Project 1

<Battleship Game>

CSC-5 40651
Name: John Olmos
Date: 2/1/18

Introduction

Battleship is a two player guessing game. The game is played on two square grids on which players place a number of ships. Players take turns picking elements within the grid to try and “hit” the opposing player’s ships until all elements have been hit and their fleet is destroyed.

Game Rules

A typical game starts out with two 10x10 grids per player using numbers and letters to help identify elements within the grid. One grid is used to arrange the player’s ships and record their opponents picks while the second grid is used to record the player’s own picks.

Before the game starts each player arranges a ships in secret either vertically or horizontally within the grid. Each ship occupies a number of elements in the grid depending on its type making sure not to overlap with any other ship.

After each player has placed their ships the game starts in a series of rounds. During a round each player picks and element within the opponent’s grid in an attempt to hit one of their ships. Their opponent then announces if the pick was a “hit” or a “miss” on one of their ships and the result is recorded on a grid. If all of the elements of a ship have been hit it is considered sunk. The game ends once all ships have been hit and sunk.

Development Summary

V1

The first version of the game simply focused on getting the program to work. This meant scaling the game down to a single 1x10 row and limiting each side to one ship. The first obstacle was creating a way to allow the user to place a ship and then correctly updating a grid depending on their input for each round. Once this was done the process was mirrored and adjusted to allow the program to act as an opponent. Most time was spent on correctly updating the board and creating some basic form of logic for the computer’s turns.

V2

With the program essentially functional, v2 was focused entirely on incorporating more of the missing elements of the checklist into the program. The most notable changes in this version were the addition of new libraries, new outputs, and incorporating external files.

V3

The third iteration of the program made no significant additions or removals but instead focused on making the code more uniform and reorganizing individual snippets of code within the program to match the turn sequence and output.

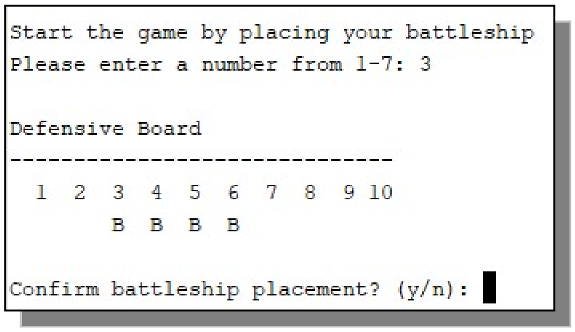
V4

The last version of the program focused on including the remaining items on the checklist and improving the comments to help better explain the process of the program.

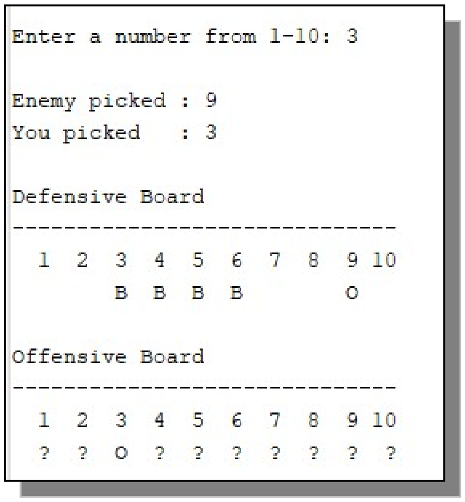
Example Inputs/Outputs

The first prompt in the game asks users to place and confirm their ship placement along the row by entering a number [1,7].

On the image to the right the program has output the player's board after they entered the number 3 for their ship placement. Here the letter B is used to represent the ship.



```
Start the game by placing your battleship
Please enter a number from 1-7: 3

Defensive Board
------------------------------
  1  2  3  4  5  6  7  8  9 10
        B  B  B  B

Confirm battleship placement? (y/n):
```

