Report

Wenbo Du

u6361796

Abstract

In general, the requirement of the assignment 1 is completing a part of Haskell code which design a game named Battleship. In the other word, when the code is completed, with the Haskell file Battelship.hs and Main.hs, players can play Battleship using Terminal. In the code I submitted, Preliminaries, Part A and Part B were all finished. However, it still has some problems that I need to improve. In the following article, the procedure of analyses the question and step of compile code will be discussed. Apart from this, the main difficulties that I encountered will be included too.

Design Goal

(1) Preliminaries

This part just used the basic knowledge of case expression (guards can be used too). By compiling the function called cStateToCell , we can translate the data Cell to String , which will be used to generate "really" grids when the players play game. The goal of function shipLength is to indicate Haskell how many grids each ship need occupy.

(2) Part A

The part focus on the generation of Ships, which is defined as [[Bool]] at the top of the file Battleship.hs. As the indication from PAL session, to compile placeShips, the code that I wrote divide to three parts: updateMatrix , updateShips and checkFinished.

- <1> updateMartix

```
updateMatrix::Ships->Coordinate ->Direction -> ShipType ->Ships
updateMatrix a (n,m) dir l = case dir  of
```

This function will swap grids full of False (which represent a map without any ship) to grids that contain 5 ships (Battleship, destroyer, Cruiser, Submarine and Carrier). The main build-in function used here is (!!), drop and take, which was showed in the function updateList. Furthermore, here the function fromIntegral is useful to change Int to Integer. the All operation here derived from this pattern:

```
take m a ++ [take (n) (a!! m) ++ [x] ++ drop (n + 1) (a !! m)] ++ drop (m+ 1) a
```

This can be said as an update of the function updateList. Using this pattern, we can change the exact value in lists of lists. Therefore, changing one “False” to “True” in the type Ships can be achieved. Accordingly, we can change several “False” to “True” in a row or column of Ships. With exact Coordination, Direction, ShipType, the ships can be placed on Grids. For instance, if the Submarine is put on (7,8) and the direction is Left, the “False” on (7,8), (6,8),(5,8) will swap the “True”.

- <2>updateShips:

  updateShipTypes::ShipsOnGrid->ShipType->ShipsOnGrid
  updateShipTypes x ship= case ship of

  his function will filter the random input and finally throw out a list that contain all ships once mentioned on the questions. For instance, if input a Destroyer and a list (Random Inputted) already contain Destroyer, the output will not change since we can only have one Destroyer. Conversely, when the list does not contain Destroyer, the ship should be added to the list. For instance , if we try to put Cruiser on Ships, and we already have Battleship,Destroyer,Submarine,Carrier ,the Cruiser will be added successfully since we do not have Cruiser in the matrix Ships.

- <3>checkFinished:

  checkFinished::ShipsOnGrid->Bool
  checkFinished x

  checkFinished function will check the number of elements in the lists. As the instruction mentioned, when the list ShipsOnGrid have 5 elements, the function will return True the stop inputting random data from Main.hs. if it has less than 5 elements, the process of generating ships will not stop. For instance,when we have Cruiser,Battleship,Destroyer,Submarine and Carrier , the output will be True since the length of list is 5.

  <4>placeShip
  By using the provided function validPlacement, we can filter the valid placement of ship. For instance, (0,2) Left Battlship is not valid placement so the output of validPlacement will be false, the Ships will not change. Then, conbation updateMatrix, updateShips and checkFinished , we can generate a matrix Ships contains 5 different ships and them meet the requirement of the game rules.

- Part B
  This step focus on play mode, and it is not as complex as Part a so that I did not divide it into several parts. Case expression and guards is useful here since there many different situations and we need to apply different operation to them.

  transitionState :: State -> Coordinate -> State
  transitionState (State grid1 grid2 c d) (y,x) = case (State grid1 grid2 t d) of

As the transition process indicated, we divide the transitionState function into two parts the discuss separately:

1 Won or Lost:

This part only need the output to be the same State as input, so the in my code I just used case the return the same value. for example:
(State a b Won d)-> (State a b c d)

2 Playing:

The Playing part needs more discussion.:

(1) when number count reach 20, we need to change condition to Lost
(State a b Playing 20)-> (State a b Lost d)

(2) When the coordinate is out of range the code will return original state.
(x>9 && x<0, y>9 && y<0) is used here to restrict the range.

(3) When the coordinate in Ships is True and the same coordinate is board(important) is Unchecked, the function will change the Unchecked in matrix to Hit.

(4) When the coordinate in Ships is False and the Board contain less than 17 Hit (As the question informed, when All ships are sunk:2+3+3+4+5=17), we change the coordinate point to Miss.

(5) When the coordinate in Broad is Miss and the Board contain less than 17 Hit, we return the same value. (easy to ignore.)

(6) When the Board contain 17 Hit, Change the condition to Win.

For this part to add Miss or Hit to the matrix, I used the pattern below again (replace x with Hit or Miss):

take m a ++ [take (n) (a!! m) ++ [x] ++ drop (n + 1) (a !! m)] ++drop (m+ 1) a

another session is to count the number of Hit. for this, I tried the used the combination of Higher order function in Haskell:
sum (map length (map (filter (==x)) a),
where a represent a Matrix, and x represent an element in Matrix. In the doctest , I gave some example than will test whether the function work correctly(however, the matrix is not 10*10.)

