

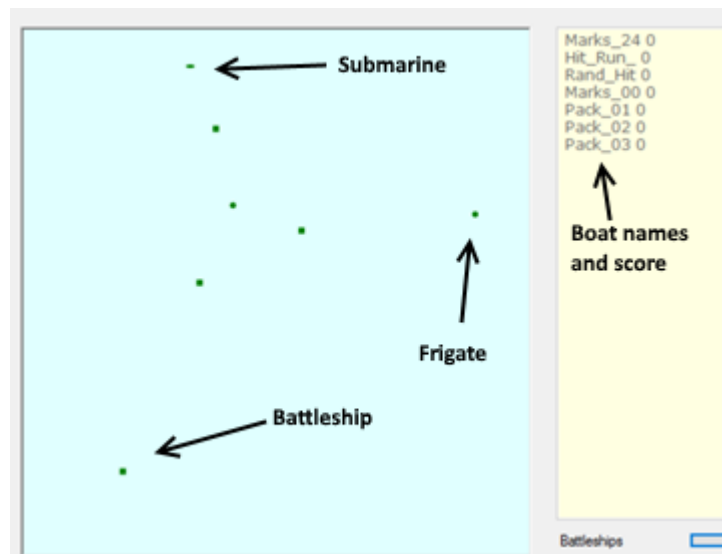
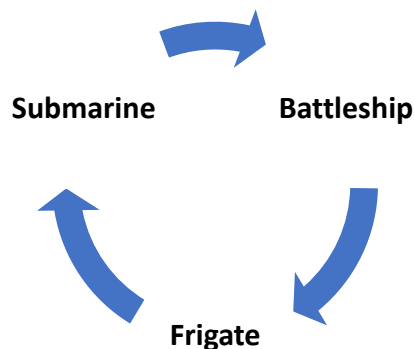
Battleship Bots Documentation

This is a help document for maintenance of the implementation of battleship bots done by *Coobie/Jacob*. The areas covered are listed below.

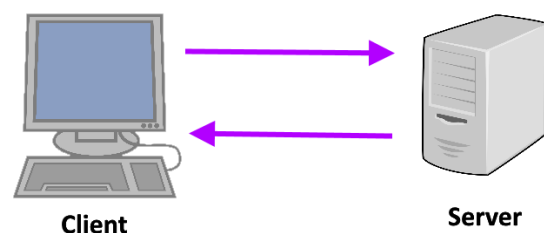
Contents

| | |
|----------------------------------|---|
| Overview of the assignment | 1 |
| Pattern | 2 |
| Spawn..... | 2 |
| Options – two patterns | 2 |
| Interaction with other bots..... | 3 |
| Swap..... | 3 |
| Representation..... | 3 |
| Implementation | 4 |
| Aesthetics..... | 4 |
| Tactics against other bots | 5 |
| Functions..... | 6 |

Overview of the assignment



This overview assumes you are aware of the concepts of this assignment and the purpose of it. There are three types of “boat” in this version of battleship bots. Submarines beat battleships, battleships beat frigates and frigates beat battleships based on equal health. If a ship is on low health it is more vulnerable to the type of ship it should win against. This means that the tactics of this are quite important as the bot needs to know when to attack a ship mostly based on its distance, health and type. The game is based on a server-client communication. For ease of use, the server can be hosted on the same computer as the client.