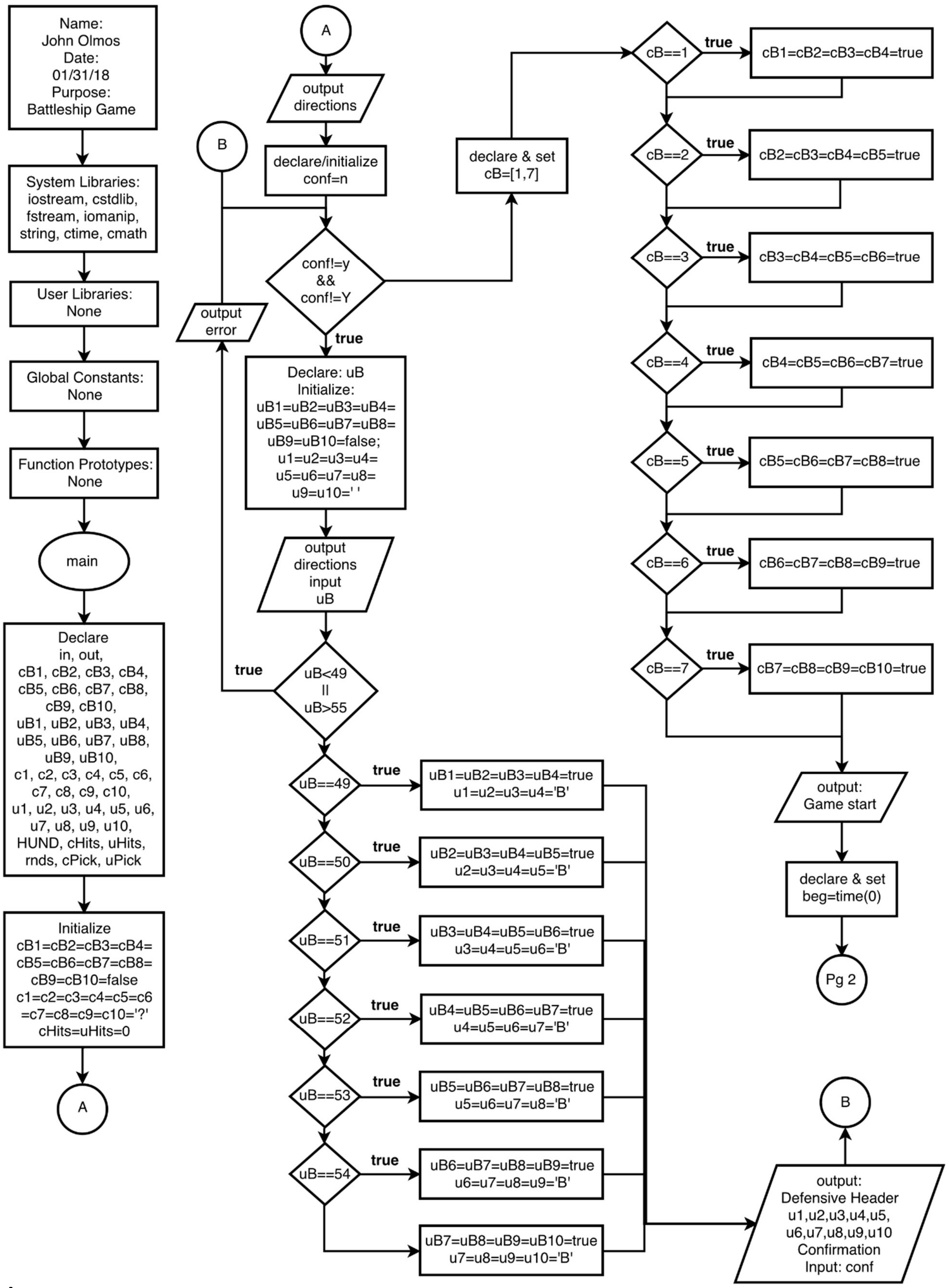


```
Enter a number from 1-10: 3

Enemy picked : 9
You picked   : 3

Defensive Board
------------------------------
  1  2  3  4  5  6  7  8  9 10
        B  B  B  B        O

Offensive Board
------------------------------
  1  2  3  4  5  6  7  8  9 10
  ?  ?  O  ?  ?  ?  ?  ?  ?  ?
```

