

SPRINT 2 SOFTWARE ENGINEERING

Pratham Shah

Andrey Shandybin

Previous Sprint that I received

World of bugs

[Start game](#)

After pressing start game this is what was displayed

Object not found!

The requested URL was not found on this server. The link on the [referring page](#) seems to be wrong or outdated. Please inform the author of [that page](#) about the error.

If you think this is a server error, please contact the [webmaster](#).

Error 404

clabsql.clamy.jacobs-university.de
Apache

As you can see the previous sprint did not have much progress in terms of UI or implementation in terms of world map file parsing nor bug brain assembler parsing the only thing that I could use from their folder was some their bug.js bugbrain.js code

The following is the progress that I made I improved the UI

Made the Welcome page better

Welcome to the Bug World!

Start!

After pressing start button you are taken to the settings page which was not there in the previous sprint

Bug World

Please upload a world map file:

Choose Files No file chosen

Please upload a bug assembler source code file #1:

Choose Files No file chosen

Please upload a bug assembler source code file #2:

Choose Files No file chosen

Please input the number of iterations:

Do you want to log the results of the session?

☐

Next

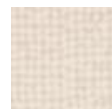
I have made progress on the world map file where I have implemented ability to upload files and parse them all on the client-side these are the instructions in the world map file

```
#      obstacle
.      empty cell (ie, no bug)
-      empty cell, black swarm nest
+      empty cell, red swarm nest
1..9   empty cell with number of food units
```

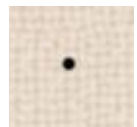
A # in one cell input in the world map file will give an obstacle output that looks like this in it's respective cell



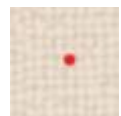
A . input in a cell will give you an empty cell that looks like this



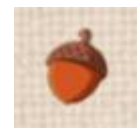
A - input in a cell will give you a black swarm nest which is indicated by a black dot that looks like this



A + input in a cell will give you a red swarm nest which is indicated by a red dot that looks like this



A 1...9 input in a cell which indicates food will give you a output that looks like this
I was not able to implement number of food units but a number input in a particular cell position will give you a food output like the image above in that position



First I will show an example what successful test case looks like suppose I have the following input world map file which accepts any format(.txt, .world etc) doesn't matter as long as the instructions pass the error checks which I have worked on

```
10
10
# # # # # # # # # #
# 9 9 . . . . 3 3 #
# 9 # . - - - - - #
# . # - - - - - - #
# . . 5 - - - - - #
# + + + + + 5 . . #
# + + + + + # . #
# + + + + + . . 9 #
# 3 3 . . . . 9 9 #
# # # # # # # # # #
```

This is an example of a successful world map file and I will input this file on the following page like this

Bug World

Please upload a world map file:

Choose Files

Please upload a bug assembler source code file #1 (for red bugs):

Choose Files

Please upload a bug assembler source code file #2 (for black bugs):

Choose Files

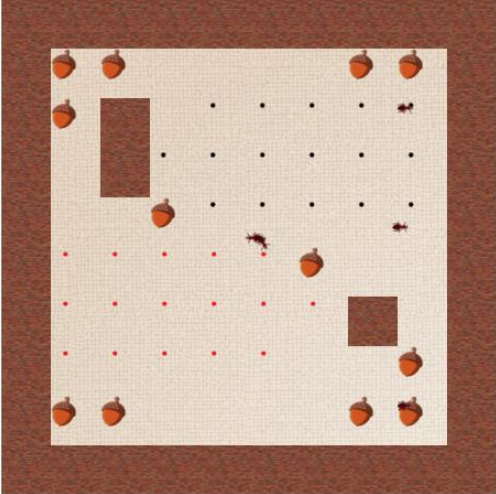
Please input the number of iterations (minimum 100):

Do you want to log the results of the session?

☒

Next

After pressing the next button we get the following output as it reads the instructions of the world map file all on the client side



Iterations: 550/1000

Amount of undetected food: 76

Red bugs remaining: 5

Red bugs killed: 0

Food brought home for red bugs: 0

Black bugs remaining: 0

Black bugs killed: 0

Food brought home for black bugs: 0

Logs will be here

Quit!