Pattern

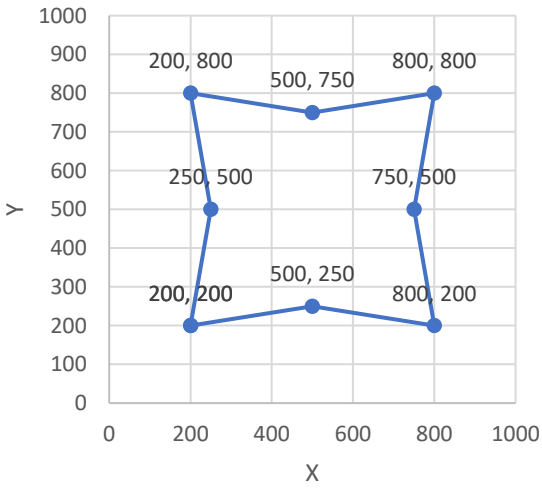
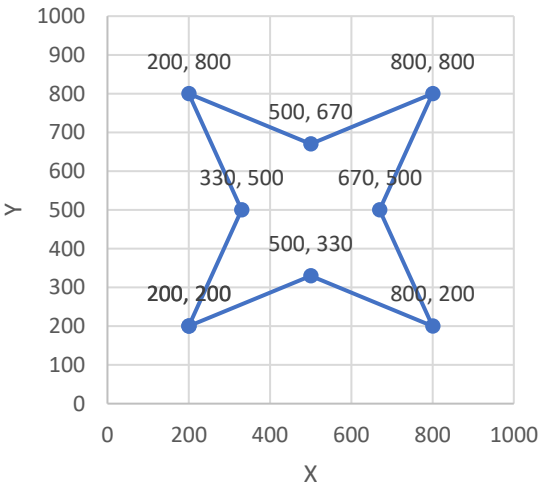
This section is about the route that the bot takes which is referred to as the “pattern”. The coordinates for the points of the pattern are located in arrays in the functions *goToPattern* and *moveOn*. The bot always moves clockwise.

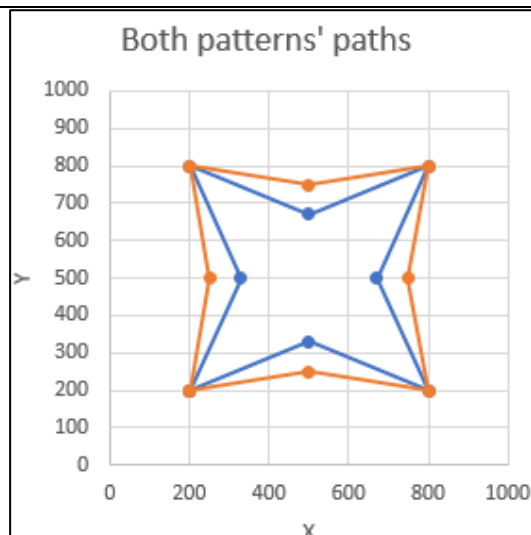
Spawn

On spawn, (detected when current health is greater than the previous health of the bot) the function *spawnPattern* is called. This generates a random number between 1 and 10. This means that the number could be 1, 2, 3, 4, 5, 6, 7, 8, 9 or 10.

Options – two patterns

Based on what occurs on spawn one of two patterns is selected for that life of the bot, details below.

| <h3>Normal Pattern</h3>  | <h3>Extreme Pattern</h3>  |
|--|---|
| <p>This is when the random number which is provided by the function <i>spawnPattern</i> is 1, 2, 3, 4 or 5. This is the less aggressive pattern of the two.</p> | <p>This is when the random number generated by <i>spawnPattern</i> is 6, 7, 8, 9 or 10. This pattern is the more aggressive of the two. This means that it approaches the middle significantly more than the normal pattern.</p> |



This graph shows both patterns plotted at the same time to show the difference between the normal pattern and the extreme pattern.

(Orange is Normal) and (Blue is Extreme). The outer corners of the pattern were located 100 more outwards however this was changed during testing as this did not yield enough kills.

When the bot closes in on one of the points a short timer is started when in the vicinity then the index will change to the next point. This means that there is little waiting between going to each point. This is found in the function *moveOn*.

Interaction with other bots

When another bot is seen, the tactics for dealing with other bots overrides the pattern code. Once the bot has been dealt with it will return to carrying out the pattern. Either that or my bot has died so the respawn code will be triggered. The only circumstance where when the pattern code will still occur when there is another bot is when they are the same type and have equal health. Then the pattern code will continue unless they get within 60 (absolute distance).