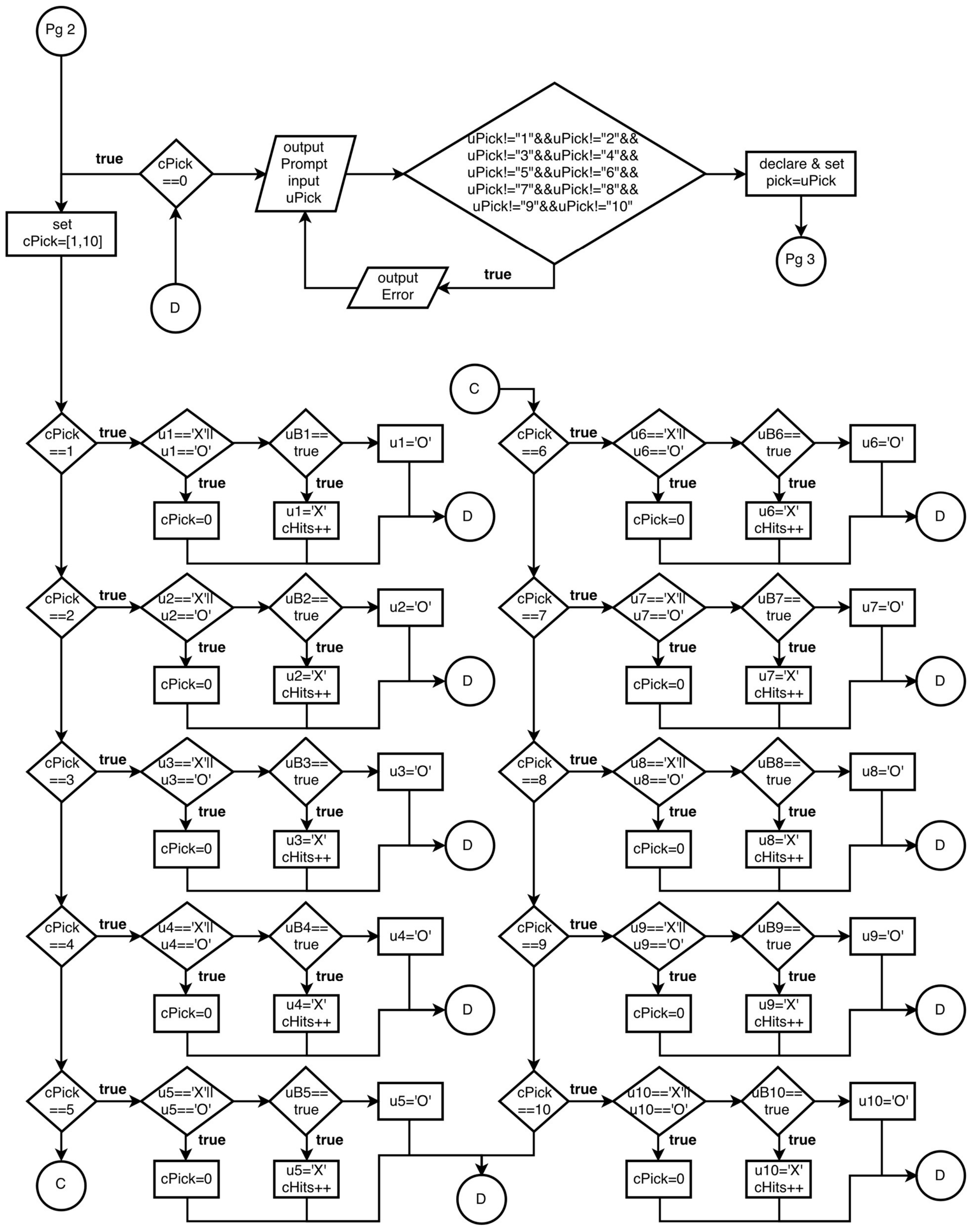
Later on the game asks users to enter a number [1,10] to try and hit the opponent's ship. Once entered, the program outputs both the player and opponent choices along with the defensive and offensive boards.