As you can see all the instructions in the world map file are implemented as mention After pressing quit button you are taken back to the welcome page

Now four error cases that I have implemented for world map file

Error case 1: incorrect dimensions

Example suppose I input the world map file below which clearly has incorrect dimension

```
10
12
# # # # # # # # # #
# 9 4 . . . . 3 3 #
# 9 # . . . . . #
# . # . . . . . #
# . . 5 . . . . #
# + + + + + 5 . . #
# + + + + + # . #
# + + + + + . # 9 #
# 3 3 . . . . 9 9 #
# # # # # # # # # #
```

After pressing next I will get the following error message

clabsql.clamv.jacobs-university.de says
Wrong Dimensions.

OK

Food brough home for red bugs:

Black bugs remaining:

Black bugs killed:

Food brough home for black bugs:

Logs will be here

Error test case 2: border is not closed

Example suppose I input the world map file below which clearly has incorrect border

```
10
10
# # # # # # # # # #
# 9 9 . . . 3 3 #
# 9 # . - - - - - #
# . # - - - - - #
# . . 5 - - - - - #
# + + + + + . . #
# + + + + + # . #
# + + + + + . # 9 #
# 3 3 . . . 9 9 .
# # # # # # # # # #
```

After pressing next I will get the following error message

clabsql.clamv.jacobs-university.de says

Borders not closed.

OK

Food brought home for red bugs:

Black bugs remaining:

Black bugs killed:

Food brought home for black bugs:

Logs will be here

Quit!

Error test case 3: Invalid character in map

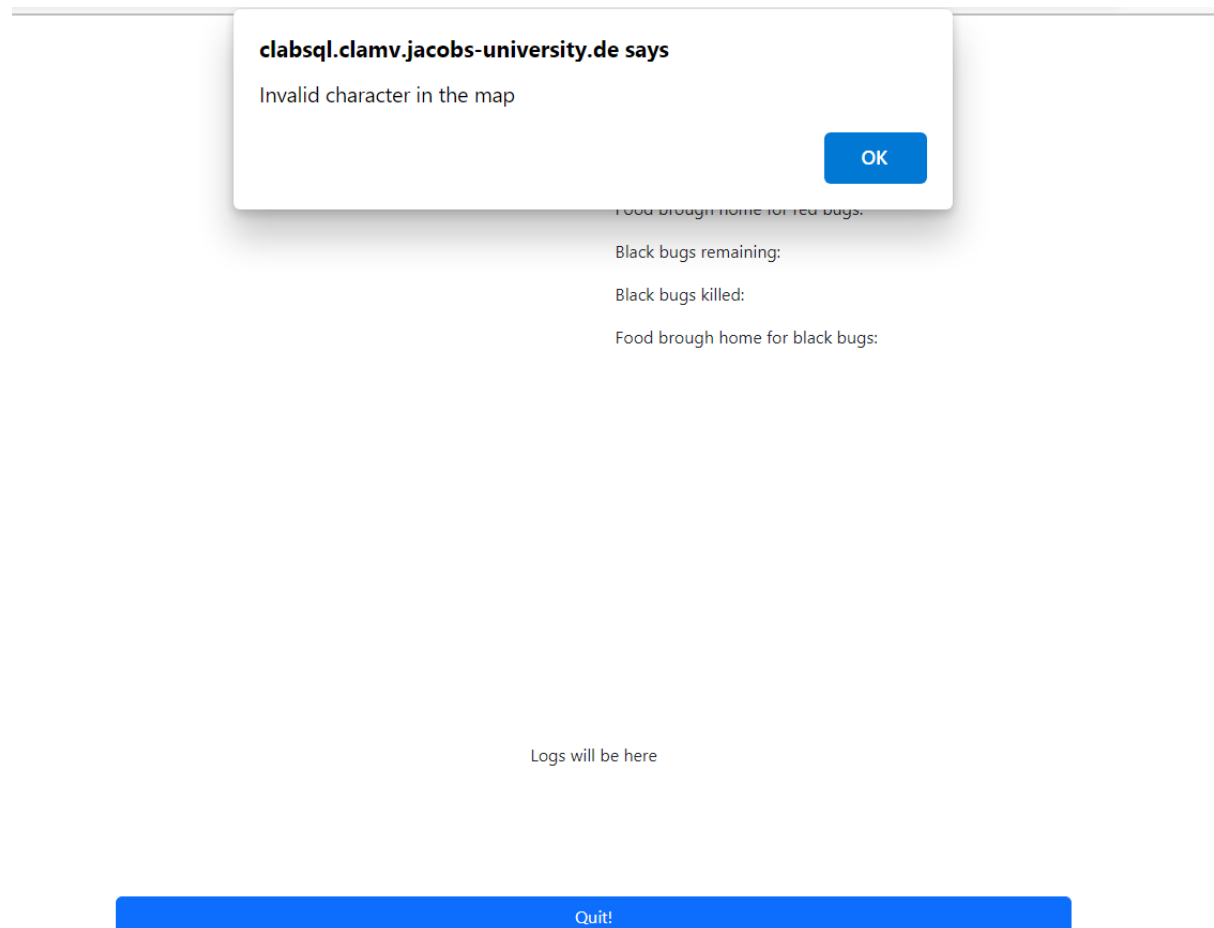
Example suppose I input the world map file below which clearly has an invalid character a

```

10
10
# # # # # # # # # #
# a 9 . . . . 3 3 #
# 9 # . - - - - - #
# . # - - - - - #
# . . 9 - - - - - #
# + + + + + 5 . . #
# + + + + + + # . #
# + + + + + . # 9 #
# 3 3 . . . . 0 9 #
# # # # # # # # # #

