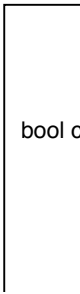
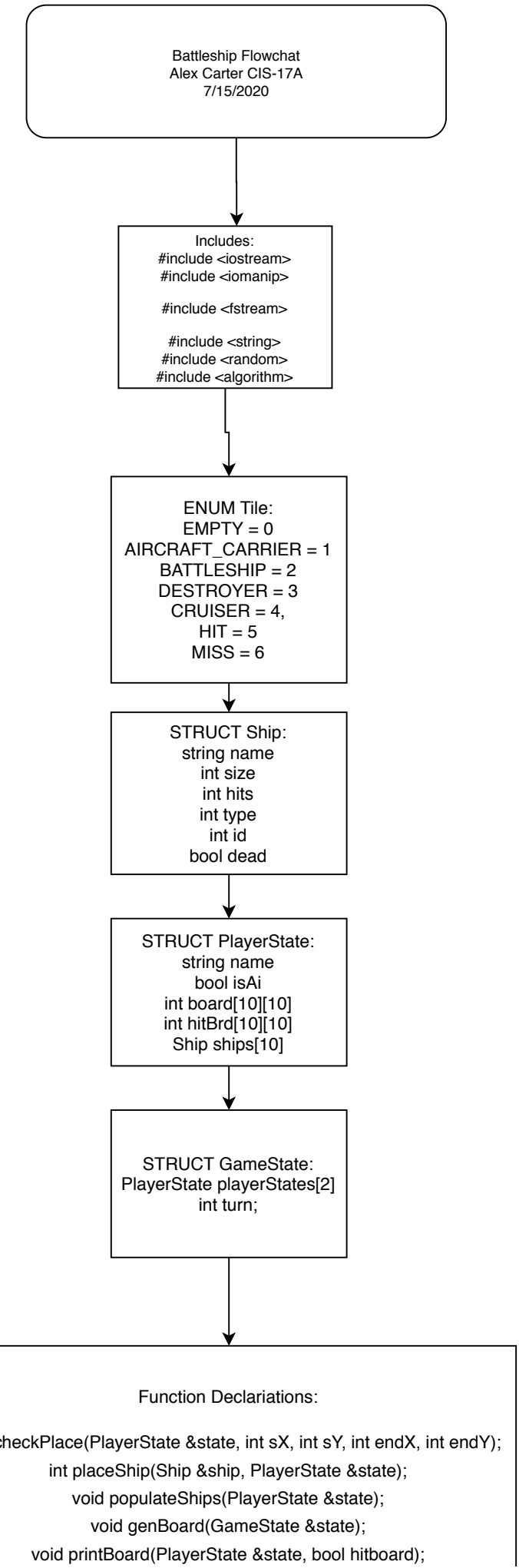
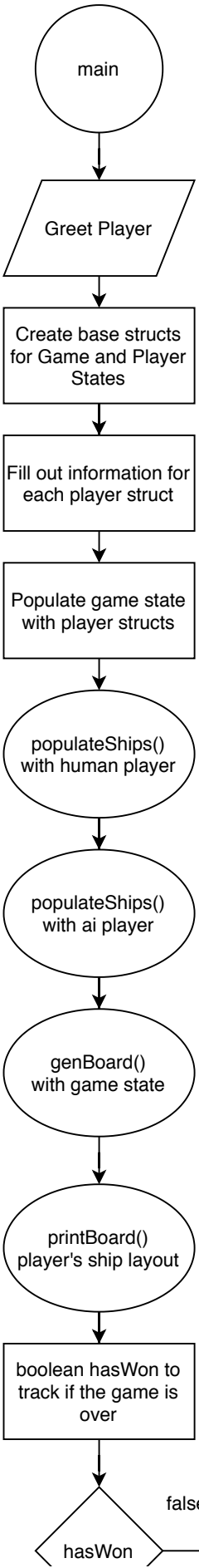
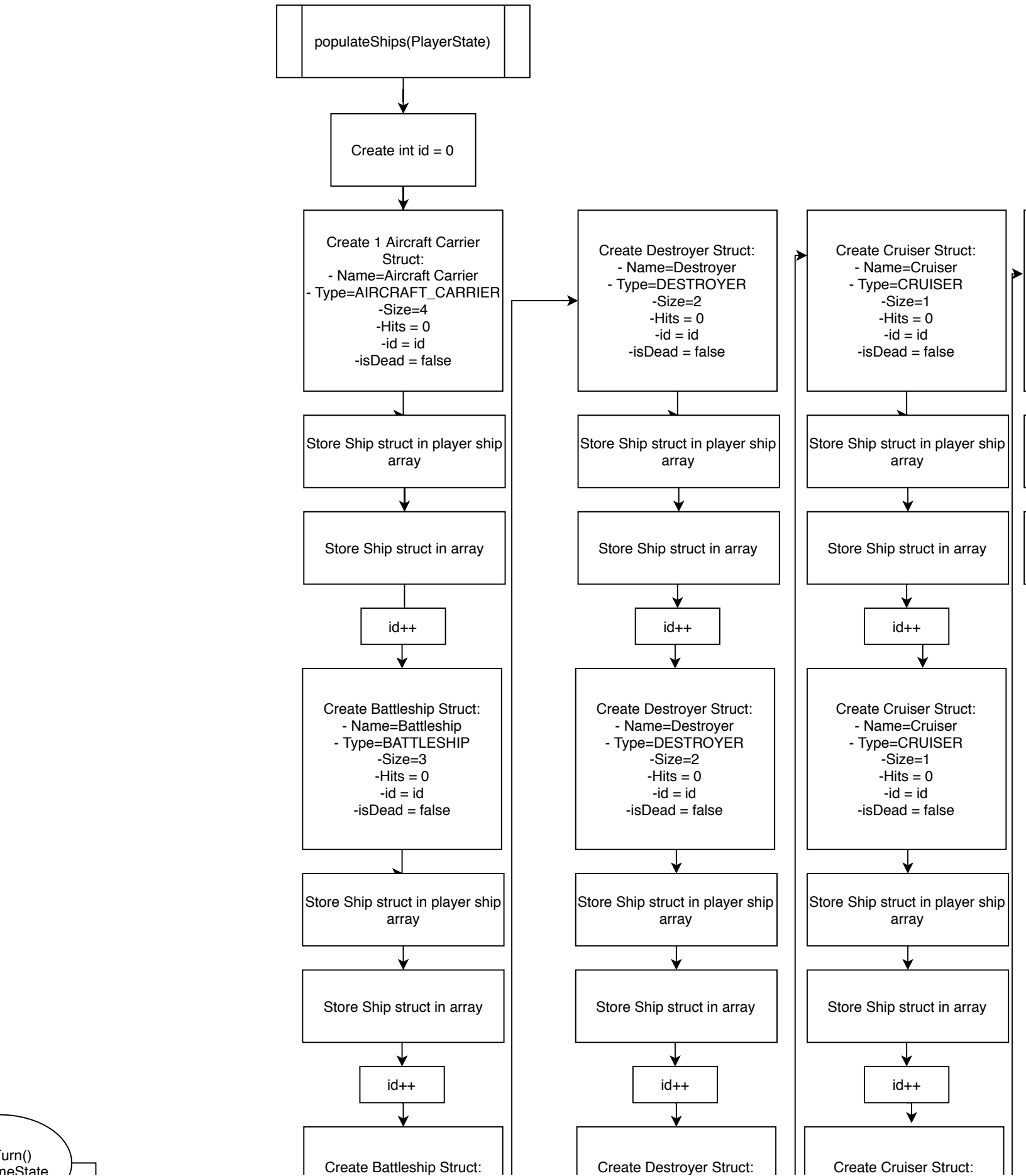
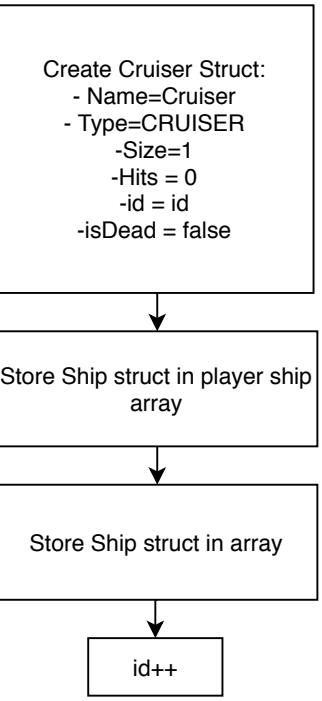
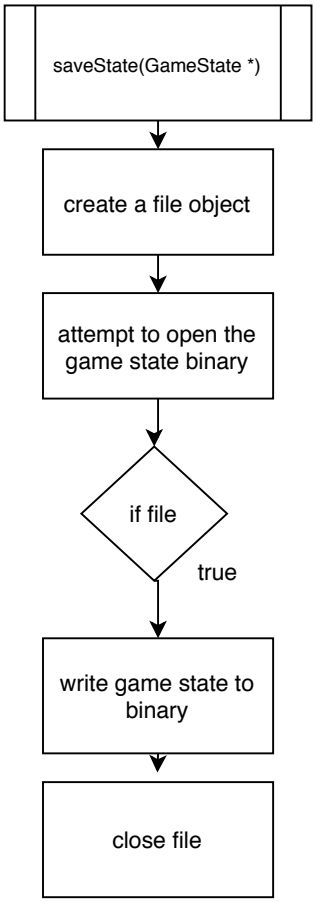

