C3C Scarlet Friend

Requirements:

-shall display an introduction message including the rules of the game

-shall automatically generate a game board for the computer

-shall allow the player to manually configure their game board

-shall include 5 crafts for each game board (the c-17, A-10, B-2, F-16, osprey)

-shall clarify the symbols and what they represent such as a hit and miss

-shall contain an array that is the game board (array)

-shall use define directive to set constant macros for the rows and columns of the game board

-shall print the game board to the screen

-shall use structs to keep track of crafts, position, and target (Structs)

-shall use a string to get the Players input of a target (strings)

-shall update the board on hits and misses in between turns

-shall insure crafts are not on top of each other

-shall inform player if a mistake has been made that needs to be fixed

-shall create cases for all the aircraft this will be done through pointers to save the memory of each craft (pointers)

-shall constantly check to see if a craft is shot down (sunk)

-shall include a file to log the player’s aircraft history (File I/O)

-shall display a winner

Main()

Intro\_screen (5%)

Design\_Board(10%)

print\_board(5%)

Craft\_SetUp(10%)

Player\_Crafts (15%)

StringToPosition(5%)

Random\_position (5%)

Computer\_crafts(5%)

Game\_Play(5%)

Target(15%)

Check\_status(15%)

Display\_winner(5%)

-Intro\_Screen

>Print statements of the rules of the game

>Establish there is one player and the computer is the other

>Clarify the [C]-17 (5), [A]-10 (4), [F]-16 (3), [B]-2 (2), [O]sprey (5). Brackets are the symbol parenthesis are the spaces

-Design\_Board

>an array -> ‘board’

>define rows and columns

-print\_board

> just prints the above board

-Craft\_SetUp

>an array to determine where the craft will go and which craft it is

>set that coordinate as the starting position

>initialize i as a the length of that aircraft

>a direction of either [h]orizontal or [v]ertical

>a loop that checks for a condition of h or v, use the starting position to place the craft for the appropriate length

-Player\_Crafts

>begin with an initial 0 for the number of crafts and continue a for loop until the number of crafts has been reached

>Use a printf statement to ask the player for cells and craft they want their aircraft to be placed in

>these will be referred to as craft[i].name and craft[i].size

>use a scanf statement to gather the cell position of the starting cell as a string

-StringToPosition

>there will be three parameters of the position, the string, and the ship length

>create a for loop with variable k set at zero and leaves when the craft[i].length is satisfied

>create an if statement allowing the craft to be placed in the air NOT on top of other crafts (return error if another craft is there)

>Make the player reenter the position

-random\_position

>create if loop with the condition being h for horizontal to keep the row the same and to change the column number

>create if loop with the condition v for vertical to keep the column the same and change the row number

-Computer\_crafts

>create a loop for until the number of crafts is satisfied

>randomly generate a number 0-1 where 0 is horizontal and 1 is vertical

>generate a position by using the direction and the size of craft

-target

>printf statements asking for the coordinates of the wished location

>for player they are inputted manually the computer will generate

>the input is the target.row and target.column

>the target is then determined to be a hit or miss

>if it is a hit then that is added to the hit counter for that player

-Game\_Play

>has to have the cell from the player

>addresses all the cases

>if missed update the air cell with an [M]

> if it is a craft update with an [X]

-check\_status

>parameters include a file to log the shot down aircraft, which player, the number of crafts, the symbol of the craft

