BBM101 COURSE
ASSİGNMENT 4TH REPORT

BATTLE OF SHIPS

Name: Nurşah

Surname: Satılmış

Number: 2210765010

Grade: One

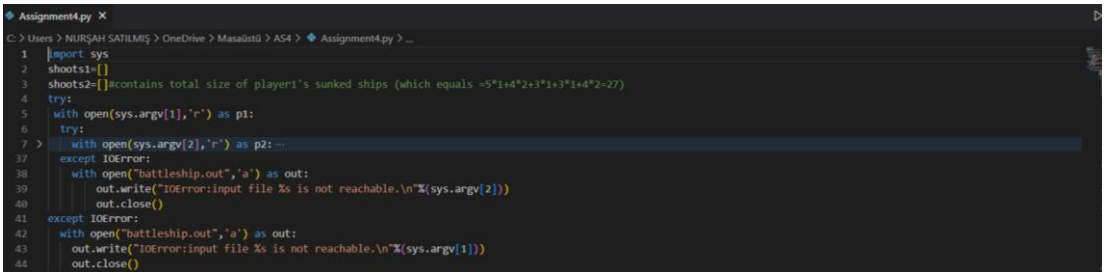
Adress: Artificial Intelligence Engineering Department , Hacettepe University, Beytepe, Ankara

Due Date: 04.01.2023

ANALYSİS

In this assignment, we handle a strategy game problem, Battle of Ships Game. Basically, our purpose is the reading data sets and analyzing them after that compare different data sets. Our data sets are two files which contain player's ship positions and another two file which include player's movements. After analyzing and comparing data sets, we show every game round on a table.

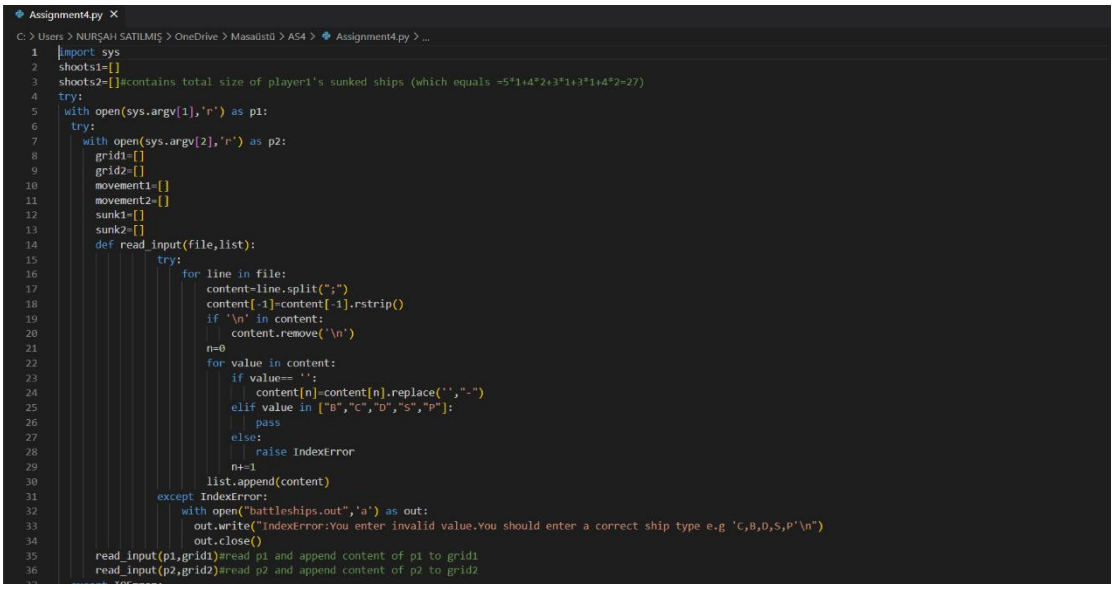
DESIGN



```
import sys
shoots1=[]
shoots2=[]#contains total size of player1's sunked ships (which equals =5*1+4*2+3*1+3*1+4*2=27)
try:
 with open(sys.argv[1],'r') as p1:
  try:
    with open(sys.argv[2],'r') as p2:
  except IOError:
    with open("battleship.out",'a') as out:
        out.write("IOError:input file %s is not reachable.\n"%(sys.argv[2]))
        out.close()
except IOError:
  with open("battleship.out",'a') as out:
    out.write("IOError:input file %s is not reachable.\n"%(sys.argv[1]))
    out.close()
```

First I created two global list for appending the parts of ships that were shot by players.Shoots1 contain player1's ship parts that were shot by player2.

