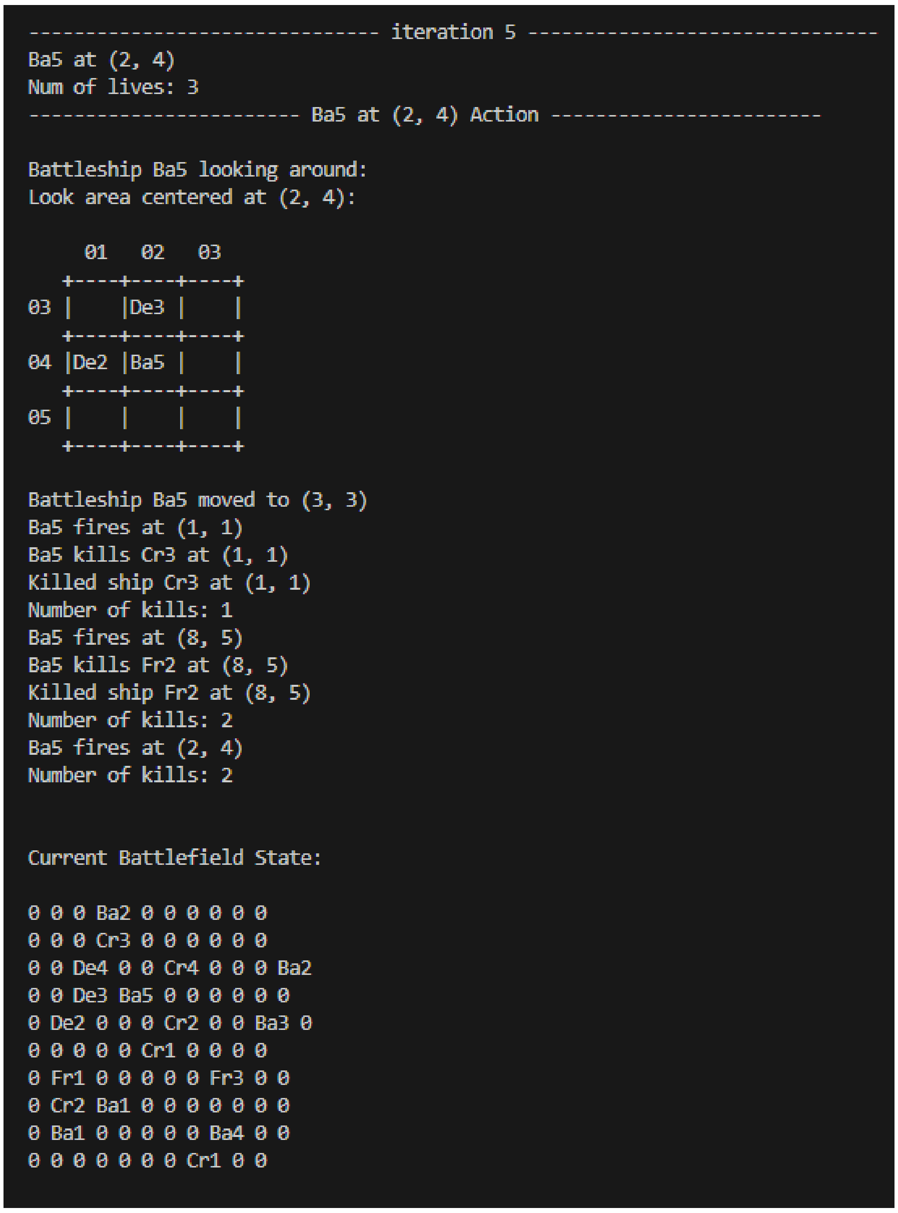
```
------------------------------- iteration 2 -------------------------------
Ba2 at (4, 0)
Num of lives: 3
------------------------ Ba2 at (4, 0) Action ------------------------

Battleship Ba2 looking around:
Look area centered at (4, 0):

     03   04   05
   +----+----+----+
   |####|####|####|
   +----+----+----+
00 |    |Ba2 |    |
   +----+----+----+
01 |Cr3 |    |    |
   +----+----+----+

Battleship Ba2 moved to (3, 0)
Ba2 fires at (0, 0)
Number of kills: 0
Ba2 fires at (1, 6)
Ba2 kills Fr1 at (1, 6)
Killed ship Fr1 at (1, 6)
Number of kills: 1
Ba2 fires at (7, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 0 0 0 0 0 0 0
0 De2 Ba5 0 0 Cr2 0 0 0 Ba3
0 0 0 0 0 Cr1 0 0 Fr2 0
0 0 0 0 0 0 0 Fr3 0 0
0 Cr2 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 0 Ba4 0
0 0 0 0 0 0 0 Cr1 0 0
```



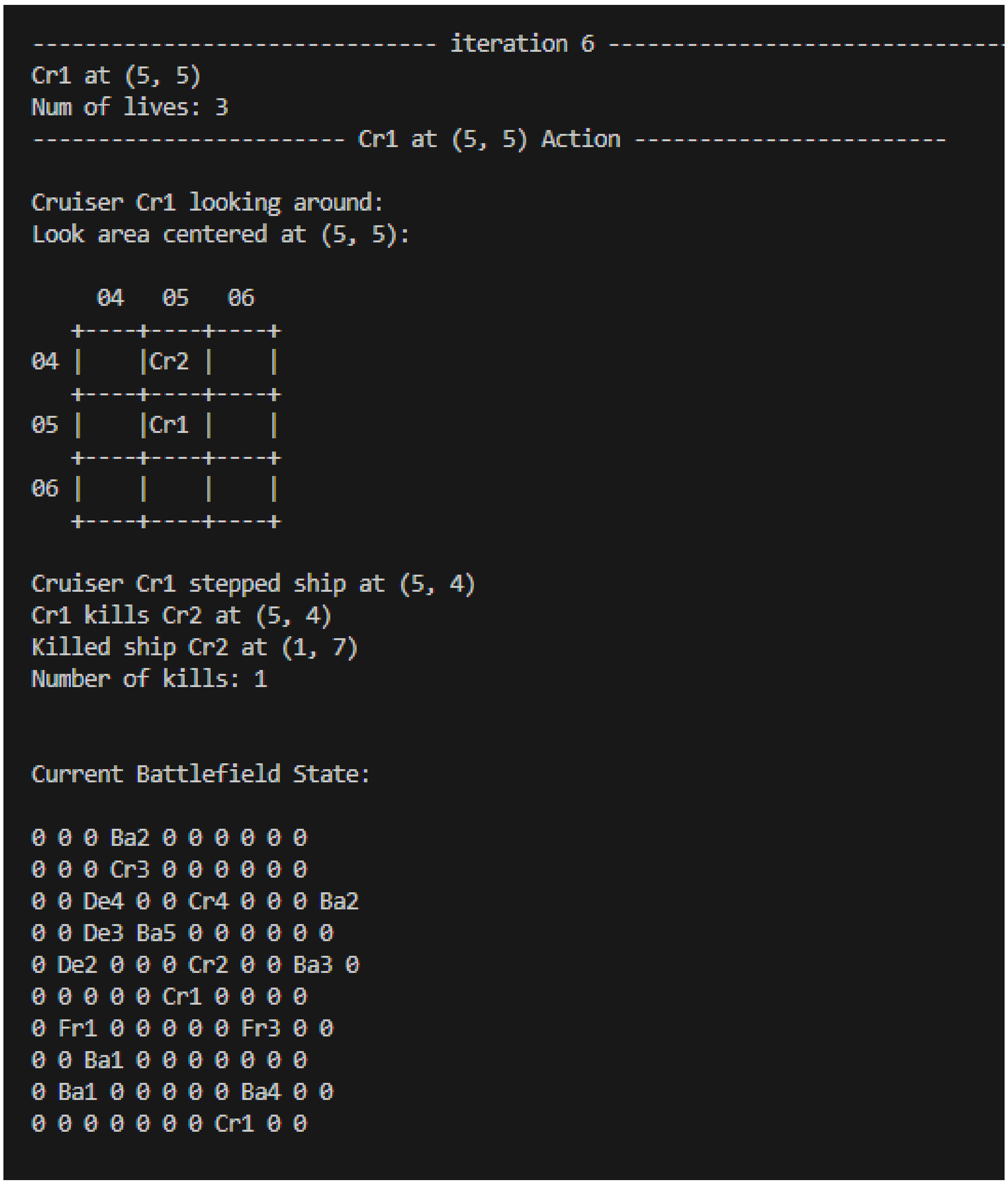
```
------------------------------- iteration 3 ------------------------------
Ba3 at (9, 4)