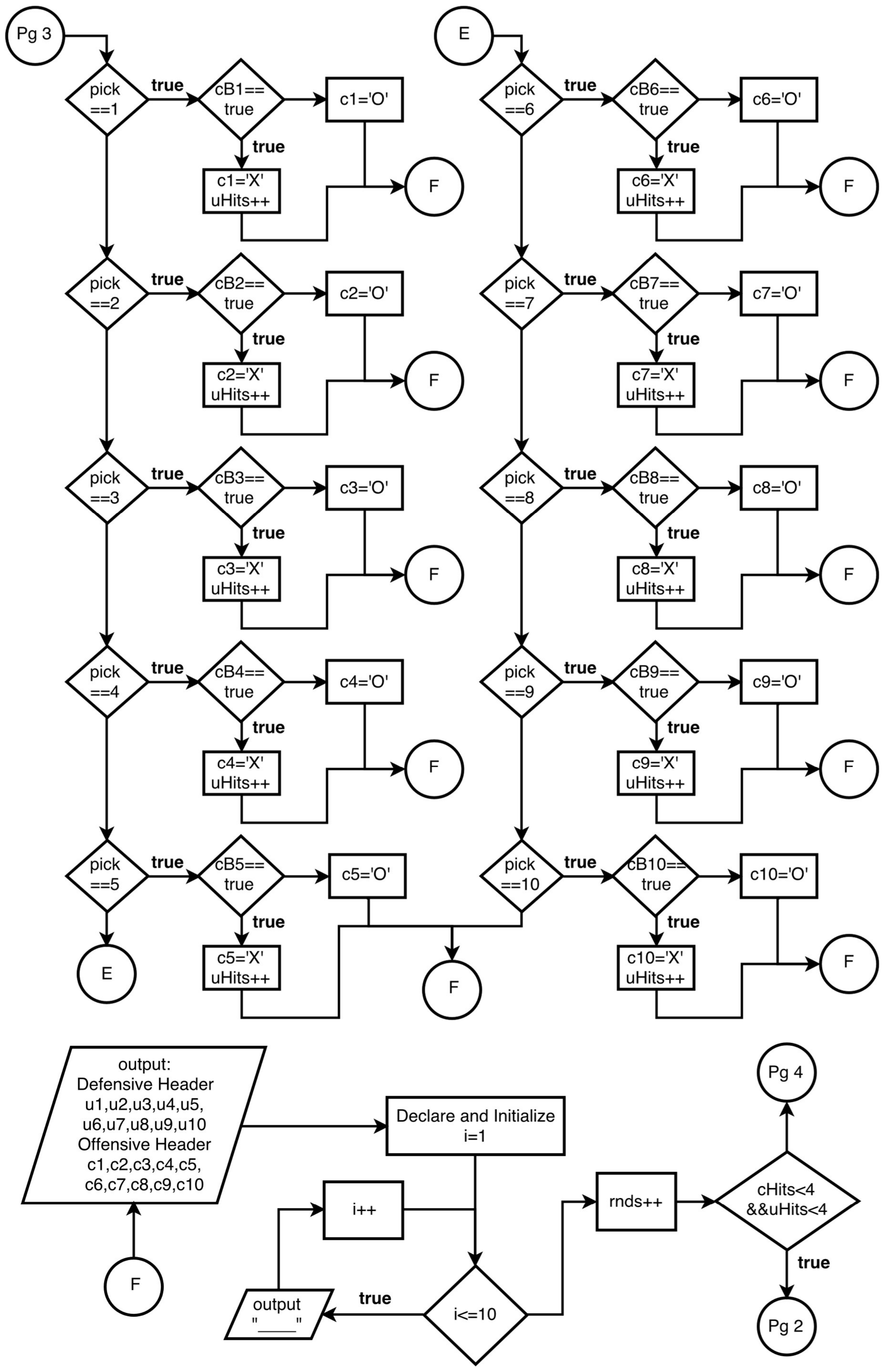
If a pick manages to hit a ship, then the board outputs an X. However, if the pick misses the program marks the board with an O as seen on the image to the left.