I opened first command line argument (expect it is "player1.txt") as p1 and opened second command line argument("player2.txt") as p2.And like be given above I except some IO errors.



```
import sys
shoots1=[]
shoots2=[]#contains total size of player1's sunked ships (which equals =5*1+4*2+3*1+3*1+4*2=27)
try:
 with open(sys.argv[1],'r') as p1:
  try:
    with open(sys.argv[2],'r') as p2:
      grid1=[]
      grid2=[]
      movement1=[]
      movement2=[]
      sunk1=[]
      sunk2=[]
      def read_input(file,list):
              try:
                  for line in file:
                      content=line.split(";")
                      content[-1]=content[-1].rstrip()
                      if '\n' in content:
                          content.remove('\n')
                      n=0
                      for value in content:
                          if value== '':
                              content[n]=content[n].replace('','-')
                          elif value in ["B","C","D","S","P"]:
                              pass
                          else:
                              raise IndexError
                          n+=1
                      list.append(content)
              except IndexError:
                  with open("battleships.out",'a') as out:
                    out.write("IndexError:You enter invalid value.You should enter a correct ship type e.g 'C,B,D,S,P'\n")
                    out.close()
      read_input(p1,grid1)#read p1 and append content of p1 to grid1
      read_input(p2,grid2)#read p2 and append content of p2 to grid2
```

After opening input files, I define a function for reading files.Read_input() function reads files line by line. Then, split it from " ; ". For removing "\n" at the end of lines, I used rstrip() function.To be able to reading easily these positions files later, I change the empty item with "-" and If there is a invalid value, function raise IndexError. After checking items , function append the items to a list.

So, I called the function for p1, and p2.After that, grid1 contain player1's ship position, grid2 contains player2's ship position.

```
Assignment4.py
C: > Users > NURŞAH SATILMIŞ > OneDrive > Masaüstü > AS4 > Assignment4.py > ...
43        out.write("IOError:input file %s is not reachable.\n"%(sys.argv[1]))
44        out.close()
45    try:
46      with open(sys.argv[3],'r') as lunge1:
47        try:
48          with open(sys.argv[4],'r') as lunge2:...
314       except IOError:
315         with open("battleship.out",'a') as out:
316           out.write("IOError:input file %s is not reachable.\n"%(sys.argv[4]))
317           out.close()
318   except IOError:
319     with open("battleship.out",'a') as out:
320           out.write("IOError:input file %s is not reachable.\n"%(sys.argv[3]))
321           out.close()
322
```

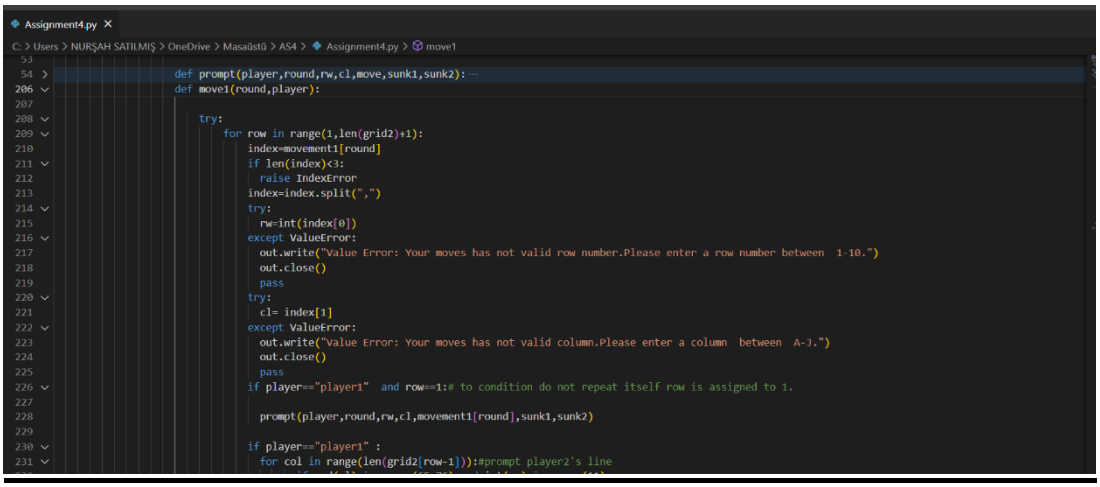
I opened the 4th command line argument (“player1.in”) as lunge1 and opened 5th argument (“player2.in”) as lunge2.