bool c

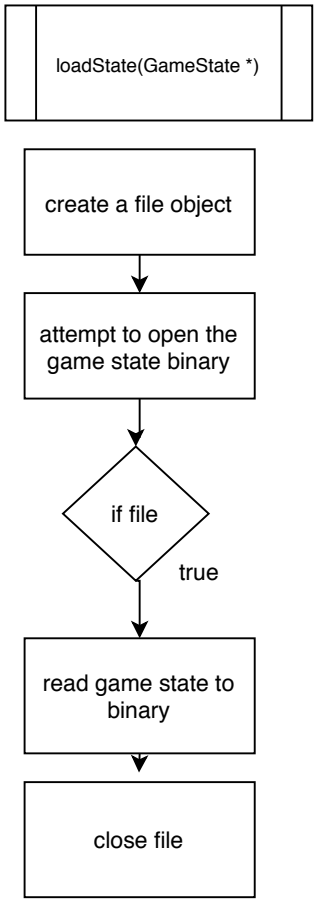

Battleship Flowchat
Alex Carter CIS-17A
7/15/2020
Includes:
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <random>
#include <algorithm>
ENUM Tile:
EMPTY = 0
AIRCRAFT_CARRIER = 1
BATTLESHIP = 2
DESTROYER = 3
CRUISER = 4,
HIT = 5
MISS = 6
STRUCT Ship:
string name
int size
int hits
int type
int id
bool dead
STRUCT PlayerState:
string name
bool isAi
int board[10][10]
int hitBrd[10][10]
Ship ships[10]
STRUCT GameState:
PlayerState playerStates[2]
int turn;
Function Declariations:
heckPlace(PlayerState &state, int sX, int sY, int endX, int endY);
int placeShip(Ship &ship, PlayerState &state);
void populateShips(PlayerState &state);
void genBoard(GameState &state);
void printBoard(PlayerState &state, bool hitboard);

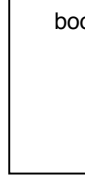

main
Greet Player
Create base structs for Game and Player States
Fill out information for each player struct
Populate game state with player structs
populateShips() with human player
populateShips() with ai player
genBoard() with game state
printBoard() player's ship layout
boolean hasWon to track if the game is over
false
hasWon
takeT
with gar

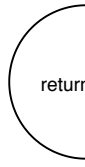

populateShips(PlayerState)
Create int id = 0
Create 1 Aircraft Carrier Struct:
- Name=Aircraft Carrier
- Type=AIRCRAFT_CARRIER
-Size=4
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
Create Battleship Struct:
- Name=Battleship
- Type=BATTLESHIP
-Size=3
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
Create Destroyer Struct:
- Name=Destroyer
- Type=DESTROYER
-Size=2
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
Create Destroyer Struct:
- Name=Destroyer
- Type=DESTROYER
-Size=2
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
Create Cruiser Struct:
- Name=Cruiser
- Type=CRUISER
-Size=1
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
Create Cruiser Struct:
- Name=Cruiser
- Type=CRUISER
-Size=1
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
urn()
neState

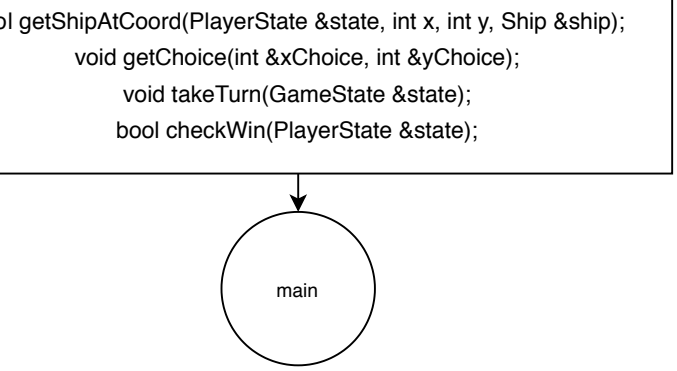

Create Cruiser Struct:
- Name=Cruiser
- Type=CRUISER
-Size=1
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++


saveState(GameState *)
create a file object
attempt to open the game state binary
if file
true
write game state to binary
close file