```

After pressing next I will get the following error message

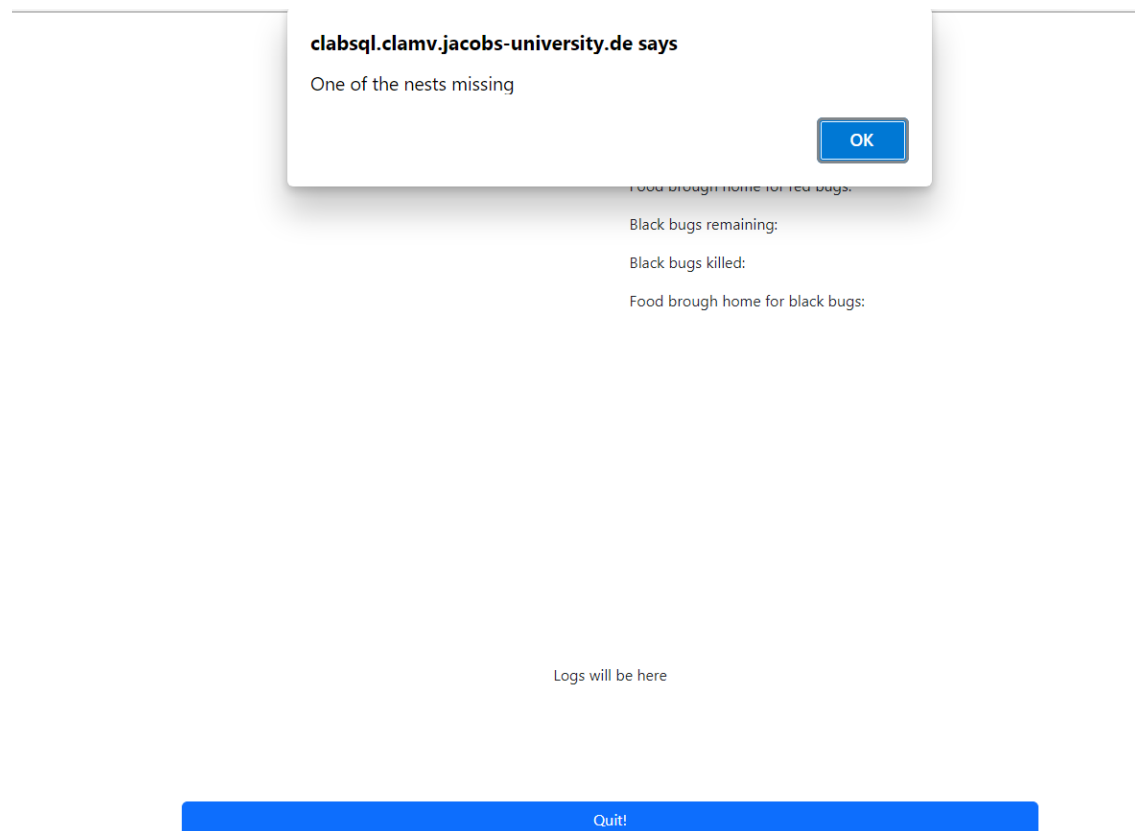


Error test case 4: one of the bug nests is missing

Example suppose I input the world map file below which clearly has a bug nest missing in this case – (black swarm nest)

```
10
10
# # # # # # # # # #
# 9 9 . . . . 3 3 #
# 9 # . . . . . . #
# . # . . . . . . #
# . . 5 . . . . . #
# + + + + + 5 . . #
# + + + + + + # . #
# + + + + + . # 9 #
# 3 3 . . . . 9 9 #
# # # # # # # # # #
```

After pressing next I will get the following error message



Any test case that does not defy the errors that I have worked on will be a successful test case and output a correct map file unless the buggy assembler files are files are wrong

I also made progress on the buggy files for the bug brain of red and black swarms where the assembler code is parsed implement all the functions of the bug brain as shown in the assembler code (move, sense ahead, turn, turn left etc)

suppose First I will show an example what successful test case looks like suppose I have the following input world map file and buggy assembler files for the red and the black bugs