Before investigate the lines below the try, take a look exception blocks .

```
45  try:
46    with open(sys.argv[3],'r') as lunge1:
47      try:
48        with open(sys.argv[4],'r') as lunge2:
49                  movement1=lunge1.readline()
50                  movement1=movement1.split(";")
51                  movement2=lunge2.readline()
52                  movement2=movement2.split(";")
53
54                  def prompt(player,round,rw,cl,move,sunk1,sunk2):...
206                 def move1(round,player):...
298                 with open("battleship.out",'a') as out:
299                     out.write("Battle of Ships Game\n\n")
300                     print("Battle of Ships Game\n")
301                     round=0
302                 while len(shoots1)<27:#contains total size of ships (which equals =5*1+4*2+3*1+3*1+4*2=27)
303                         move1(round,"player1")
304                         move1(round,"player2")
305                         if len(shoots1)==27:
306                           with open("battleship.out",'a') as out:
307                             out.write("Player2 wins!")
308                             break
309                         elif len(shoots2)==27:
310                           with open("battleship.out",'a') as out:
311                             out.write("Player1 wins!")
312                             break
313                         round+=1
```

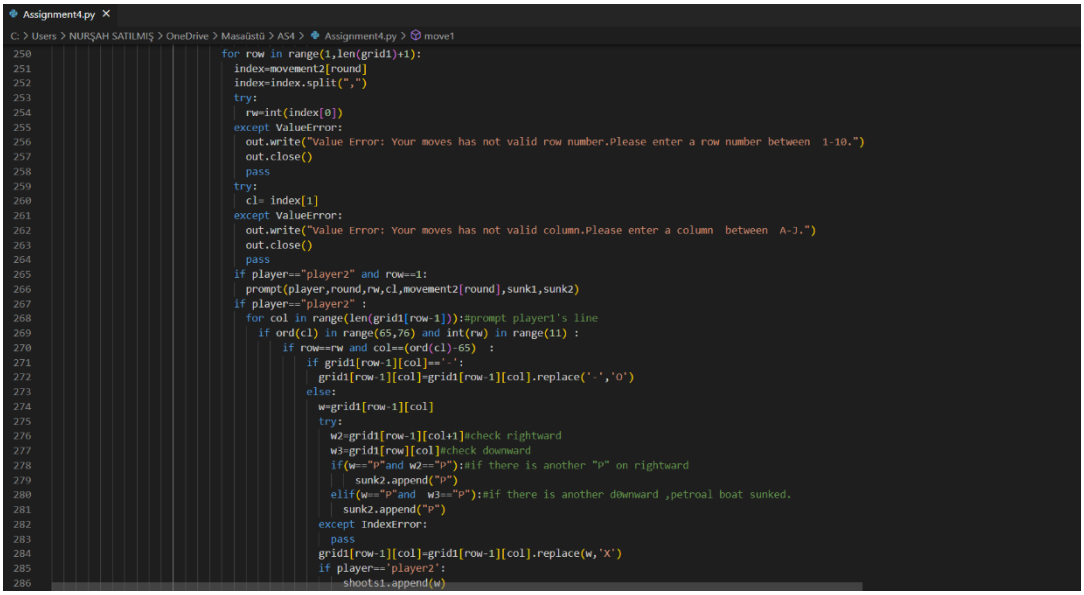
I create two empty list after opening p2 which is movement1 and movement2. By readline() functions(I used readline() instead of readlines() because input file include 1 line) I added the contents to movement1 and movement2. So, movement1 includes player1’s moves, movement2 includes player2’s moves.

I create two function: move1() is for evaluate the moves and prompt() printing the data which taken from move1, it means I called prompt from inside of the move1() function. By using while loop I called move1() function for player1 and player2. Loop run until len(shoots2)==27 , I used this limitation because total size of ships is 27 and last turn belongs second player. Inside of the loop I created condition for determining the winner.