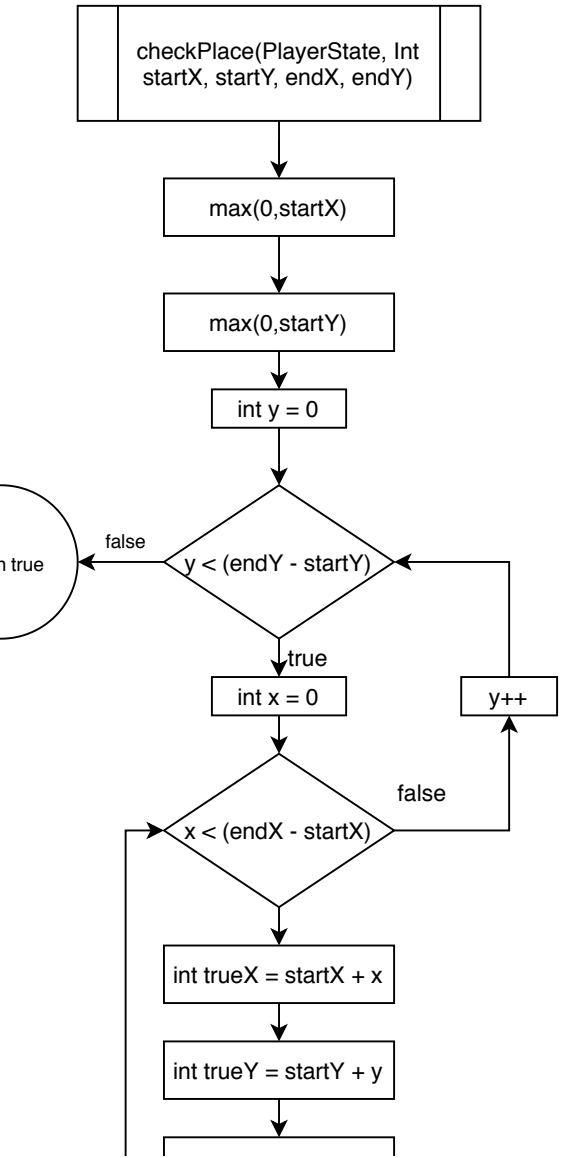

loadState(GameState *)
create a file object
attempt to open the game state binary
if file
true
read game state to binary
close file