The input windows will look as follows

Bug World

Please upload a world map file:

Choose Files
successworldtestcase1.txt

Please upload a bug assembler source code file #1 (for red bugs):

Choose Files
successred.buggy.txt

Please upload a bug assembler source code file #2 (for black bugs):

Choose Files
successblack.buggy.txt

Please input the number of iterations (minimum 100):
1000

Do you want to log the results of the session?
☒

Next

```

10
10
# # # # # # # #
# 9 9 . . . 3 3 #
# 9 # . - - - - #
# . # - - - - - #
# . . 5 - - - - - #
# + + + + + 5 . . #
# + + + + + + # . #
# + + + + + . . 9 #
# 3 3 . . . 9 9 #
# # # # # # # #

```

World map file

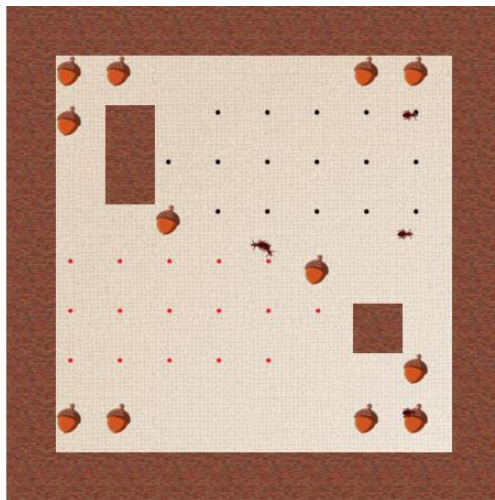
```

sense ahead 1 3 food ; [ 0]
move 2 0 ; [ 1]
pickup 8 0 ; [ 2]
flip 3 4 5 ; [ 3]
turn left 0 ; [ 4]
flip 2 6 7 ; [ 5]
turn right 0 ; [ 6]
move 0 3 ; [ 7]
sense ahead 9 11 home ; [ 8]
move 10 8 ; [ 9]
drop 0 ; [ 10]
flip 3 12 13 ; [ 11]
turn left 8 ; [ 12]
flip 2 14 15 ; [ 13]
turn right 8 ; [ 14]
move 8 11 ; [ 15]