Num of lives: 3
------------------------ Ba3 at (9, 4) Action ------------------------

Battleship Ba3 looking around:
Look area centered at (9, 4):

     08   09
   +----+----+----+
03 |    |    |####|
   +----+----+----+
04 |    |Ba3 |####|
   +----+----+----+
05 |Fr2 |    |####|
   +----+----+----+

Battleship Ba3 moved to (8, 4)
Ba3 fires at (2, 6)
Number of kills: 0
Ba3 fires at (7, 5)
Number of kills: 0
Ba3 fires at (6, 3)
Number of kills: 0


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 0 0 0 0 0 0 0
0 De2 Ba5 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 0 Fr2 0
0 0 0 0 0 0 0 Fr3 0 0
0 Cr2 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 0 Ba4 0
0 0 0 0 0 0 0 Cr1 0 0
```



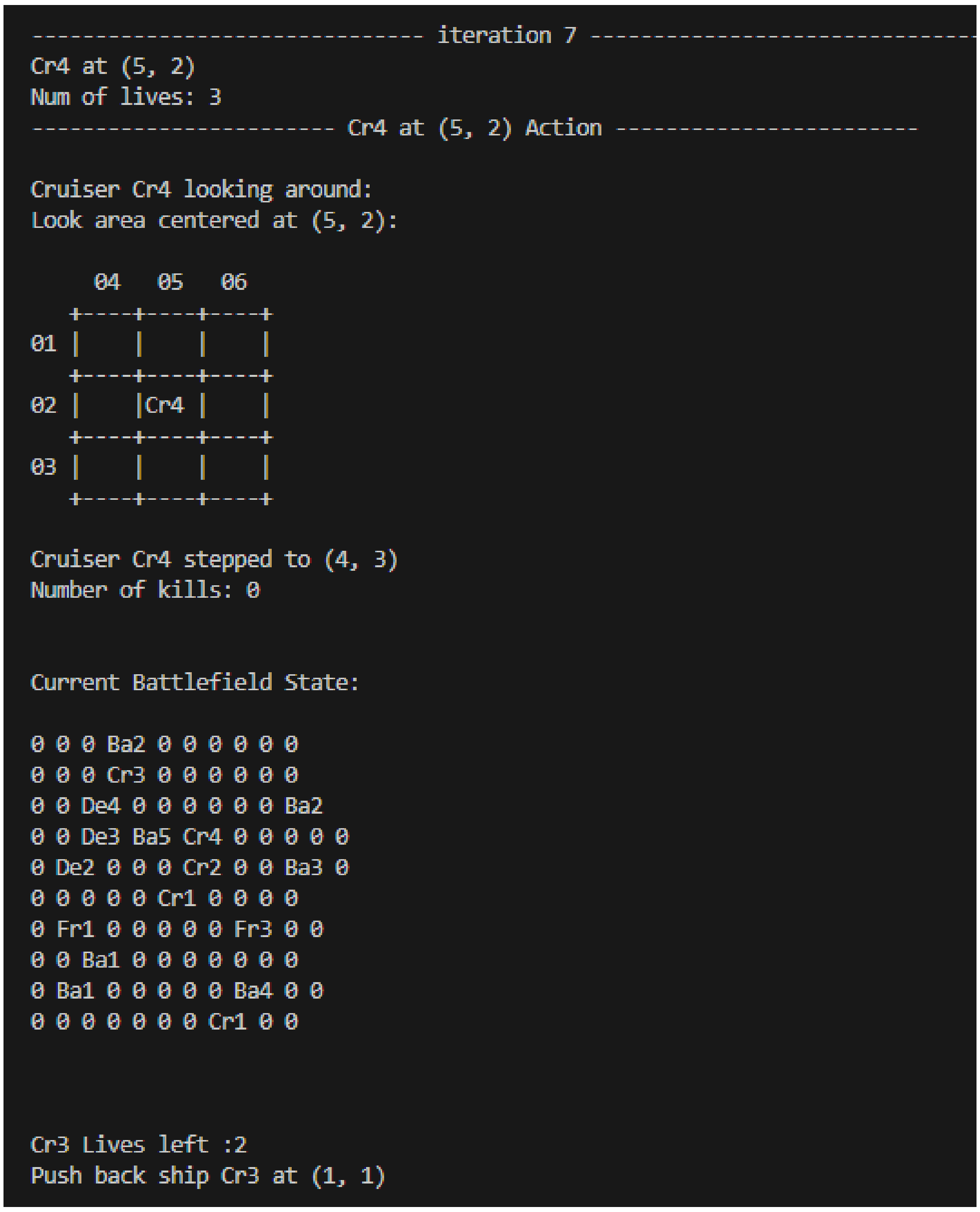
```
------------------------------ iteration 4 ------------------------------
Ba4 at (8, 8)
Num of lives: 3
----------------------- Ba4 at (8, 8) Action -----------------------

Battleship Ba4 looking around:
Look area centered at (8, 8):