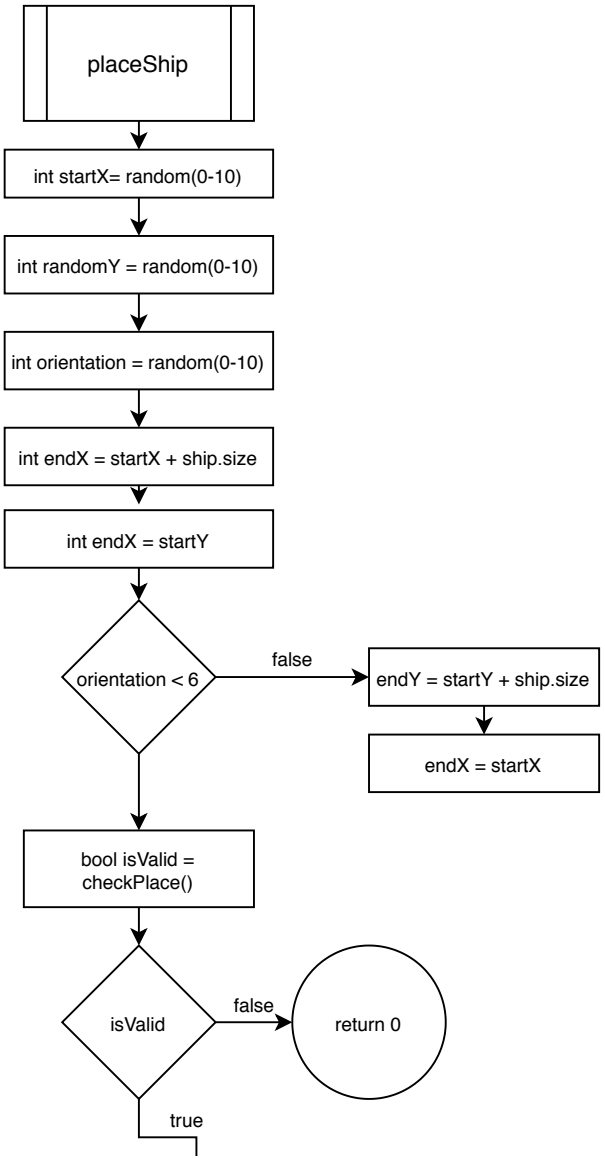

boo

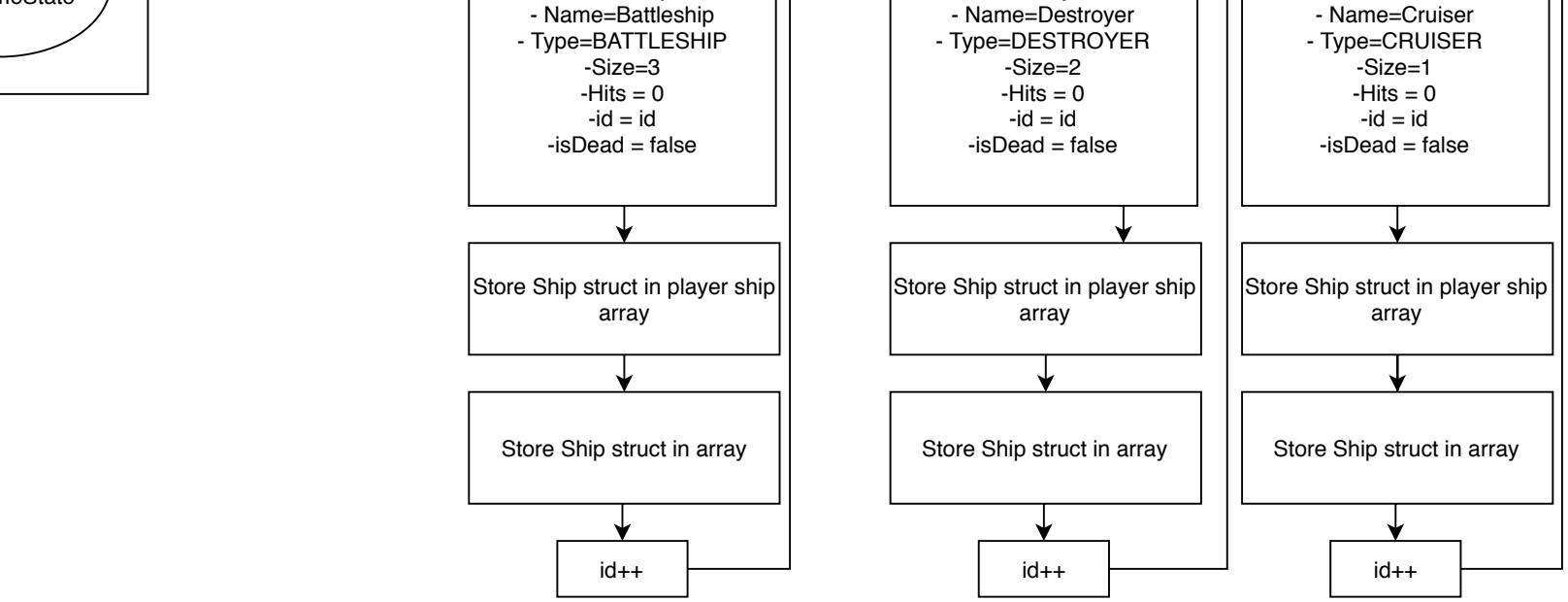

return

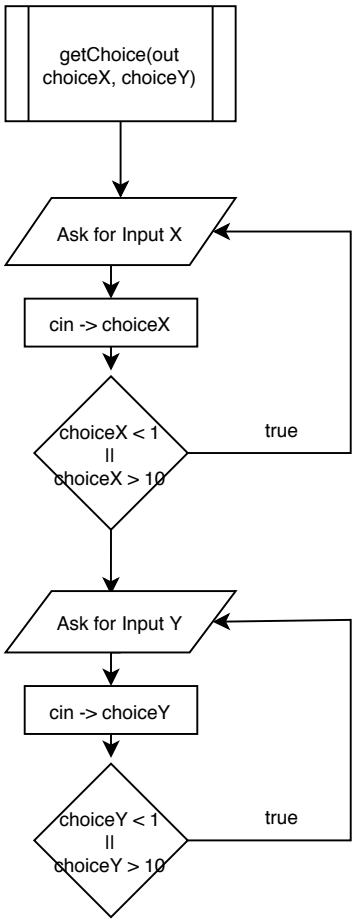

ol getShipAtCoord(PlayerState &state, int x, int y, Ship &ship);
void getChoice(int &xChoice, int &yChoice);
void takeTurn(GameState &state);
bool checkWin(PlayerState &state);

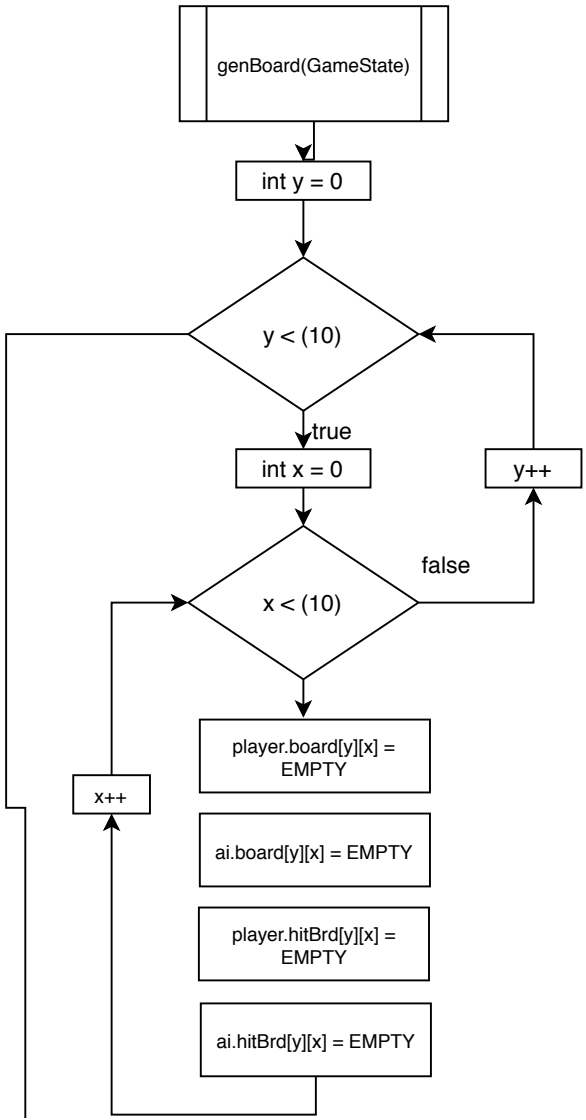

main

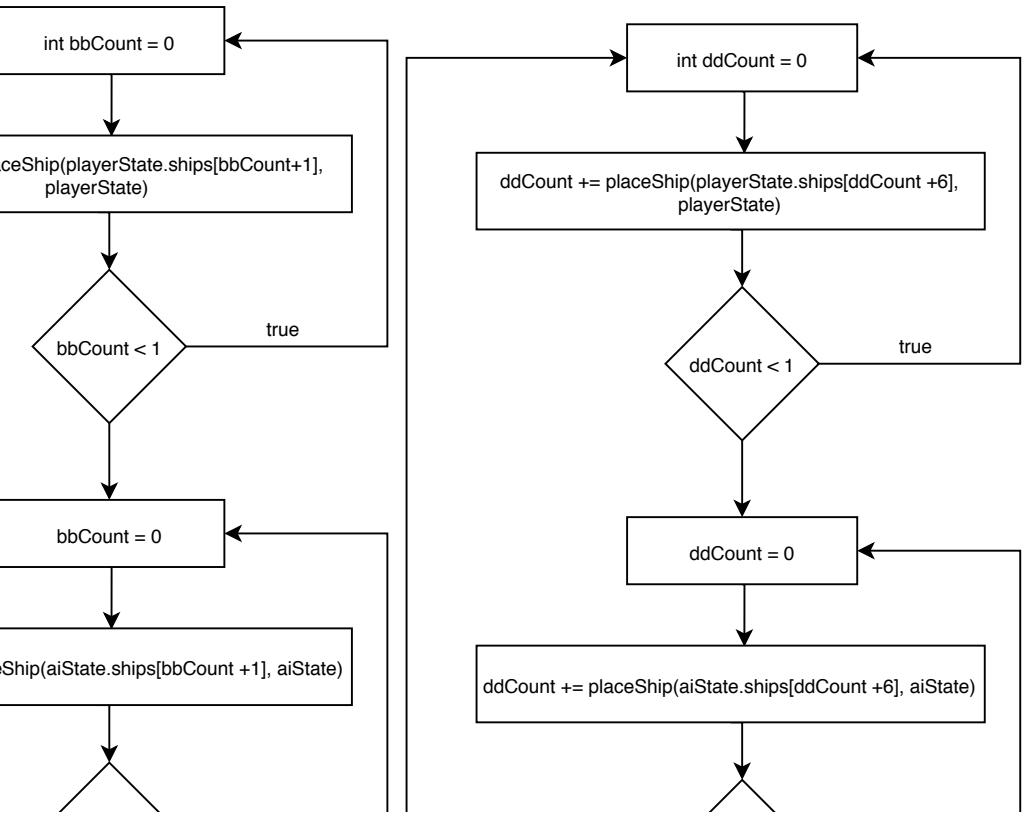

checkPlace(PlayerState, Int startX, startY, endX, endY)
max(0,startX)
max(0,startY)
int y = 0
y < (endY - startY)
false
true
int x = 0
y++
x < (endX - startX)
false
int trueX = startX + x
int trueY = startY + y