```

Buggy assembler files for red and black bugs

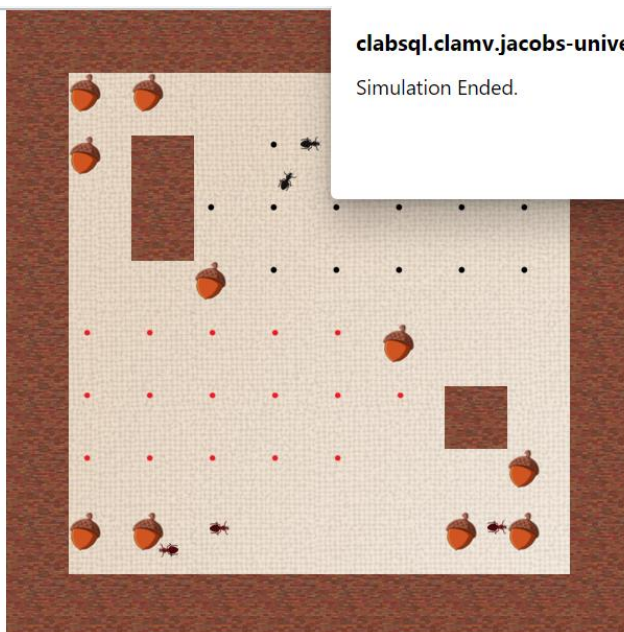
The output will be a simulation of the instructions in the buggy assembler code the logs will appear below the output screen if the option is selected



Iterations: 550/1000
Amount of undetected food: 76
Red bugs remaining: 5
Red bugs killed: 0
Food brought home for red bugs: 0
Black bugs remaining: 0
Black bugs killed: 0
Food brought home for black bugs: 0

Logs will be here

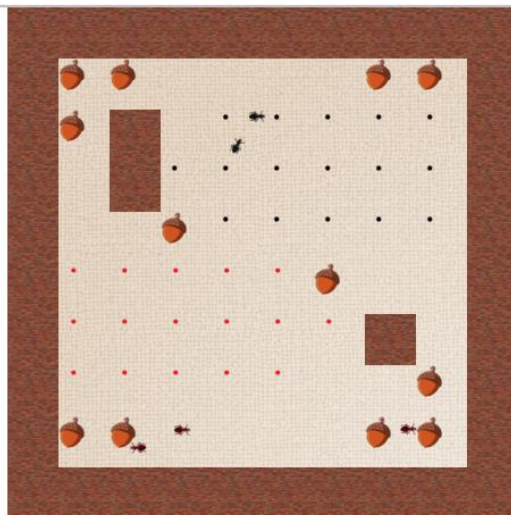
Quit!



Food brought home for red bugs: 0
Black bugs remaining: 2
Black bugs killed: 0
Food brought home for black bugs: 0

Logs will be here

Quit!



Iterations: 1000/1000
 Amount of undetected food: 76
 Red bugs remaining: 3
 Red bugs killed: 0
 Food brought home for red bugs: 0
 Black bugs remaining: 2
 Black bugs killed: 0
 Food brought home for black bugs: 0

Iterations: 1000/1000
 Amount of undetected food: 76
 Red bugs remaining: 3
 Red bugs killed: 0
 Food brought home for red bugs: 0
 Black bugs remaining: 2
 Black bugs killed: 0
 Food brought home for black bugs: 0

Quit!

Now four error cases that I have implemented for buggy assembler files

Error case 1: link to a non-existing line

Example suppose I input the following buggy error file which clearly has a link to a non-existing line (line 5)

```
sense ahead 1 3 food ; [ 0]
move 2 0 ; [ 1]
pickup 8 0 ; [ 2]
flip 3 4 5 ; [ 3]
turn left 0 ; [ 4]
flip 2 6 1223674 ; [ 5]
turn right 0 ; [ 6]
move 0 3 ; [ 7]
sense ahead 9 11 home ; [ 8]
move 10 8 ; [ 9]
drop 0 ; [10]
flip 3 12 13 ; [11]
turn left 8 ; [12]
flip 2 14 15 ; [13]
turn right 8 ; [14]
move 8 11 ; [15]
```

The input window looks like this

Bug World

Please upload a world map file:

Choose Files

Please upload a bug assembler source code file #1 (for red bugs):

Choose Files

Please upload a bug assembler source code file #2 (for black bugs):

Choose Files

Please input the number of iterations (minimum 100):

Do you want to log the results of the session?