     07   08   09
   +----+----+----+
07 |    |    |    |
   +----+----+----+
08 |    |Ba4 |    |
   +----+----+----+
09 |Cr1 |    |    |
   +----+----+----+

Battleship Ba4 moved to (7, 8)
Ba4 fires at (9, 7)
Number of kills: 0
Ba4 fires at (5, 1)
Number of kills: 0
Ba4 fires at (8, 9)
Number of kills: 0


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 0 0 0 0 0 0 0
0 De2 Ba5 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 0 Fr2 0
0 0 0 0 0 0 0 Fr3 0 0
0 Cr2 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0



Fr1 Lives left :2
Push back ship Fr1 at (1, 6)
```



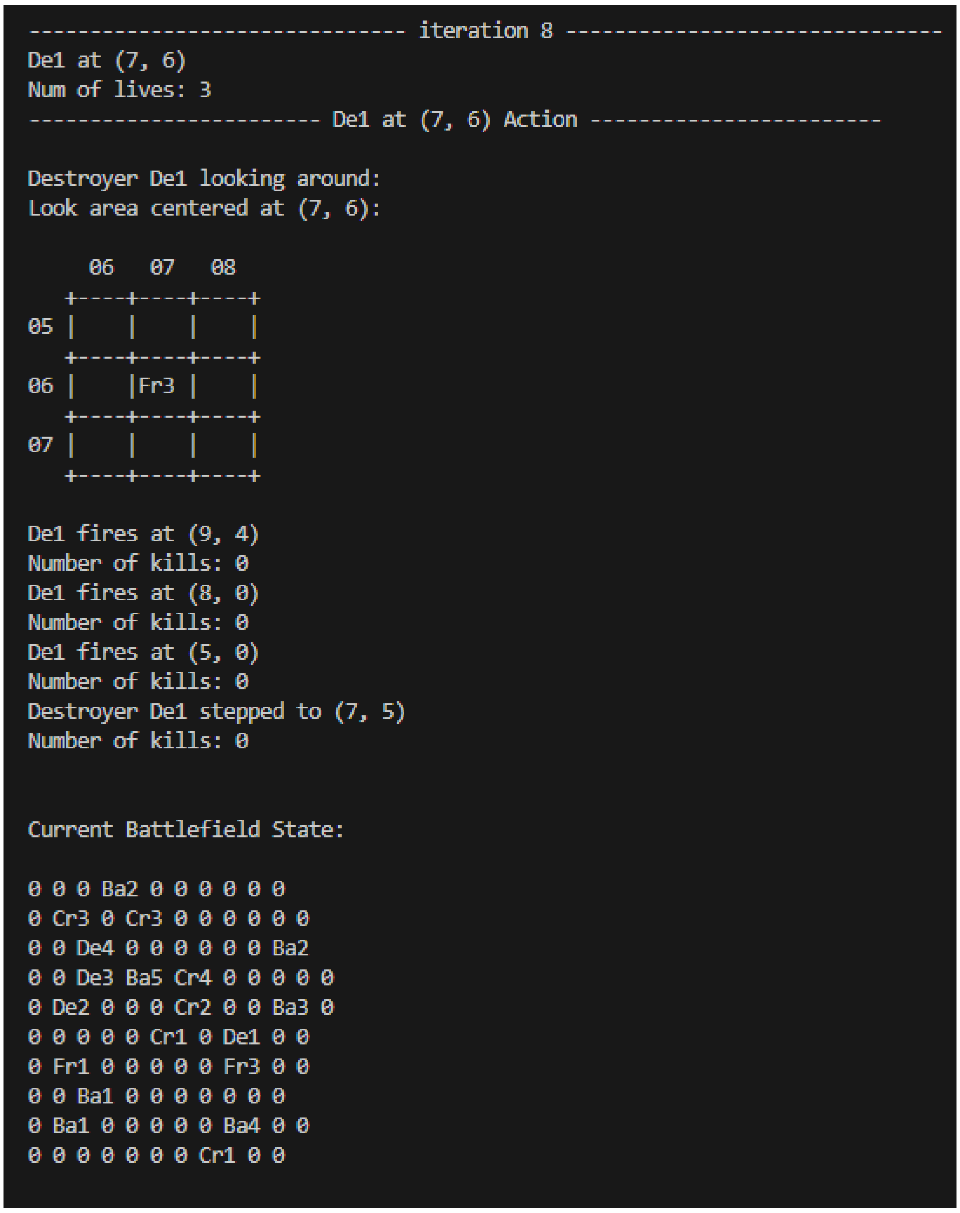
```
------------------------------- iteration 5 -------------------------------
Ba5 at (2, 4)
Num of lives: 3
------------------------ Ba5 at (2, 4) Action ------------------------

Battleship Ba5 looking around:
Look area centered at (2, 4):

     01   02   03
   +----+----+----+
03 |    |De3 |    |
   +----+----+----+
04 |De2 |Ba5 |    |