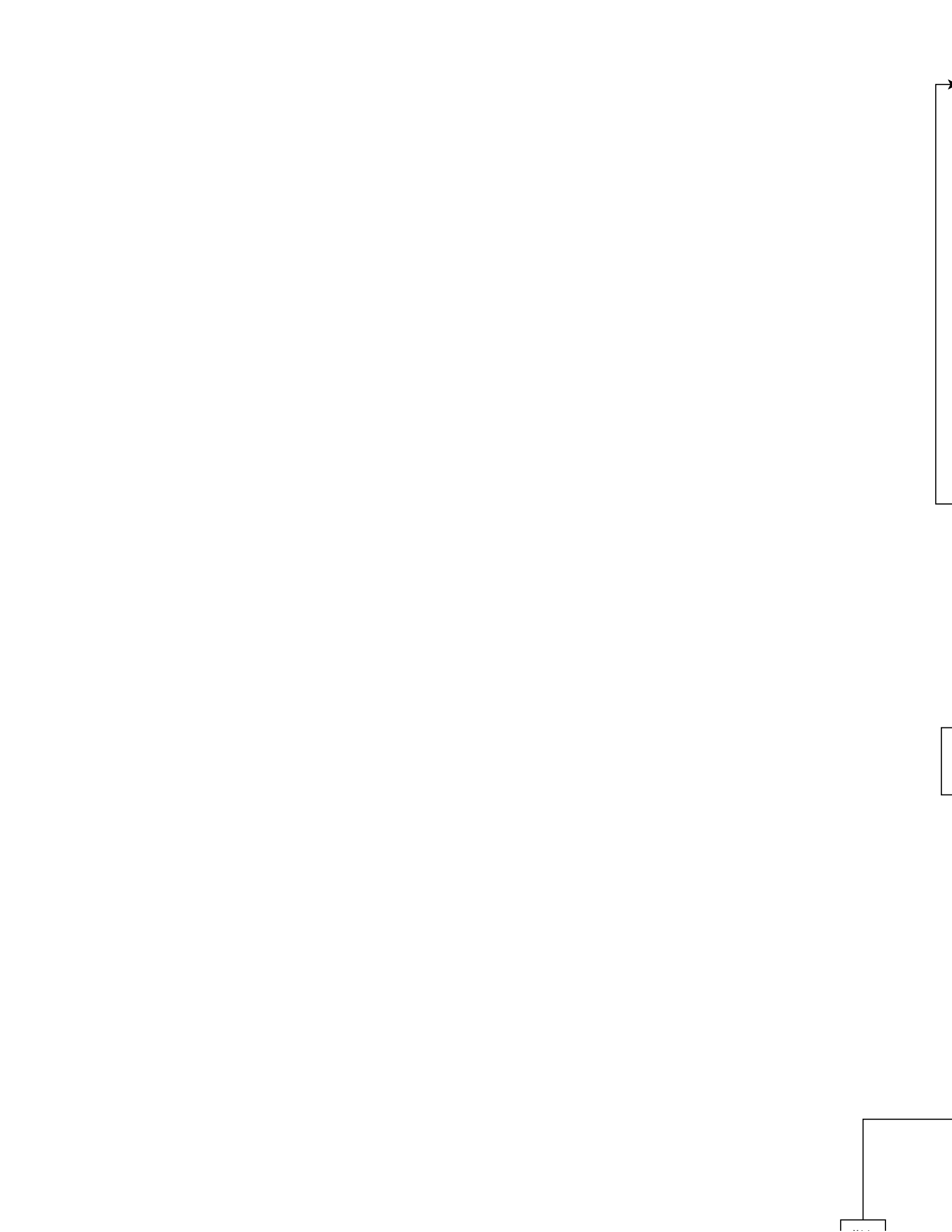

placeShip
int startX= random(0-10)
int randomY = random(0-10)
int orientation = random(0-10)
int endX = startX + ship.size
int endX = startY
orientation < 6
false
endY = startY + ship.size
endX = startX
bool isValid = checkPlace()
isValid
false
return 0
true

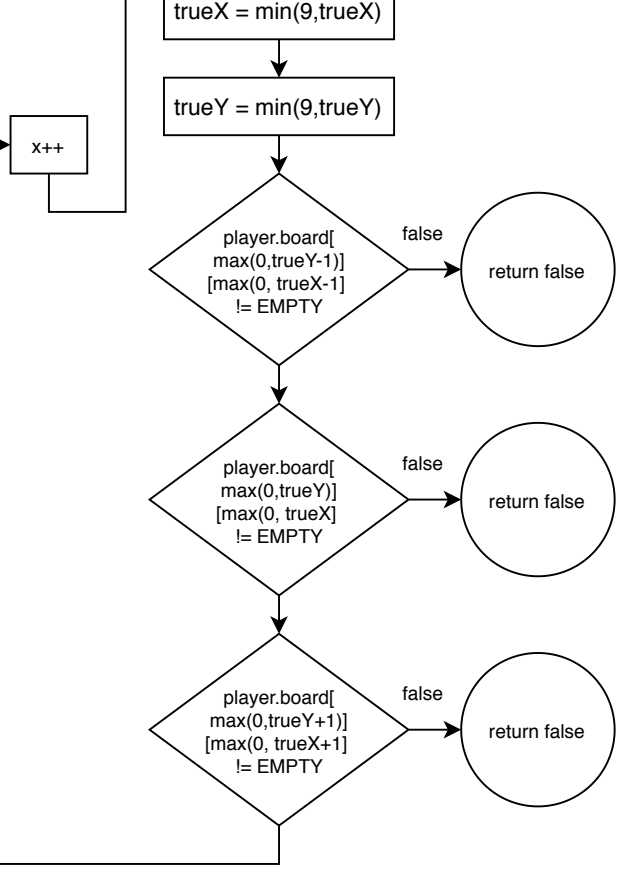

- Name=Battleship
- Type=BATTLESHIP
-Size=3
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
- Name=Destroyer
- Type=DESTROYER
-Size=2
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++
- Name=Cruiser
- Type=CRUISER
-Size=1
-Hits = 0
-id = id
-isDead = false
Store Ship struct in player ship array
Store Ship struct in array
id++