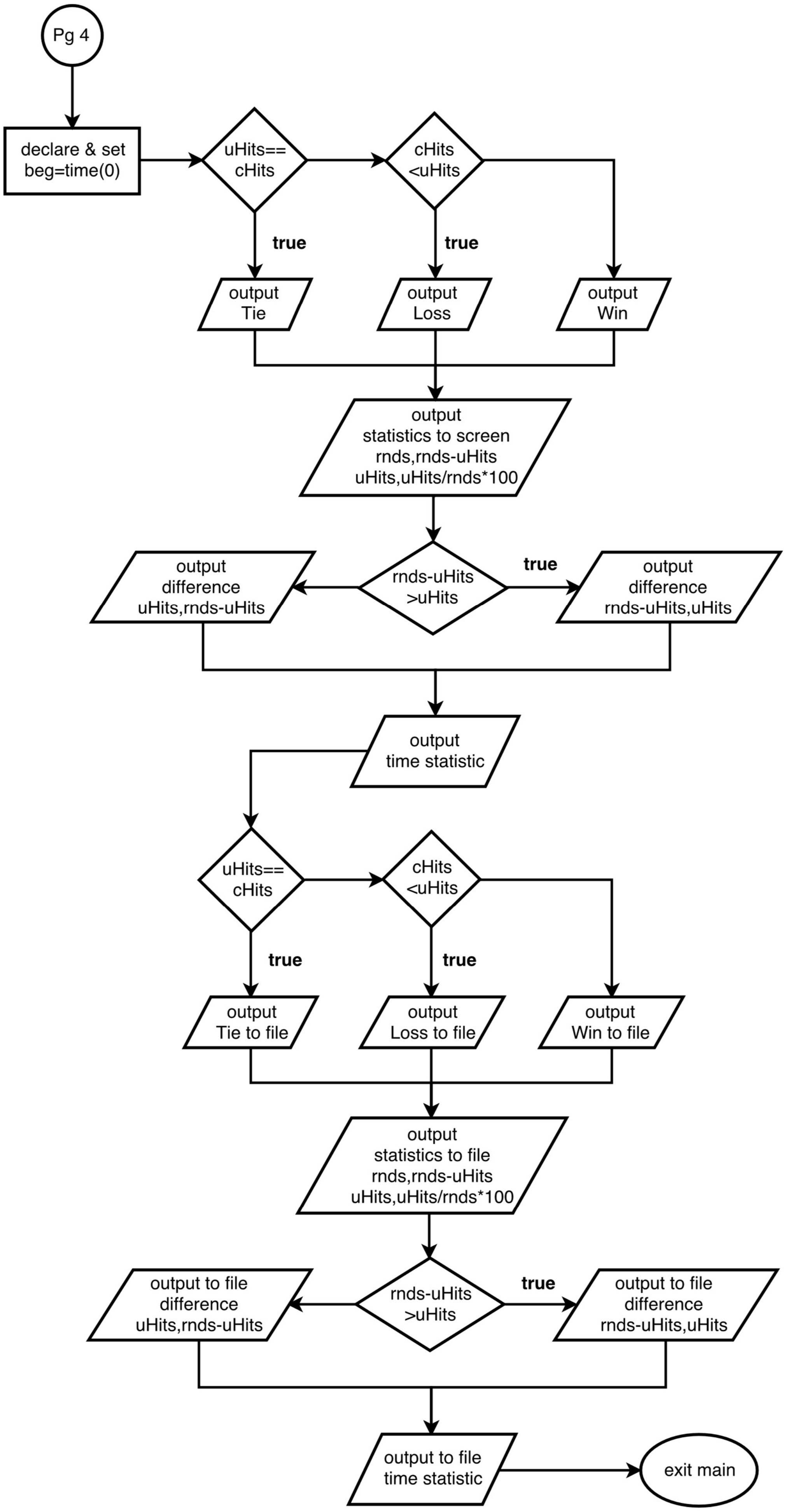
Flowchart



Name: John Olmos Date: 01/31/18 Purpose: Battleship Game
System Libraries: iostream, cstdlib, fstream, iomanip, string, ctime, cmath
User Libraries: None
Global Constants: None
Function Prototypes: None
main
Declare in, out, cB1, cB2, cB3, cB4, cB5, cB6, cB7, cB8, cB9, cB10, uB1, uB2, uB3, uB4, uB5, uB6, uB7, uB8, uB9, uB10, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, u1, u2, u3, u4, u5, u6, u7, u8, u9, u10, HUND, cHits, uHits, rnds, cPick, uPick
Initialize cB1=cB2=cB3=cB4= cB5=cB6=cB7=cB8= cB9=cB10=false c1=c2=c3=c4=c5=c6 =c7=c8=c9=c10='?' cHits=uHits=0
A
A
output directions
declare/initialize conf=n
B
conf!=y && conf!=Y
true
output error
Declare: uB Initialize: uB1=uB2=uB3=uB4= uB5=uB6=uB7=uB8= uB9=uB10=false; u1=u2=u3=u4= u5=u6=u7=u8= u9=u10=' '
output directions input uB
uB<49 || uB>55
true
uB==49
true
uB1=uB2=uB3=uB4=true u1=u2=u3=u4='B'
uB==50
true
uB2=uB3=uB4=uB5=true u2=u3=u4=u5='B'
uB==51
true
uB3=uB4=uB5=uB6=true u3=u4=u5=u6='B'
uB==52
true
uB4=uB5=uB6=uB7=true u4=u5=u6=u7='B'
uB==53
true
uB5=uB6=uB7=uB8=true u5=u6=u7=u8='B'
uB==54
true
uB6=uB7=uB8=uB9=true u6=u7=u8=u9='B'
uB7=uB8=uB9=uB10=true u7=u8=u9=u10='B'
declare & set cB=[1,7]
cB==1
true
cB1=cB2=cB3=cB4=true
cB==2
true
cB2=cB3=cB4=cB5=true
cB==3
true
cB3=cB4=cB5=cB6=true
cB==4
true
cB4=cB5=cB6=cB7=true
cB==5
true
cB5=cB6=cB7=cB8=true
cB==6
true
cB6=cB7=cB8=cB9=true
cB==7
true
cB7=cB8=cB9=cB10=true
output: Game start
declare & set beg=time(0)
Pg 2
B
output: Defensive Header u1,u2,u3,u4,u5, u6,u7,u8,u9,u10 Confirmation Input: conf