   +----+----+----+
05 |    |    |    |
   +----+----+----+

Battleship Ba5 moved to (3, 3)
Ba5 fires at (1, 1)
Ba5 kills Cr3 at (1, 1)
Killed ship Cr3 at (1, 1)
Number of kills: 1
Ba5 fires at (8, 5)
Ba5 kills Fr2 at (8, 5)
Killed ship Fr2 at (8, 5)
Number of kills: 2
Ba5 fires at (2, 4)
Number of kills: 2


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 0 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 Ba5 0 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 0 0 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 Cr2 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0
```



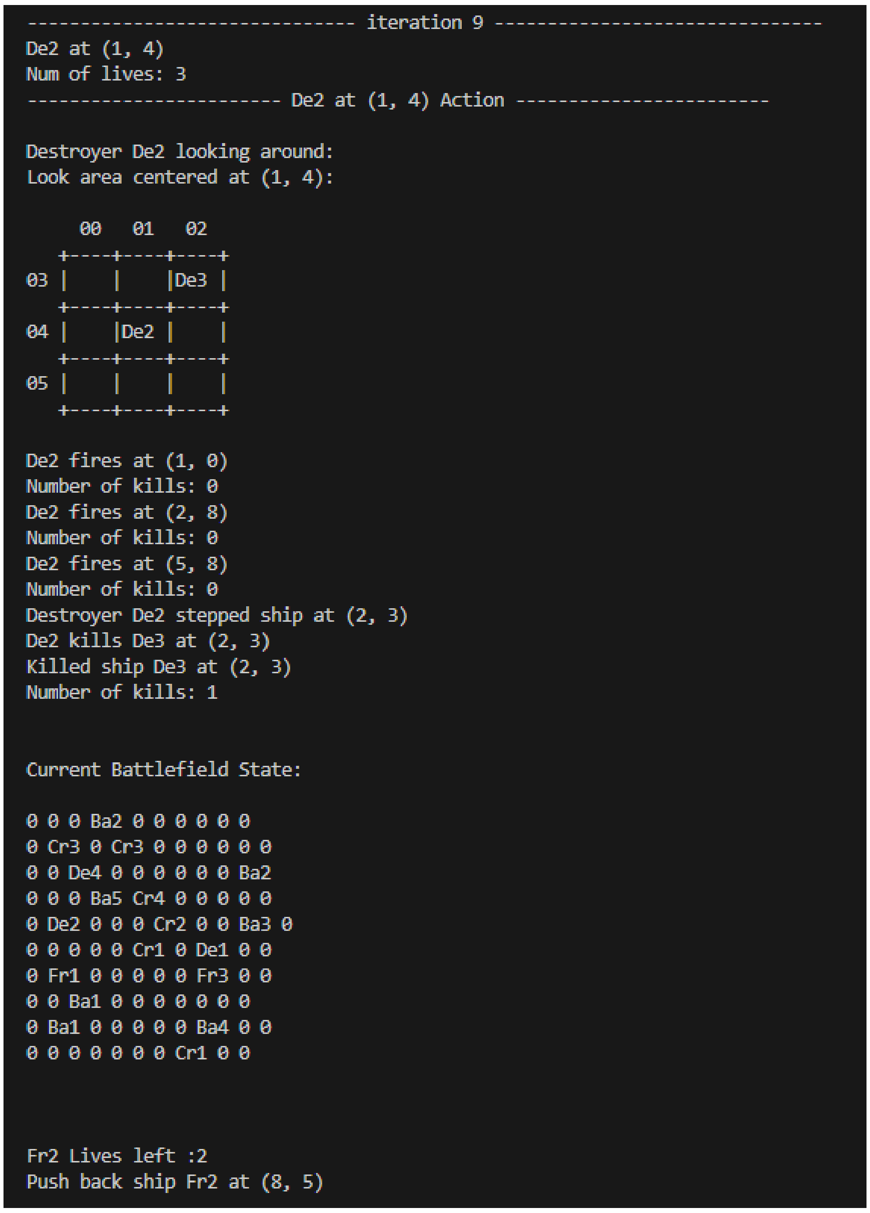
```
------------------------------- iteration 6 ------------------------------
Cr1 at (5, 5)
Num of lives: 3
------------------------ Cr1 at (5, 5) Action ------------------------

Cruiser Cr1 looking around:
Look area centered at (5, 5):

     04   05   06
   +----+----+----+
04 |    |Cr2 |    |
   +----+----+----+
05 |    |Cr1 |    |