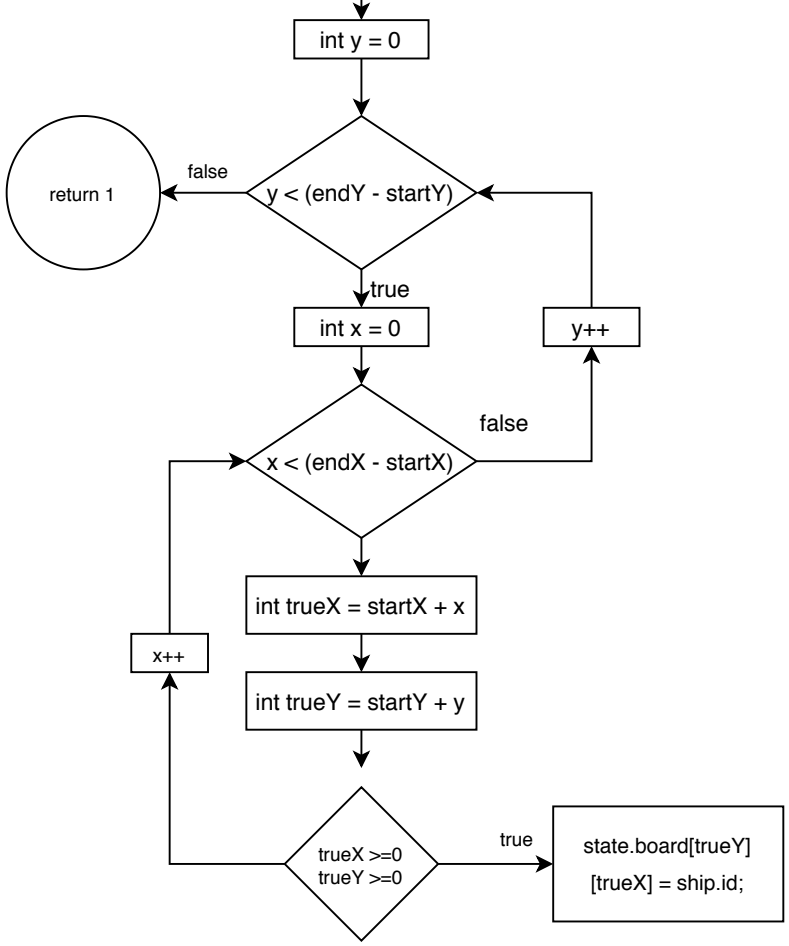

getChoice(out choiceX, choiceY)
Ask for Input X
cin -> choiceX
choiceX < 1 || choiceX > 10
true
Ask for Input Y
cin -> choiceY
choiceY < 1 || choiceY > 10
true

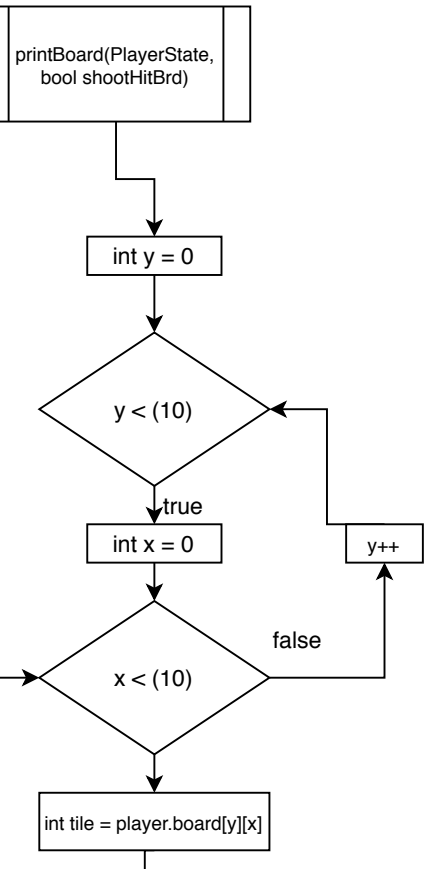

genBoard(GameState)
int y = 0
y < (10)
true
int x = 0
y++
x < (10)
false
player.board[y][x] = EMPTY
ai.board[y][x] = EMPTY
player.hitBrd[y][x] = EMPTY
ai.hitBrd[y][x] = EMPTY
x++
bbCount += pla
bbCount += place

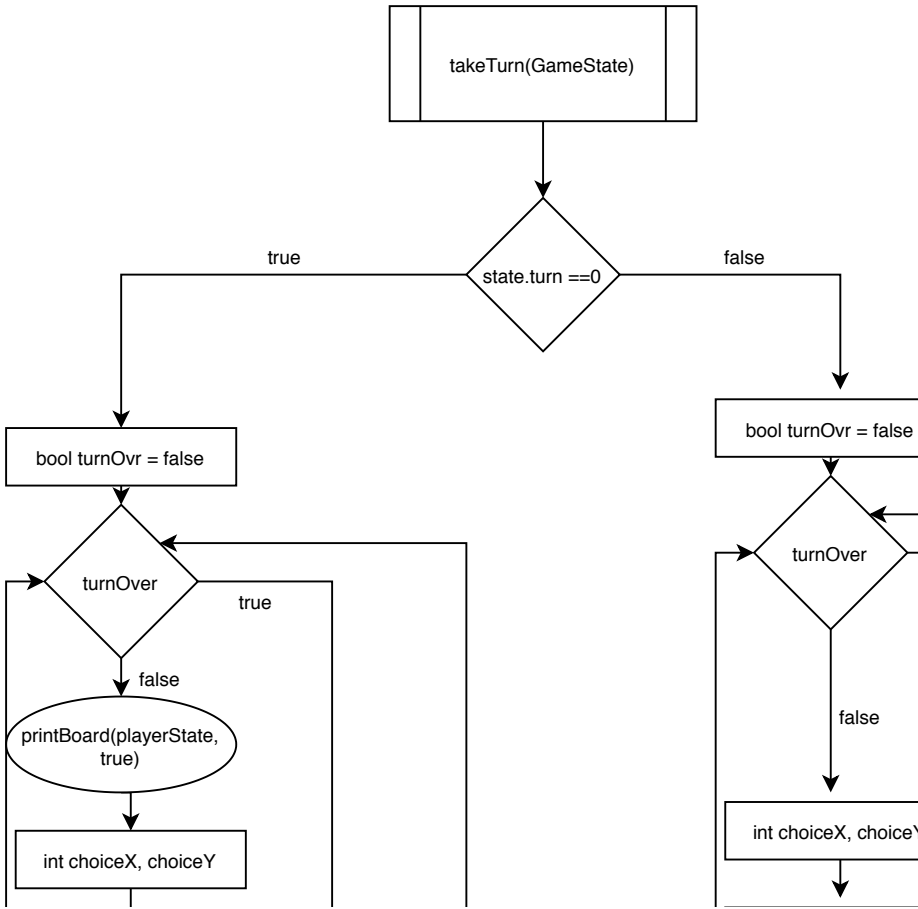

int bbCount = 0
ceShip(playerState.ships[bbCount+1],
playerState)
bbCount < 1
true
bbCount = 0
Ship(aiState.ships[bbCount +1], aiState)
int ddCount = 0
ddCount += placeShip(playerState.ships[ddCount +6],
playerState)
ddCount < 1
true
ddCount = 0
ddCount += placeShip(aiState.ships[ddCount +6], aiState)