Swap

Representation

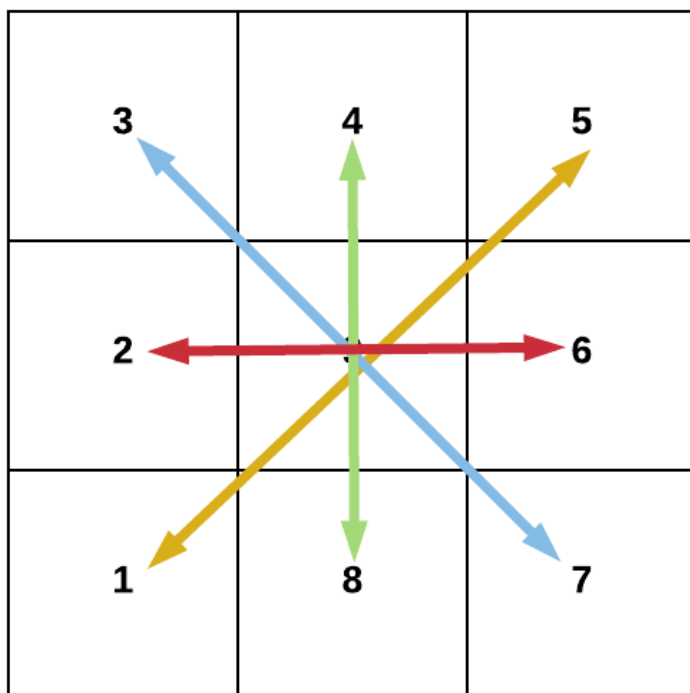
When no enemy has been detected for a while the bot switches sides of the grid system.

| | | |
|---|---|---|
| 3 | 4 | 5 |
| 2 | 9 | 6 |
| 1 | 8 | 7 |

This is a diagram of the zone grid system.

| Current zone | Swap to zone |
|--------------|--|
| 0 | Doesn't swap, waits until in another zone. |
| 1 | 5 |
| 2 | 6 |
| 3 | 7 |
| 4 | 8 |
| 5 | 1 |
| 6 | 2 |
| 7 | 3 |
| 8 | 4 |
| 9 | Doesn't swap, waits until in another zone |

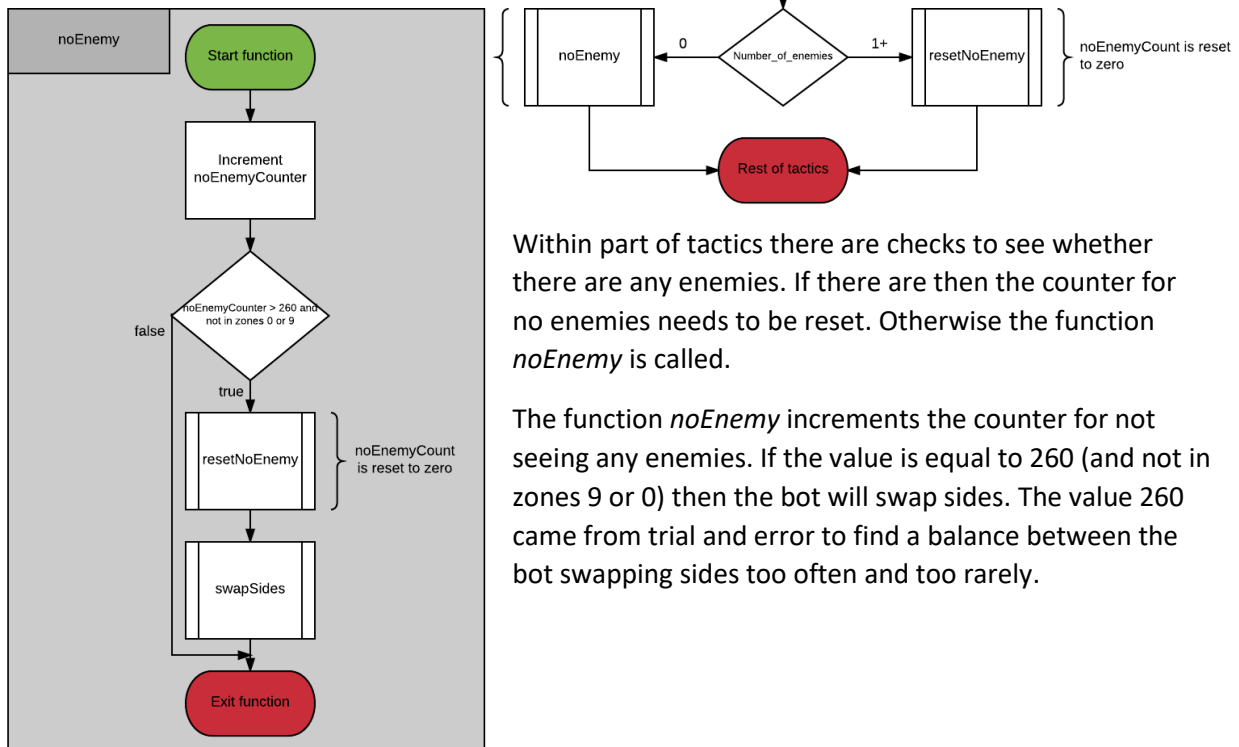
This table outlines the swaps that occur.