   +----+----+----+
06 |    |    |    |
   +----+----+----+

Cruiser Cr1 stepped ship at (5, 4)
Cr1 kills Cr2 at (5, 4)
Killed ship Cr2 at (1, 7)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 0 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 Cr4 0 0 0 Ba2
0 0 De3 Ba5 0 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 0 0 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 0 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0
```



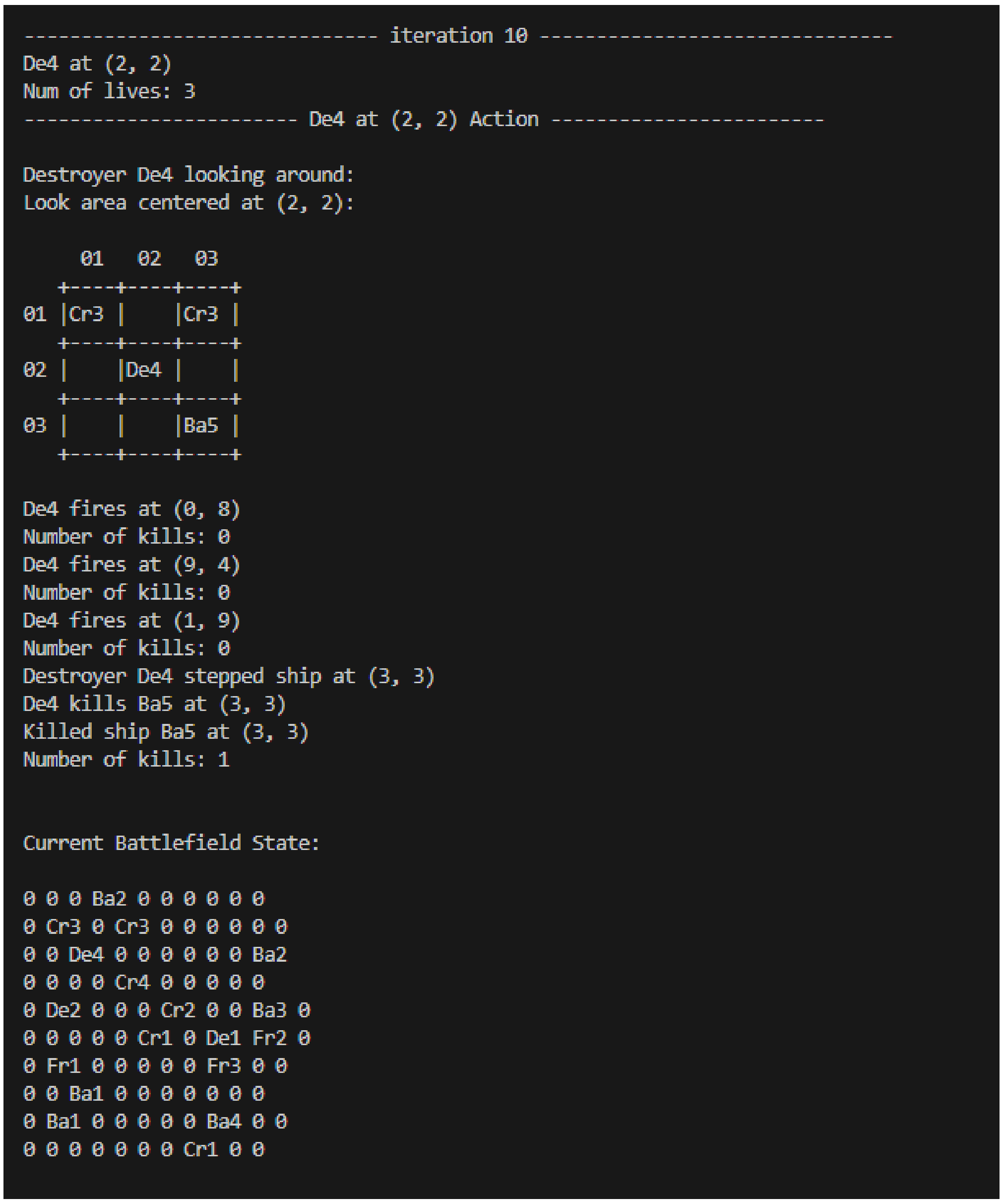
```
------------------------------- iteration 7 ------------------------------
Cr4 at (5, 2)
Num of lives: 3
------------------------ Cr4 at (5, 2) Action ------------------------

Cruiser Cr4 looking around:
Look area centered at (5, 2):

     04   05   06
   +----+----+----+
01 |    |    |    |
   +----+----+----+
02 |    |Cr4 |    |
   +----+----+----+
03 |    |    |    |
   +----+----+----+

Cruiser Cr4 stepped to (4, 3)
Number of kills: 0