Results

After several days work, my code passes all 4 tests on Pipeline. However, it still has 2 warning that I cannot not fix currently. This the following text will contain two main part: 1. the difficulty and error that I encountered and how I solved them. 2.things need to be improved.

1. the difficulty and error

- The understanding of placeShip function.

Previously, as many classmates, I am confusing about the meaning of placeShip function. I just do not understand the process that “utilised the IO type in the Main.hs file in order to take care of randomness generation” (Although to write the function we do not need to know the meaning of this). This time PAL helps me. On the Study Event, PAL give us some indication and let us the draw the process of

placeShip by ourselves. Without drawing a map satisfy the requirement of Battleship rules, it is pointless the focus on the next steps.

- swap False to True (also Unchecked to Hit of Miss).

  The function updateMatrix was hard to write. The only that I have thought is how to update a list. Fortunately, updateList function told me how to do this. I noticed the build-in function “take” “drop” and “!!” and googled the basic usage of them. Then, just compute the pattern “take m a ++ [take (n) (a!! m) ++ [x] ++ drop (n + 1) (a !! m)] ++drop (m+ 1) a”, which use many times in my code. To fully use this pattern, I can change several coordinates in a matrix.

- Integer and Int.

  I did not understand the different between Integer and Int previously and how to swap Int and Integer. That is, the use of fromIntegral. So, every time I want to run my code it throws out the error message: “Couldn't match `Integer' with actual type `Int'”. That is because the type Coord is defined as Integer at the top of Battleship.hs, but the type I input is Int. First, I just change all Integer in Battleship.hs and Main.hs into Int. It works fine locally, but it cannot pass the compilation test on Pipeline (since I changed Main.hs function). I ask for help from friend, he informed me the use of fromIntegral function can be effective.

- How the count the numbers of Hits?

  This question about the Won condition that all ships are sunk really stunk me for a long time. The problem I am facing is that I am unfamiliar with higher order function. When I review the lecture video and try to rewrite some code by myself, I finally figure out the combination of some build-in function can be helpful (sum, filter, length, map). For instance, just get a simple matrix such as “[[1,2], [1,2]] “and use it to try how to compute a function that figure out how many 2 it has. Finally, apply it to the really matrix I need to face.
- Condition to change coordinate.

  When I tested my code on Pipeline, it failed play mode and comprehensive. I think 1 hours but did not figure out how to pass it, so I post the error massage on Piazza. Someone help me with the indication about game rules: If the target coordinate has already been checked, the move counter increases by one, and everything else remains unchanged. That is the core part that I ignored! Also, I find it is useful to analyse the error massage. That is, do the process of test step by step by myself, and this will assist me the find the exact line and column I did wrong. To be specific, in the function transitionState, use “&&” to make ensure only when the coordinate in Ships is True and the coordinate in Board in Unchecked, the coordinate in Board will change to Hit.

- Warning solving

  I encountered 2 kinds of warning massages. First is named Pattern match(es) are non-exhaustive. I spent lots of time to solve the problems. When I type this problem into stackoverflow, I find lots of people have the same kind of issues since we do not

consider all the situations. Using "_" I solved this because "_" consider all the other situations that is no used in my functions. Second warning massage is: "This binding for `otherwise' shadows the existing binding." By post the problem on Piazza, I get the indication that it means that I have variables defined in the same name, so I just changed the name of variables.

2. things need to be improved:

- Speed/Effiency
  My updateMatrix function needs to be improved. In my code, though I finished preliminaries, the function such as shipLength, showBoard, they were not used in updateMatrix and placeShips since until I finished my code, I did not notice these useful function (including updateList and getCoord). That means these functions is redundant in for me Battleship.hs. At the same time, using ":set +s" in ghci, I found some of my function is slow. If I tried to use these functions effectively, Part A and Part B will be much simple, clear and not as long as I wrote (though the theorem is easy to understand). Furthermore, I believe more higher order function like map and filter can make the code simpler too.

- Studying Skills

  It is the first time I learn how to write code. In my view, the most important I learned from doing the assignment about the courses is about how the keep balance between asking for help and think independently. I find it is completely different between understanding a function and write it by myself. I can understand majority of code on lectures, however, when I tried to write them by myself I have to spend a lot of time about how to compute them. Furthermore, it is necessary to find help on Google or others such as PAL session. It is wrong the spend all day writing a function without any help, because sometime that just need a new build-in function that you do not know or unfamiliar with. At the same, thinking does matter too. If I just read and understand the code from others and never tried to do them, I will never "create myself" and develop my programming ability.

Conclusion

In conclusion, in this report, I mainly showed the process of my function design and what kind of knowledge I used. Furthermore, I gave the description of some serious problems I faced during writing the function and how I solved them. Then, the drawback of my code and the goals that I did not achieve was listed too. Finally, I describe what I learned when I did my assignment 1. By this assignment, my understanding of Haskell programming, such as case, guards, list and higher order function became clearer and deeper. In future, I think one day I will be able to understand the whole code in Battleship.hs and Main.hs and try to write them by myself.