☒

Next

Note: Doesn't matter where you input the buggy error in assembler source code for red bugs or black bugs the error will still show the following is the order in which error is detected first for the world map file then in assembler code for red bugs after that in assembler code for black bugs

We get the following error message after pressing next for buggy error case 1

clabsql.clamv.jacobs-university.de says
Link to a non existent line

OK

Food brought home for red bugs:

Black bugs remaining:

Black bugs killed:

Food brought home for black bugs:

Logs will be here

Quit!

Error case 2: invalid token

Example suppose I input the following buggy error file which clearly has invalid token (line 0)

```
sense unrecognizable_word 1 3 food
; [ 0]
move 2 0 ; [ 1]
pickup 8 0 ; [ 2]
flip 3 4 5 ; [ 3]
turn left 0 ; [ 4]
flip 2 6 7 ; [ 5]
turn right 0 ; [ 6]
move 0 3 ; [ 7]
sense ahead 9 11 home ; [ 8]
move 10 8 ; [ 9]
drop 0 ; [10]
flip 3 12 13 ; [11]
turn left 8 ; [12]
flip 2 14 15 ; [13]
turn right 8 ; [14]
move 8 11 ; [15]
```

We get the following error message after pressing next for buggy error case 2

clabsql.clamv.jacobs-university.de says

Invalid token: unrecognizable_word

OK

Food brought home for red bugs:

Black bugs remaining:

Black bugs killed:

Food brought home for black bugs:

Logs will be here

Quit!

Error case 3: incorrect arguments for function

Example suppose I input the following buggy error file which clearly has Incorrect arguments for function

```
sense ahead 1 3 food ; [ 0]
move 2 ; [ 1] absent value
pickup 8 0 ; [ 2]
flip 3 4 5 ; [ 3]
turn left 0 ; [ 4]
flip 2 6 7 ; [ 5]
turn right 0 ; [ 6]
move 0 3 ; [ 7]
sense ahead 9 11 home ; [ 8]
move 10 8 ; [ 9]
drop 0 ; [10]
flip 3 12 13 ; [11]
turn left 8 ; [12]
flip 2 14 15 ; [13]
turn right 8 ; [14]
move 8 11 ; [15]
```

We get the following error message after pressing next for buggy error case 3

clabsql.clamv.jacobs-university.de says

Incorrect arguments for function

OK

Food brought home for red bugs:

Black bugs remaining:

Black bugs killed:

Food brought home for black bugs:

Logs will be here

Quit!

Any test case that does not defy the errors that I have worked on will be a successful test case and output a working simulation file unless the world map file is invalid

Note: Doesn't matter where you input the buggy error in assembler source code for red bugs or black bugs the error will still show the following is the order in which error is detected first for the world map file then in assembler code for red bugs after that in assembler code for black bugs

For example suppose I input incorrect world map and buggy

Bug World

Please upload a world map file:

Choose Fileserrortestworldcase1.txt

Please upload a bug assembler source code file #1 (for red bugs):

Choose Filesbuggyerror1.txt

Please upload a bug assembler source code file #2 (for black bugs):

Choose Filesbuggyerror2.txt

Please input the number of iterations (minimum 100):

10000

Do you want to log the results of the session?

☐

Next

Following will be the output

clabsql.clamv.jacobs-university.de says
Wrong Dimensions.

OK

Food brough home for red bugs:
Black bugs remaining:
Black bugs killed:
Food brough home for black bugs:

Logs will be here

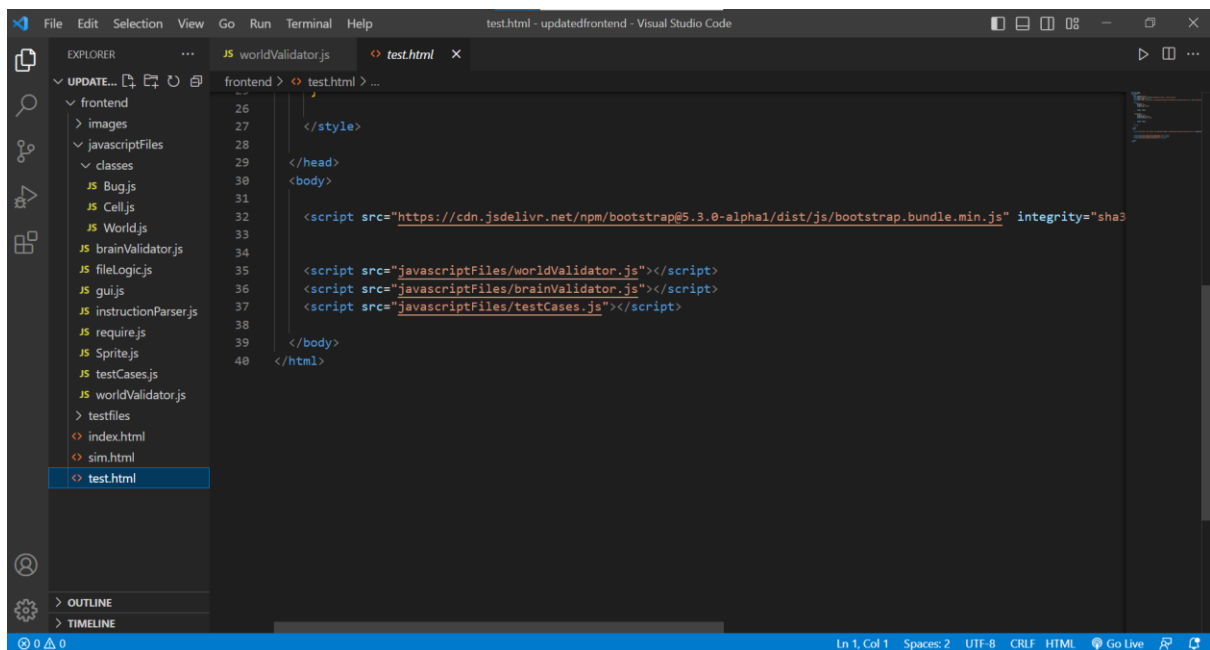
Quit!

Now the final progress I made based on my interpretation of the feedback of the last sprint was create a test.js file to test my functions within my code

This is the test.js file which will validate the map.js and bug.js

```
JS worldValidator.js JS testCases.js X
frontend > javascriptFiles > JS testCases.js > ...
1
2
3 let testCase1 = "4\n4\n# a # #\n# # - #\n# + # #\n# # #";
4
5 let testCase2 = "10\n11\n# # # # # # # #\n# 9 9 . . . . 3 3 #\n# 9 # . . . . . #\n# . # . . . . . #\n# . . 5
6
7 let testCase3 = "sense ahead 1333 3 food ; [ 0]\nmove 2 0 ; [ 1]\npickup 8 0 ; [ 2]\nflip 3 4 5 ; [ 3]\nreturn left 0 ;
8
9 let testCase4 = "sense notaword 1 3 food ; [ 0]\nmove 2 0 ; [ 1]\npickup 8 0 ; [ 2]\nflip 3 4 5 ; [ 3]\nreturn left 0 ;
10
11
12 console.log(validateMap(testCase1));
13 console.log(validateMap(testCase2));
14
15 console.log(validateBrain(testCase3));
16 console.log(validateBrain(testCase4));
17
```