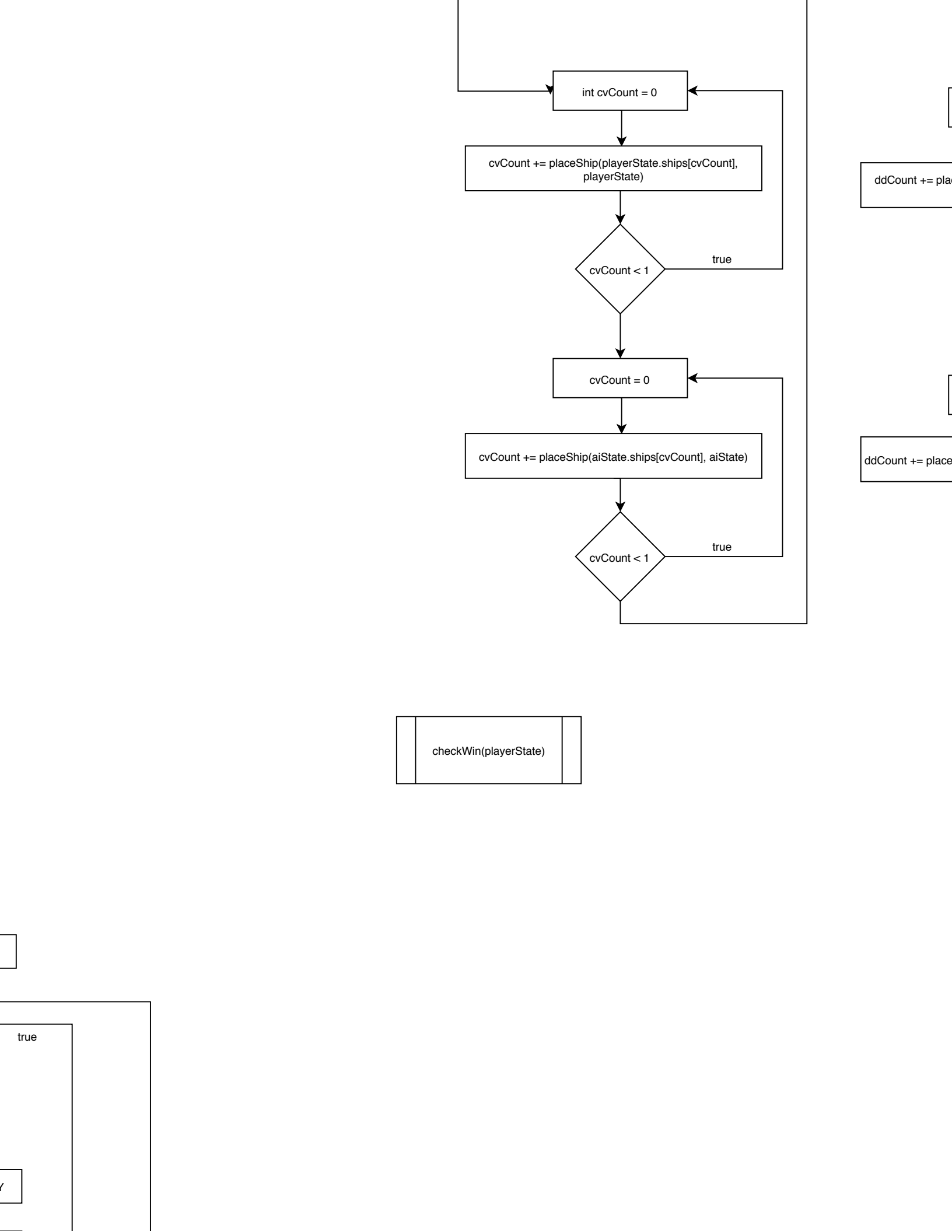



trueX = min(9,trueX)
trueY = min(9,trueY)
x++
player.board[
max(0,trueY-1)]
[max(0, trueX-1]
!= EMPTY
false
return false
player.board[
max(0,trueY)]
[max(0, trueX]
!= EMPTY
false
return false
player.board[
max(0,trueY+1)]
[max(0, trueX+1]
!= EMPTY
false
return false

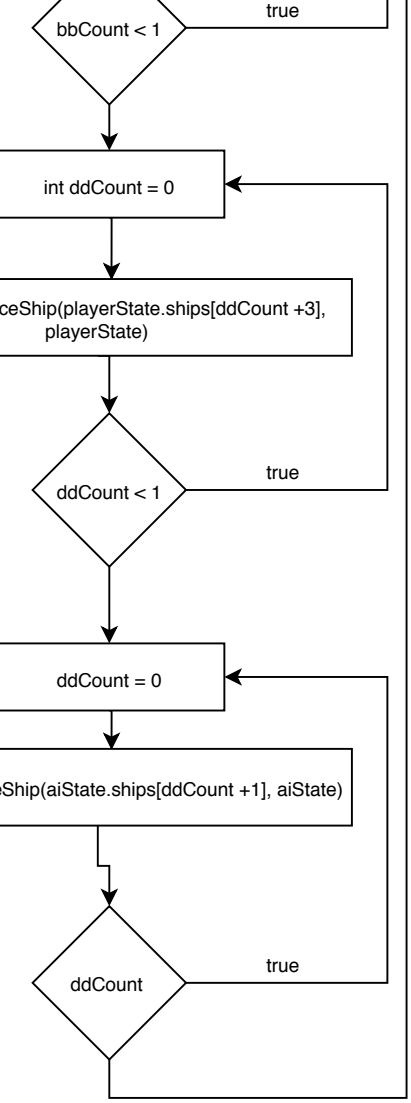

int y = 0
return 1
false
y < (endY - startY)
true
int x = 0
y++
false
x < (endX - startX)
int trueX = startX + x
x++
int trueY = startY + y
trueX >=0
trueY >=0
true
state.board[trueY]
[trueX] = ship.id;


printBoard(PlayerState,
bool shootHitBrd)
int y = 0
y < (10)
true
int x = 0
y++
false
x < (10)
int tile = player.board[y][x]