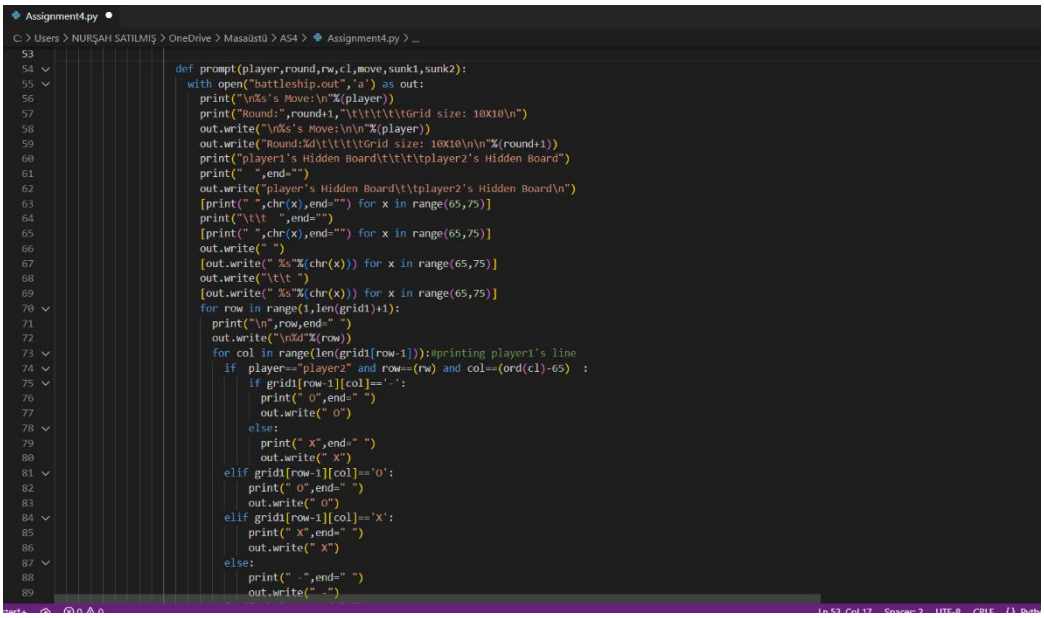
In move1() funciton I created an outer loop for printing the rows.Inside the outer loop I get the moves and check it.Length of the each moves should 3 or 4 (e.g. with comma length of 1,A is 3) , if length is less than 3 it means there is missing value ,so in this condition ValueError raises.I get the moves with “index=movement1[round]” and after split the index,the firt element of index represent the row number and the second is the column number.For detecting invalid row and column value, different ValueErrors is prompts.If player1 turns prompt funciton is called with values (player,round,rw,cl,movement1(round),sunk1,sunk2)

There is a inner loop , it is created for columns. İnitially we check whether the move’s row and column is equal to loops values.(In this condition I used the integer values of characters by using ASCII values.) If they are matched, inner conditions check whether is there any ship part in this position. If there is a ship part, Function change the items with “X” , if there is not any ship part then it changes the item with “0”.In this situation with other condition I checked the leftward and downward for detecting petrol boat.(to be honest my code failed for counting petrol boat.I couldn’t applied this techniques well.)Lastly, we append the fired parts of ships to shoots2 (because the ships belongs the second player).