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 0 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 0 0 0 0 Ba2
0 0 De3 Ba5 Cr4 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 0 0 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 0 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0



Cr3 Lives left :2
Push back ship Cr3 at (1, 1)
```



```
------------------------------ iteration 8 ------------------------------
De1 at (7, 6)
Num of lives: 3
------------------------ De1 at (7, 6) Action ------------------------

Destroyer De1 looking around:
Look area centered at (7, 6):

     06   07   08
   +----+----+----+
05 |    |    |    |
   +----+----+----+
06 |    |Fr3 |    |
   +----+----+----+
07 |    |    |    |
   +----+----+----+

De1 fires at (9, 4)
Number of kills: 0
De1 fires at (8, 0)
Number of kills: 0
De1 fires at (5, 0)
Number of kills: 0
Destroyer De1 stepped to (7, 5)
Number of kills: 0


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 0 0 0 0 Ba2
0 0 De3 Ba5 Cr4 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 De1 0 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 0 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0
```



```
------------------------------ iteration 9 ------------------------------
De2 at (1, 4)
Num of lives: 3
------------------------ De2 at (1, 4) Action ------------------------

Destroyer De2 looking around:
Look area centered at (1, 4):

     00   01   02
   +----+----+----+
03 |    |    |De3 |
   +----+----+----+
04 |    |De2 |    |
   +----+----+----+
05 |    |    |    |
   +----+----+----+

De2 fires at (1, 0)
Number of kills: 0
De2 fires at (2, 8)
Number of kills: 0
De2 fires at (5, 8)
Number of kills: 0
Destroyer De2 stepped ship at (2, 3)
De2 kills De3 at (2, 3)
Killed ship De3 at (2, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 0 0 0 0 Ba2
0 0 0 Ba5 Cr4 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 De1 0 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 0 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0