This shows a visual representation of the swap. As you can see, the ship will swap sides to the opposite. The purpose of this is to try and find some other battleship bots to shoot.

Implementation

This is the logic behind the swapping side occurrence.



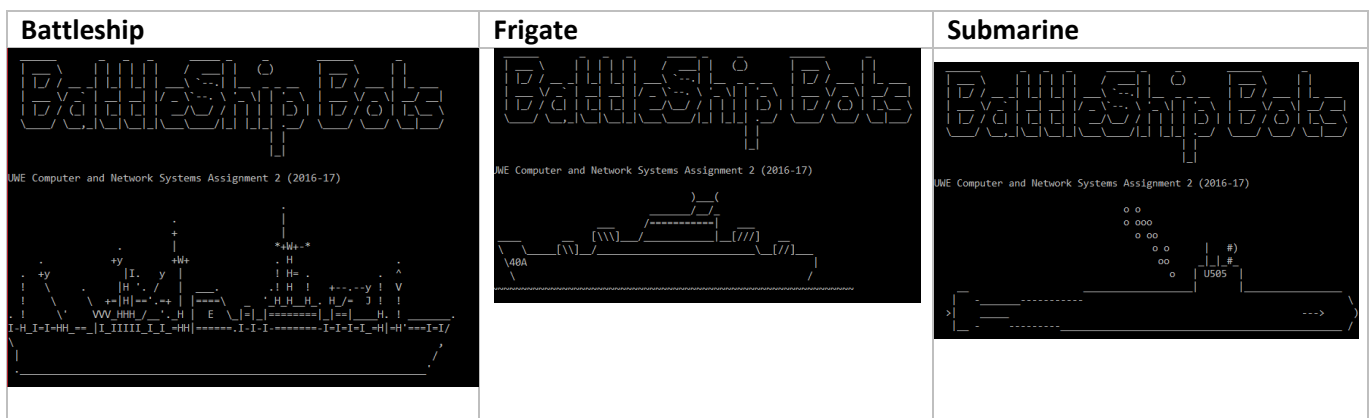
Within part of tactics there are checks to see whether there are any enemies. If there are then the counter for no enemies needs to be reset. Otherwise the function `noEnemy` is called.

The function `noEnemy` increments the counter for not seeing any enemies. If the value is equal to 260 (and not in zones 9 or 0) then the bot will swap sides. The value 260 came from trial and error to find a balance between the bot swapping sides too often and too rarely.

The code for the actual swapping sides can be found in the function `swapSides`.