Pg 2
true
cPick ==0
output Prompt input uPick
uPick!="1"&&uPick!="2"&& uPick!="3"&&uPick!="4"&& uPick!="5"&&uPick!="6"&& uPick!="7"&&uPick!="8"&& uPick!="9"&&uPick!="10"
declare & set pick=uPick
Pg 3
true
output Error
set cPick=[1,10]
D
C
cPick ==1
true
u1=='X'll u1=='O'
uB1== true
u1='O'
true
true
cPick=0
u1='X' cHits++
D
cPick ==2
true
u2=='X'll u2=='O'
uB2== true
u2='O'
true
true
cPick=0
u2='X' cHits++
D
cPick ==3
true
u3=='X'll u3=='O'
uB3== true
u3='O'
true
true
cPick=0
u3='X' cHits++
D
cPick ==4
true
u4=='X'll u4=='O'
uB4== true
u4='O'
true
true
cPick=0
u4='X' cHits++
D
cPick ==5
true
u5=='X'll u5=='O'
uB5== true
u5='O'
true
true
C
cPick=0
u5='X' cHits++
D
cPick ==6
true
u6=='X'll u6=='O'
uB6== true
u6='O'
true
true
cPick=0
u6='X' cHits++
D
cPick ==7
true
u7=='X'll u7=='O'
uB7== true
u7='O'
true
true
cPick=0
u7='X' cHits++
D
cPick ==8
true
u8=='X'll u8=='O'
uB8== true
u8='O'
true
true
cPick=0
u8='X' cHits++
D
cPick ==9
true
u9=='X'll u9=='O'
uB9== true
u9='O'
true
true
cPick=0
u9='X' cHits++
D
cPick ==10
true
u10=='X'll u10=='O'
uB10== true
u10='O'
true
true
cPick=0
u10='X' cHits++
D


Pg 3
pick ==1
true
cB1== true
c1='O'
true
c1='X' uHits++
F
pick ==2
cB2== true
c2='O'
c2='X' uHits++
pick ==3
cB3== true
c3='O'
c3='X' uHits++
pick ==4
cB4== true
c4='O'
c4='X' uHits++
pick ==5
cB5== true
c5='O'
c5='X' uHits++
E
E
pick ==6
cB6== true
c6='O'
c6='X' uHits++
pick ==7
cB7== true
c7='O'
c7='X' uHits++
pick ==8
cB8== true
c8='O'
c8='X' uHits++
pick ==9
cB9== true
c9='O'
c9='X' uHits++
pick ==10
cB10== true
c10='O'
c10='X' uHits++
output: Defensive Header u1,u2,u3,u4,u5, u6,u7,u8,u9,u10 Offensive Header c1,c2,c3,c4,c5, c6,c7,c8,c9,c10
Declare and Initialize i=1
i++
i<=10
output "____"
rnds++
cHits<4 &&uHits<4
Pg 4
Pg 2


Pg 4
declare & set beg=time(0)
uHits== cHits
cHits <uHits
true
true
output Tie
output Loss
output Win
output statistics to screen rnds,rnds-uHits uHits,uHits/rnds*100
output difference uHits,rnds-uHits
rnds-uHits >uHits
true
output difference rnds-uHits,uHits
output time statistic
uHits== cHits
cHits <uHits
true
true
output Tie to file
output Loss to file
output Win to file
output statistics to file rnds,rnds-uHits uHits,uHits/rnds*100
output to file difference uHits,rnds-uHits
rnds-uHits >uHits
true
output to file difference rnds-uHits,uHits
output to file time statistic
exit main