>check the craft symbol to assess the case

>C-17 : c

>F-16 : f

>A-10: a

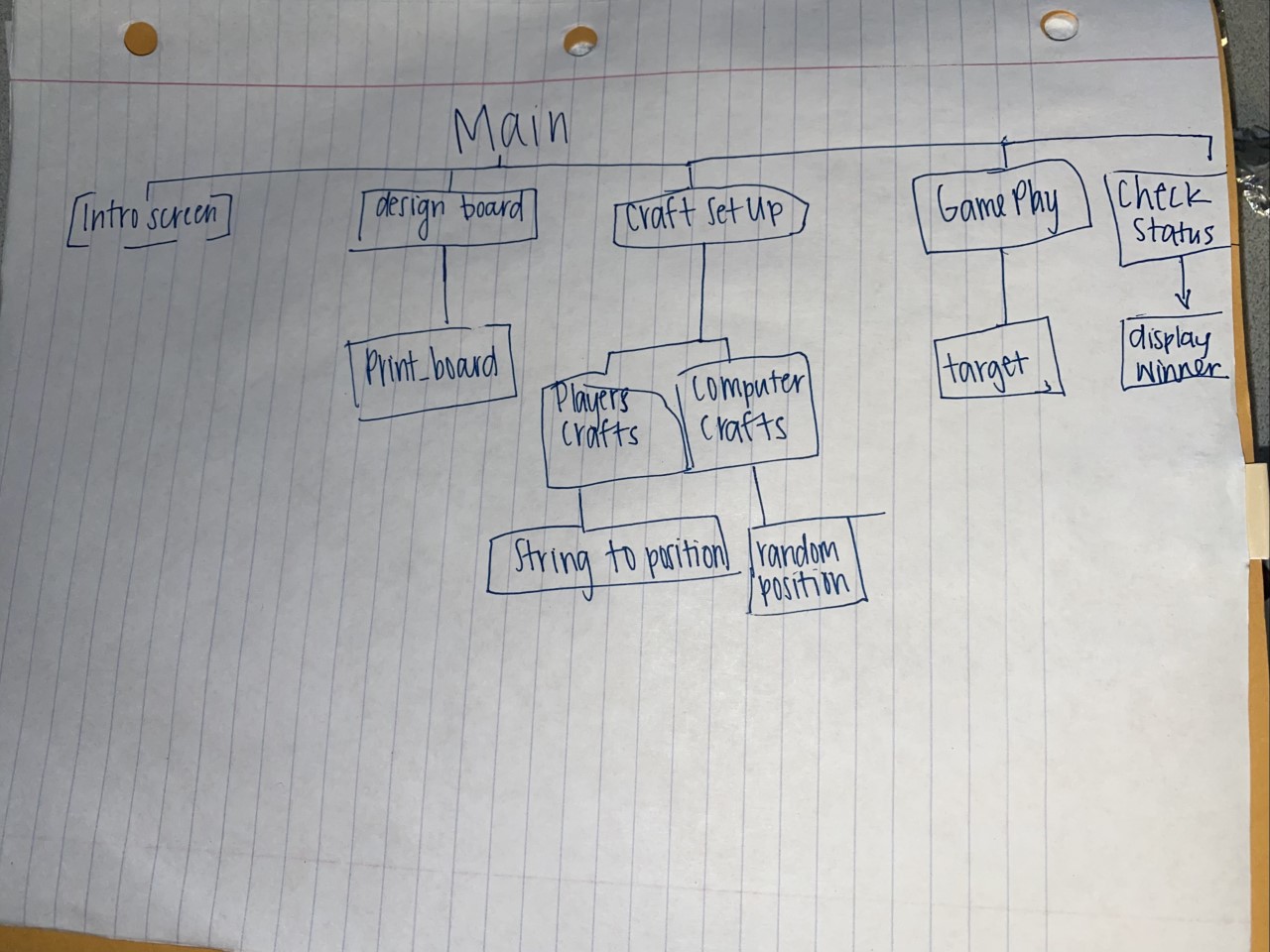
>B-2: b

>Osprey: o

>if those symbols == 0 then report the appropriate craft as Shotdown to the log(fprint)

-display\_winner

>Once one player has all 17 HITs they are the winner and a printf statement ends the game



Documentation: I referenced zybooks to make sure I understood all the requirements. I reviewed the lesson 16 slides. I also looked over battleship rules to start brainstorming how it would work