Aesthetics

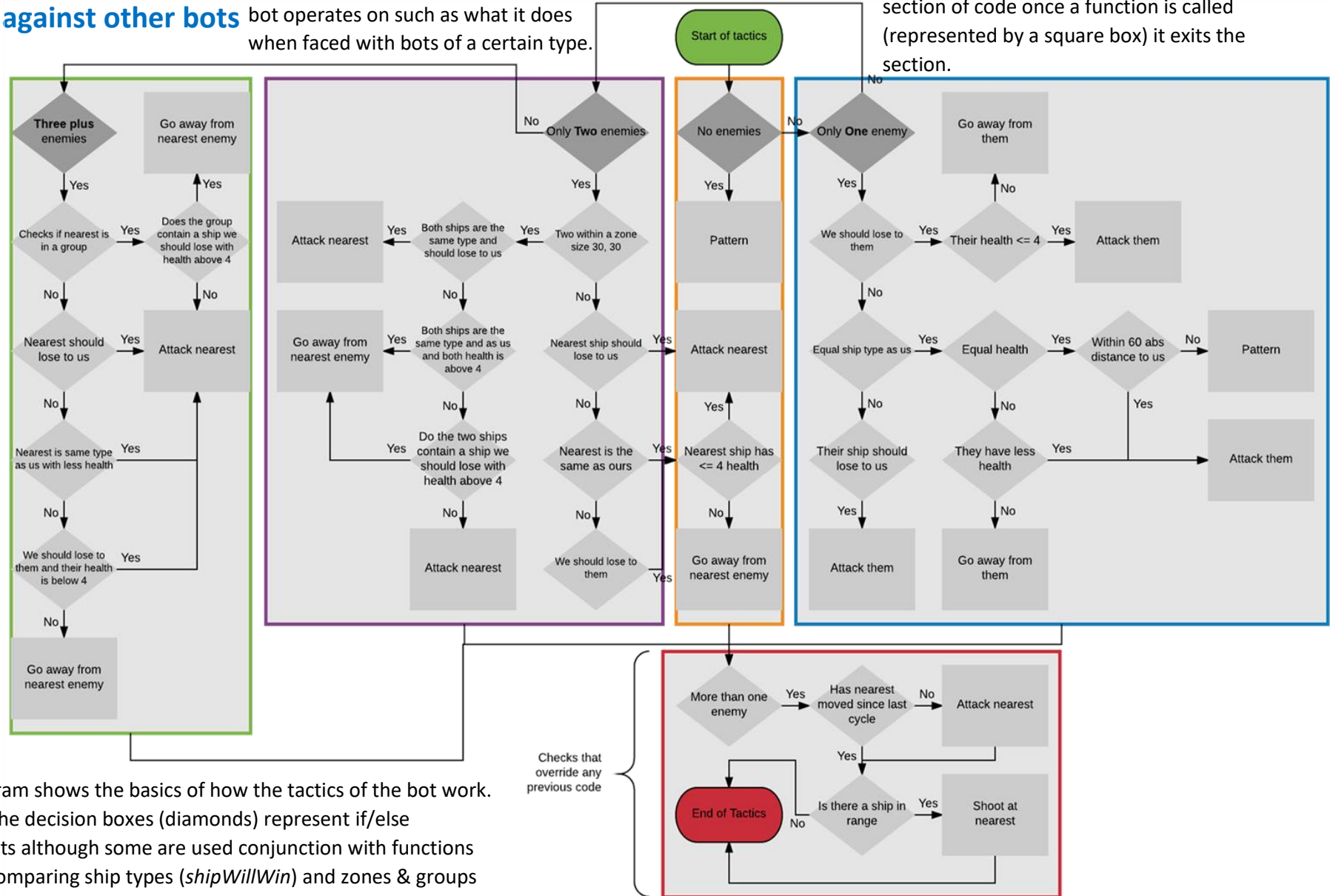
For feedback to the user, when the battleship bot client is started I set it to show a ascii representation of the ship type; examples can be seen below. I have also changed the title on the command line to more clearly show the user what the project is.



Tactics against other bots

This section is about the tactics that the bot operates on such as what it does when faced with bots of a certain type.

For the purposes of this diagram within each section of code once a function is called (represented by a square box) it exits the section.

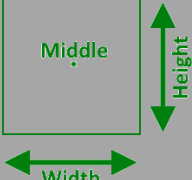


This diagram shows the basics of how the tactics of the bot work. Most of the decision boxes (diamonds) represent if/else statements although some are used conjunction with functions such as comparing ship types (*shipWillWin*) and zones & groups (*inzones*) for example.

Functions

Listed below is a table that contains the function name, a brief description, any parameters of the function, what the function returns and which part of the implementation the function is used in.

| Function name | Description | Parameters | Returns | Linked to section |
|----------------------|--|---|--|--|
| isaFriend | checks whether ship is a friendly, currently set to only return false as I don't want friends. | Index (int) – the ship index that you want to check is a friend or not. | will return true if they are your friend. False if they are not your friend. | Default – pre-existing code which I didn't remove. |
| goTowards | Moves the ship towards coordinates. Only moves "fast". Based on whether the coordinate is larger or smaller than your current position. | x (int), y (int) – the x and y coordinates you want to move towards | Void | Tactics and pattern |
| outputHeader | Outputs a title header composed of ascii text to the command line just for aesthetic purposes. It is set to read from the file Header.txt | none | Void | aesthetics |
| secondaryOuputHeader | Outputs an ascii text picture of the current type of ship to the command line only for aesthetic purposes. Depending on the current ship type, the function either uses Submarine.txt, Frigate.txt or battleship.txt | none | Void | aesthetics |
| inZone | Tells you whether a set of coordinates are within a square | X1 (int), Y1 (int) - x and y coordinates you | Boolean, true if the coordinates are in the zone, false if not. | Tactics and pattern |