Pseudo-Code

```
/*
 * File:   main.cpp
 * Author: John Olmos
 * Created on Jan 30, 2018, 12:06 AM
 * Purpose: Simple Battleship
 */

//System Libraries Here
//Input - Output Library
//Srand to set the seed
//File I/O
//Format the output
//Strings
//Time library
//Math functions

//User Libraries Here

//Global Constants
//Such as PI, Vc, -> Math/Science values
//as well as conversions from system of units to another
//Percentage Conversion

//Function Prototypes Here

//Program Execution Begins Here
    //Set the random number seed

    //Instantiate and Open files

    //Declare Variables
    //I/O Files
    //Computer battleship location
    //User battleship location
    //Offensive Board
    //Defensive board
    //Constant for 100
    //User hit counter
    //round counter
    //Computer pick
    //User pick

    //Input or initialize values Here
    //Open I/O files
    //Last value in file becomes rounds
    //Clear offensive ship
    //Mask offensive board
```

```
//Reset hit counters

//Output instructions
//Set confirmation to 'n'
//Loop until ship is set and confirmed
  //Clear defensive ship & board
  //Output directions
  //Input ship location & validate
  //Set user battleship location
      //Output Defensive Board
      //Confirm

//Generate and battleship location & place

//Output Game start

//Play the game by -> Looping until user or computer has 4 hits
//Start time
    //Computer turn -> Loop if element has been picked before
        //Generate Computer pick
      //Computer pick result
        //For hits: change element to X and increment computer hit
        //For misses: change element to O
  //User turn
      //Input user pick
      //validate input
        //User pick result
            //For hits: change element to X and increment computer hit
            //For misses: change element to O
        //Output both picks
        //Output Defensive Header & Board
        //Output Offensive Header & Board
        //Output divider and then increment round
//Stop time

//Output all the statistics to the screen

//Output all the statistics to a file

//Close files and Exit stage right!
```

Check-off Sheet

Chap	Section	Topic	Where Line #''s	Pts	Notes
2	2	cout	56-58, 66, 67, 72, 86-96, 116-121, 158, 164, 184-209, 217-225, 227, 228, 230, 231, 233		
	3	libraries	9-15	8	iostream, iomanip, cmath, cstdlib, fstream, string, ctime
	4	variables/literals	31-43		No variables in global area, failed project!
	5	Identifiers	31-43		
	6	Integers	28, 39-42, 124, 169, 213	3	
	7	Characters	37, 38, 60, 62	3	
	8	Strings	43, 46	3	
	9	Floats No Doubles	225, 244	3	Using doubles will fail the project, floats OK!
	10	Bools	33-36	4	
	11	Sizeof *****			
	12	Variables 7 characters or less	31-43		All variables <= 7 characters
	13	Scope ***** No Global Variables			
	14	Arithmetic operators	222, 225, 226, 228, 231, 233, 241, 244, 245, 247, 250, 252		
	15	Comments 20%+	1-256	5	Model as pseudo code
	16	Named Constants			All Local, only Conversions/Physics/Math in Global area
	17	Programming Style ***** Emulate			Emulate style in book/in class repository
3	1	cin	68, 97, 159		
	2	Math Expression	222, 225, 226, 228, 231, 233, 241, 244, 245, 247, 250, 252		
	3	Mixing data types ****			
	4	Overflow/Underflow ****			

	5	Type Casting	28, 225, 244	4	
	6	Multiple assignment *****			
	7	Formatting output	91-93, 194-196, 204-206, 224, 228, 231, 243, 247, 250	4	
	8	Strings	43, 46	3	
	9	Math Library	228, 231, 247, 250	4	All libraries included have to be used
	10	Hand tracing ******			
4	1	Relational Operators			
	2	if	107-113	4	Independent if
	4	If-else	71-98, 162-211, 226-232, 245-251	4	
	5	Nesting	77-97	4	
	6	If-else-if	77-83	4	
	7	Flags *****			
	8	Logical operators	61, 71, 133-151, 162, 163, 212	4	
	11	Validating user input	70-73, 161-165	4	
	13	Conditional Operator	133-152, 171-180, 217, 218, 236, 237	4	
	14	Switch	132-153, 170-181	4	
5	1	Increment/Decrement	134, 136, 138, 140, 142, 144, 146, 148, 150, 152, 171-180, 209, 210	4	
	2	While	61	4	
	5	Do-while	125, 127	4	
	6	For loop	209	4	
	11	Files input/output both	48-50, 236-252	8	
	12	No breaks in loops ******			Failed Project if included
****** **Not required to show**			Total	100	