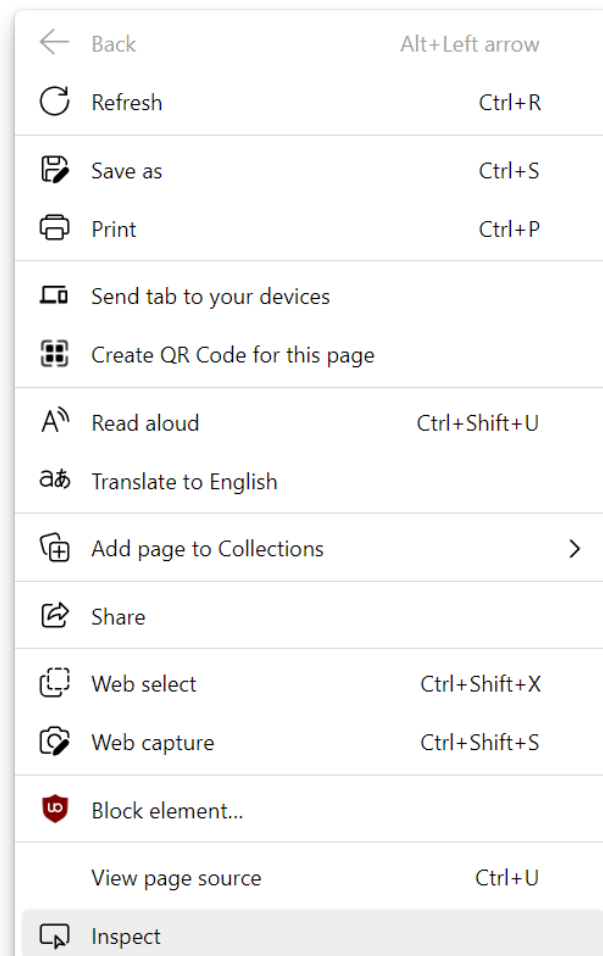
test.js and test.html are linked



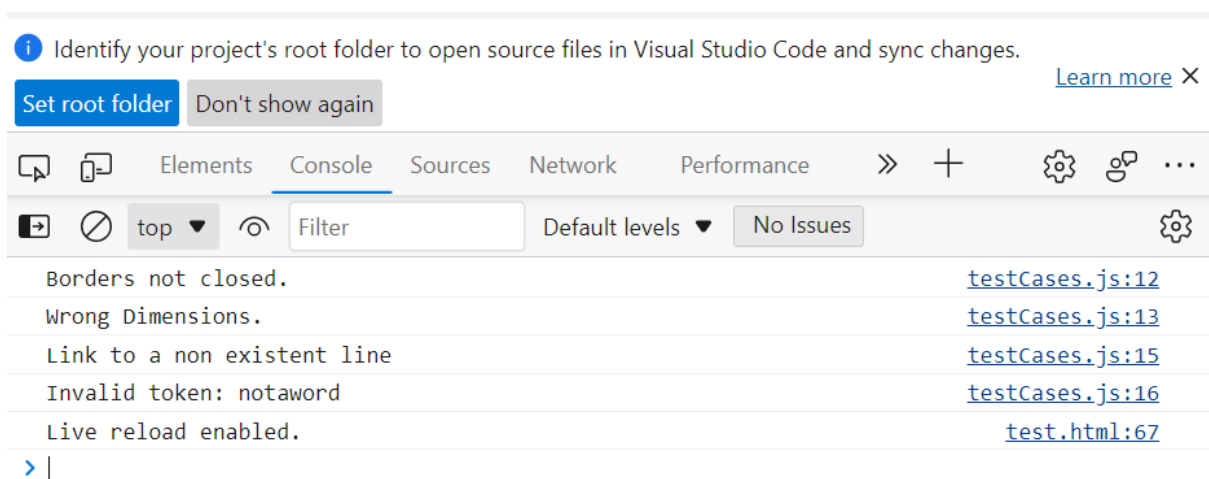
```
File Edit Selection View Go Run Terminal Help test.html - updatedfrontend - Visual Studio Code
EXPLORER
  frontend
    images
    javascriptFiles
      classes
        Bug.js
        Cell.js
        World.js
        brainValidator.js
        fileLogic.js
        gui.js
        instructionParser.js
        require.js
        Sprite.js
        testCases.js
        worldValidator.js
    testfiles
      index.html
      sim.html
      test.html
  OUTLINE
  TIMELINE

26
27 </style>
28
29 </head>
30 <body>
31
32 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha3
33
34
35 <script src="javascriptFiles/worldValidator.js"></script>
36 <script src="javascriptFiles/brainValidator.js"></script>
37 <script src="javascriptFiles/testCases.js"></script>
38
39 </body>
40 </html>
```


After pressing Go live in test.html we go to the page and press inspect



After pressing inspect we go on console and we can see the test case validation



Link to the website: <http://clabsql.clamv.jacobs-university.de/~ashandybin/>