Fr2 Lives left :2
Push back ship Fr2 at (8, 5)
```



```
------------------------------- iteration 10 -------------------------------
De4 at (2, 2)
Num of lives: 3
------------------------ De4 at (2, 2) Action ------------------------

Destroyer De4 looking around:
Look area centered at (2, 2):

     01   02   03
   +----+----+----+
01 |Cr3 |    |Cr3 |
   +----+----+----+
02 |    |De4 |    |
   +----+----+----+
03 |    |    |Ba5 |
   +----+----+----+

De4 fires at (0, 8)
Number of kills: 0
De4 fires at (9, 4)
Number of kills: 0
De4 fires at (1, 9)
Number of kills: 0
Destroyer De4 stepped ship at (3, 3)
De4 kills Ba5 at (3, 3)
Killed ship Ba5 at (3, 3)
Number of kills: 1


Current Battlefield State:

0 0 0 Ba2 0 0 0 0 0 0
0 Cr3 0 Cr3 0 0 0 0 0 0
0 0 De4 0 0 0 0 0 0 Ba2
0 0 0 0 Cr4 0 0 0 0 0
0 De2 0 0 0 Cr2 0 0 Ba3 0
0 0 0 0 0 Cr1 0 De1 Fr2 0
0 Fr1 0 0 0 0 0 Fr3 0 0
0 0 Ba1 0 0 0 0 0 0 0
0 Ba1 0 0 0 0 0 Ba4 0 0
0 0 0 0 0 0 0 Cr1 0 0
```