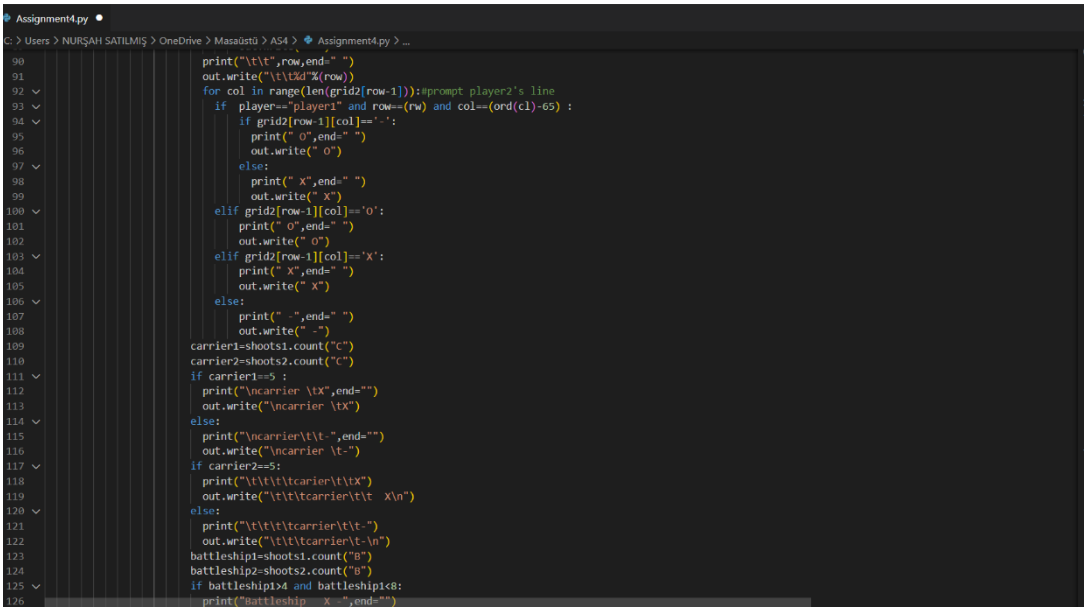
```
Assignment4.py
C: > Users > NURŞAH SATILMIŞ > OneDrive > Masaüstü > AS4 > Assignment4.py > move1
250                 for row in range(1,len(grid1)+1):
251                   index=movement2[round]
252                   index=index.split(",")
253                   try:
254                     rw=int(index[0])
255                   except ValueError:
256                     out.write("Value Error: Your moves has not valid row number.Please enter a row number between  1-10.")
257                     out.close()
258                     pass
259                   try:
260                     cl= index[1]
261                   except ValueError:
262                     out.write("Value Error: Your moves has not valid column.Please enter a column  between  A-J.")
263                     out.close()
264                     pass
265                   if player=="player2" and row==1:
266                     prompt(player,round,rw,cl,movement2[round],sunk1,sunk2)
267                   if player=="player2" :
268                     for col in range(len(grid1[row-1])):#prompt player1's line
269                       if ord(cl) in range(65,76) and int(rw) in range(11) :
270                           if row==rw and col==(ord(cl)-65)  :
271                               if grid1[row-1][col]=='-':
272                                 grid1[row-1][col]=grid1[row-1][col].replace('-','O')
273                               else:
274                                 w=grid1[row-1][col]
275                                 try:
276                                   w2=grid1[row-1][col+1]#check rightward
277                                   w3=grid1[row][col]#check downward
278                                   if(w=="P"and w2=="P"):#if there is another "P" on rightward
279                                       sunk2.append("P")
280                                   elif(w=="P"and  w3=="P"):#if there is another d0wnward ,petroal boat sunked.
281                                     sunk2.append("P")
282                                 except IndexError:
283                                   pass
284                                 grid1[row-1][col]=grid1[row-1][col].replace(w,'X')
285                                 if player=='player2':
286                                     shoots1.append(w)
```

İn the below part function does the same process for player2 .



```
Assignment4.py
C: > Users > NURŞAH SATILMIŞ > OneDrive > Masaüstü > AS4 > Assignment4.py > ...
53
54          def prompt(player,round,rw,cl,move,sunk1,sunk2):
55            with open("battleship.out",'a') as out:
56              print("\n%s's Move:\n"%(player))
57              print("Round:",round+1,"\t\t\t\t\tGrid size: 10X10\n")
58              out.write("\n%s's Move:\n\n"%(player))
59              out.write("Round:%d\t\t\t\tGrid size: 10X10\n\n\n"%(round+1))
60              print("player1's Hidden Board\t\t\t\tplayer2's Hidden Board")
61              print("  ",end="")
62              out.write("player's Hidden Board\t\tplayer2's Hidden Board\n")
63              [print(" ",chr(x),end="") for x in range(65,75)]
64              print("\t\t  ",end="")
65              [print(" ",chr(x),end="") for x in range(65,75)]
66              out.write(" ")
67              [out.write(" %s"%(chr(x))) for x in range(65,75)]
68              out.write("\t\t ")
69              [out.write(" %s"%(chr(x))) for x in range(65,75)]
70              for row in range(1,len(grid1)+1):
71                print("\n",row,end=" ")
72                out.write("\n%d"%(row))
73                for col in range(len(grid1[row-1])):#printing player1's line
74                  if  player=="player2" and row==(rw) and col==(ord(cl)-65)  :
75                      if grid1[row-1][col]=='-':
76                        print(" O",end=" ")
77                        out.write(" O")
78                      else:
79                        print(" X",end=" ")
80                        out.write(" X")
81                  elif grid1[row-1][col]=='O':
82                      print(" O",end=" ")
83                      out.write(" O")
84                  elif grid1[row-1][col]=='X':
85                      print(" X",end=" ")
86                      out.write(" X")
87                  else:
88                      print(" -",end=" ")
89                      out.write(" -")
```