| | | | | |
|-------------|---|---|--|---------------------|
| | <p>of a determined size of another set of coordinates. The middle of the square to the outer edge is half of the height or width.</p>  | <p>in see are in zone. zoneX (int), zoneY(int) - x and y coordinates set for the MIDDLE of the zone. width (int), height (int) - The dimensions of the zone</p> | | |
| zone | <p>Outputs the zone which your ship is currently in. 9 zones total. Unfortunately, 1000 (the width and height of the map) doesn't divide by three perfectly so outside of the ranges means you could be in zone 0, however all of the code accounts for this.</p> | none | zoneValue (int) - the zone the ship in currently. 0 to 9 | Tactics and pattern |
| shipWillWin | <p>Returns the most likely outcome between two ships for type1. This means that if type1 is a battleship and type2 is a frigate, then the battleship should win thus the function will return 1.</p> | type1 (int), type2 (int) - The two ship types you want to compare | Returns an int; 1 is win 2 is lose 3 is draw | Tactics |
| goAway | <p>Moves the ship away from coordinates. Only moves "fast". Based on whether the coordinate is larger or smaller</p> | x (int), y (int) – the x and y coordinates you want to move away from | Void | Tactics |

| | | | | |
|--------------------|--|---|---|---------------------|
| | than your current position. | | | |
| nearest_enemy | returns the index of the nearest enemy. | none | nearest_ship (int) - the index of the closest enemy | Tactics |
| most_damaged_ship | returns the index of the most damaged enemy. | none | most_damaged (int) the index of the most damaged enemy | |
| attack | fires at and moves towards enemy. | i (int) - the index of the enemy you want to attack | Void | Tactics |
| measureDistance | Calculates distance between two coordinates. | x1 (int), y1 (int) - first set of coordinates. x2 (int), y2 (int) - second set of coordinates. | output (int) - the distance between the two sets of coordinates. Not sure if this is an absolute value so it would be safe to call abs first. | Tactics and pattern |
| killStationary | Attacks the nearest ship if it has not moved. Useful against ships which are not currently logged in to the server. | none | Void | Tactics |
| setRespawnZone | Sets which point of the pattern to go to first on spawn. Will set the point to the one in the same zone as spawned in or it will choose the closest. | none | Void | Pattern |
| checkTypeAndHealth | Checks if the enemies contain a ship that we would lose to and that if they are their health is above a value. | none | output (boolean) - true is contains ship we would lose to and health is above value 4. | Tactics |

| | | | | |
|----------------|---|---|------|---------------------|
| spawnPattern | Generates a random number on spawn which chooses the pattern. | none | Void | Pattern |
| goToPattern | Moves around the pattern based on the index. | index (int) - pattern index | Void | Pattern |
| moveOn | When the ship reaches a point on the pattern it will move on to the next after a short period has passed. | index (int) - pattern index | Void | Pattern |
| shipInRange | If there is a ship within range shoot it. | ship (int) enemy you want to shoot (normally closest) | Void | Tactics |
| swapSides | swaps sides to go towards for the pattern. Used when there is no enemy for a while then the ship will swap sides. | none | Void | Tactics and pattern |
| resetNoEnemy | resets the counter for no enemy - use for when there are enemies or on respawn. | none | Void | Tactics |
| noEnemy | if there are no enemies the counter increments and moves on if none seen in a while. | none | Void | Tactics |
| flagSetter | Changes my flag to closest enemies if they have one otherwise sets as default friend one | none | Void | Tactics |
| establishShips | In default code this is located at the top of tactics. | none | Void | Tactics |

| | | | | |
|--|--|-----------|------|---------|
| | Just moved out of tactics. | | | |
| tactics | This is where the strategies used by the bot are located. | none | Void | Tactics |
| messageReceived | Checks if the message received is matching the required criteria. All commented out to stop people overloading it. | msg char* | Void | Default |
| Any remaining functions are pre-existing and have not be modified. | | | | |