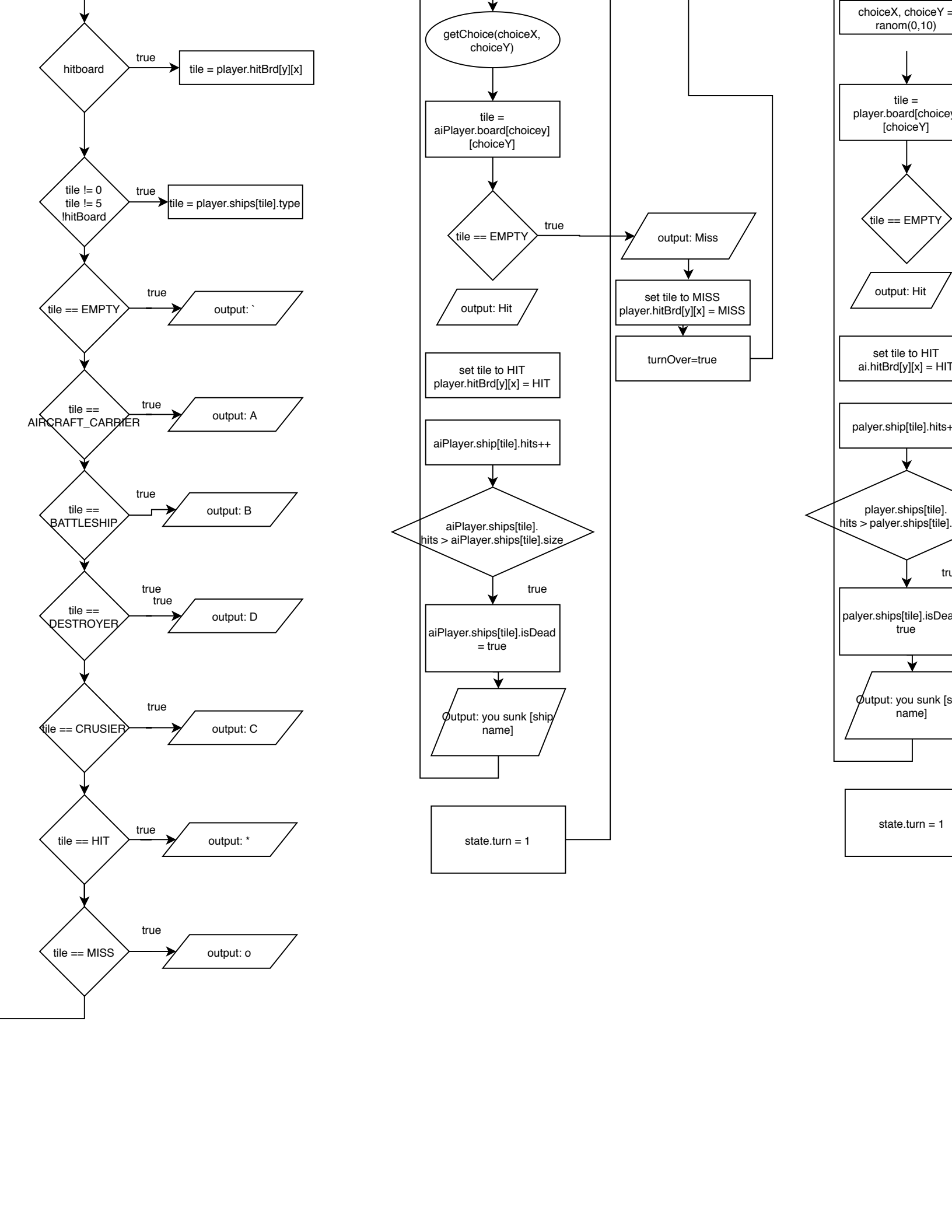

takeTurn(GameState)
state.turn ==0
true
false
bool turnOvr = false
turnOver
true
false
printBoard(playerState,
true)
int choiceX, choiceY
bool turnOvr = false
turnOver
false
int choiceX, choiceY

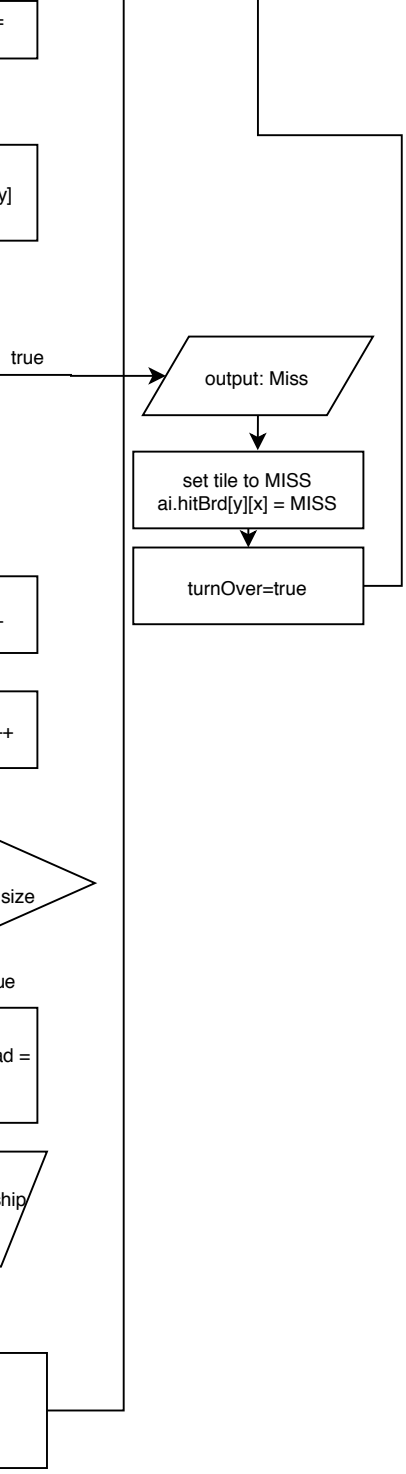

int cvCount = 0
cvCount += placeShip(playerState.ships[cvCount], playerState)
cvCount < 1
true
cvCount = 0
cvCount += placeShip(aiState.ships[cvCount], aiState)
cvCount < 1
true
checkWin(playerState)
ddCount += pla
ddCount += place
true

ddCount < 1
true


bbCount < 1
true
int ddCount = 0
ceShip(playerState.ships[ddCount +3],
playerState)
ddCount < 1
true
ddCount = 0
Ship(aiState.ships[ddCount +1], aiState)
ddCount
true

x++


hitboard
true
tile = player.hitBrd[y][x]
tile != 0
tile != 5
!hitBoard
true
tile = player.ships[tile].type
tile == EMPTY
true
output: `
tile ==
AIRCRAFT_CARRIER
true
output: A
tile ==
BATTLESHIP
true
output: B
tile ==
DESTROYER
true
true
output: D
tile == CRUSIER
true
output: C
tile == HIT
true
output: *
tile == MISS
true
output: o
getChoice(choiceX,
choiceY)
tile =
aiPlayer.board[choicey]
[choiceY]
tile == EMPTY
true
output: Miss
output: Hit
set tile to MISS
player.hitBrd[y][x] = MISS
turnOver=true
set tile to HIT
player.hitBrd[y][x] = HIT
aiPlayer.ship[tile].hits++
aiPlayer.ships[tile].
hits > aiPlayer.ships[tile].size
true
aiPlayer.ships[tile].isDead
= true
Output: you sunk [ship
name]
state.turn = 1
choiceX, choiceY =
ranom(0,10)
tile =
player.board[choicey]
[choiceY]
tile == EMPTY
output: Hit
set tile to HIT
ai.hitBrd[y][x] = HIT
palyer.ship[tile].hits+
player.ships[tile].
hits > palyer.ships[tile].
palyer.ships[tile].isDea
true
Output: you sunk [s
name]
state.turn = 1


true
output: Miss
set tile to MISS
ai.hitBrd[y][x] = MISS
turnOver=true