In prompt function, first we prompt the initial sentences then, header row which covers the column characters. After that, the outer loop scan the lines, and the inner loop scan the columns. At the first condition on inner loop, it checks if the moves matching the loops values (row, col). If they are matched checked the items if item is “-“ prompt “O” ,else (ship letter) prompt “X”. If they are not matched, then it checked “X” and “O” in the lists because it represents the previous moves.



```
Assignment4.py
C: > Users > NURŞAH SATILMIŞ > OneDrive > Masaüstü > AS4 > Assignment4.py > ...
                print("\t\t",row,end=" ")
                out.write("\t\t%d"%(row))
                for col in range(len(grid2[row-1])):#prompt player2's line
                  if  player=="player1" and row==(rw) and col==(ord(cl)-65) :
                      if grid2[row-1][col]=='-':
                        print(" O",end=" ")
                        out.write(" O")
                      else:
                        print(" X",end=" ")
                        out.write(" X")
                  elif grid2[row-1][col]=='O':
                      print(" O",end=" ")
                      out.write(" O")
                  elif grid2[row-1][col]=='X':
                      print(" X",end=" ")
                      out.write(" X")
                  else:
                      print(" -",end=" ")
                      out.write(" -")
              carrier1=shoots1.count("C")
              carrier2=shoots2.count("C")
              if carrier1==5 :
                print("\ncarrier \tX",end="")
                out.write("\ncarrier \tX")
              else:
                print("\ncarrier\t\t-",end="")
                out.write("\ncarrier \t-")
              if carrier2==5:
                print("\t\t\t\tcarier\t\tX")
                out.write("\t\t\tcarrier\t\t  X\n")
              else:
                print("\t\t\t\tcarrier\t\t-")
                out.write("\t\t\tcarrier\t-\n")
              battleship1=shoots1.count("B")
              battleship2=shoots2.count("B")
              if battleship1>4 and battleship1<8:
                print("Battleship  X ",end="")
```

After prompt tabs

Function does the same prompting process for player2. As I said before, the inner loop is for columns, after the complete the columns It passed second row and it repeated until the last row.

After prompting table, we count the fired ships in the shoots lists. Then, prompting the proper output.

Pseudocode of the program:

1. Take the files name from command line
2. Read player1 and player2' s ship position file
3. Transfer the contents to lists (grid1 and grid2)
4. Read players moves files
5. Transfer the contents of files to lists (movement1 and movement2)
6. By scanning the ship positions lists, prompt the contents of lists as a table
7. Check the shoots lists and count the fired ship parts
8. (if first we change the lists then there is a mistake on the board)
9. Compare the items of ship positions list and other player's moves lists step by step
10. If they are matched, change the item of ship positions lists with "X" and append the items to shoots list.
11. If they are not matched, change the item with "O"
12. If one of shoots list include all ships part (the size should be 27) then go to line 15
13. Return line 7
14. Stop

Programmer's Catalogue

the time spent I analyze	2 hours
the time I spent designing,	8 hours
the time I spent implementing	24 hours
the time I spent testing	4 hours
the time I spent reporting	4 hours

User Catalogue:

With using this program, you can play battle of ship. But program has some restrictions.

One of the limitations of this program is that the moves are not read simultaneously. If we look at the operation of the program, both players enter the moves first, then we read the entered moves and print the table. That is, the moves are not created simultaneously with the game.

The other limiting factor is that the game is played on a 10x10 table, which shortens the possibilities and playing time.