Shawn Wu
Eric Marcelo

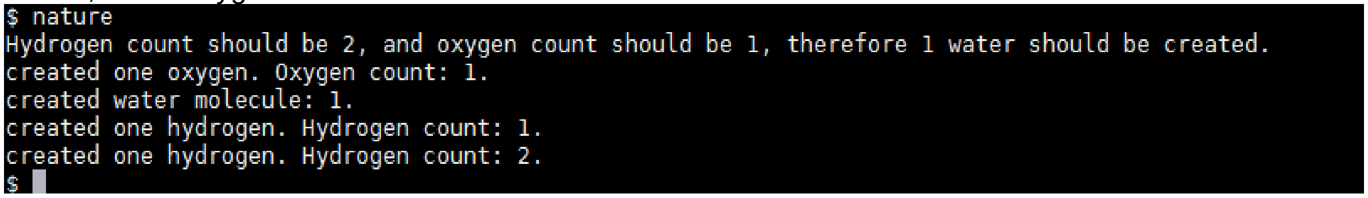
Project 2 Report

Before we begin the report, we would like to notify you that we are using 3 slack days for this project.

For our project, we separated part 1 and part 2 into two different folders. For part 2, our only test case is NULL.c. While for part 1, we have a total of 10 test cases.

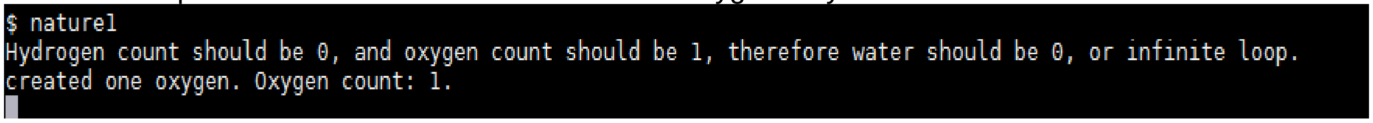
Part 1.3 H20 Problem 1

Here is a snapshot of nature.c, our base case for the H2O problem. We created 2 Hydrogen thread, and 1 oxygen thread in order to make 1 water molecule.



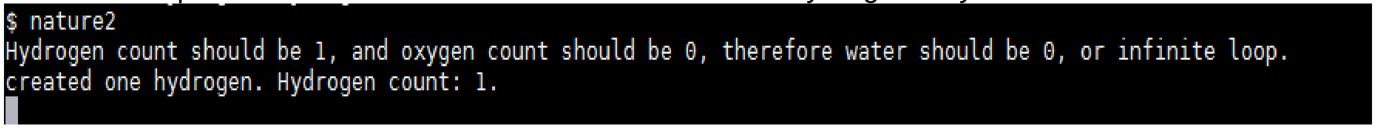
```
$ nature
Hydrogen count should be 2, and oxygen count should be 1, therefore 1 water should be created.
created one oxygen. Oxygen count: 1.
created water molecule: 1.
created one hydrogen. Hydrogen count: 1.
created one hydrogen. Hydrogen count: 2.
$
```

Here is a snapshot of nature1.c where we created 1 oxygen only.



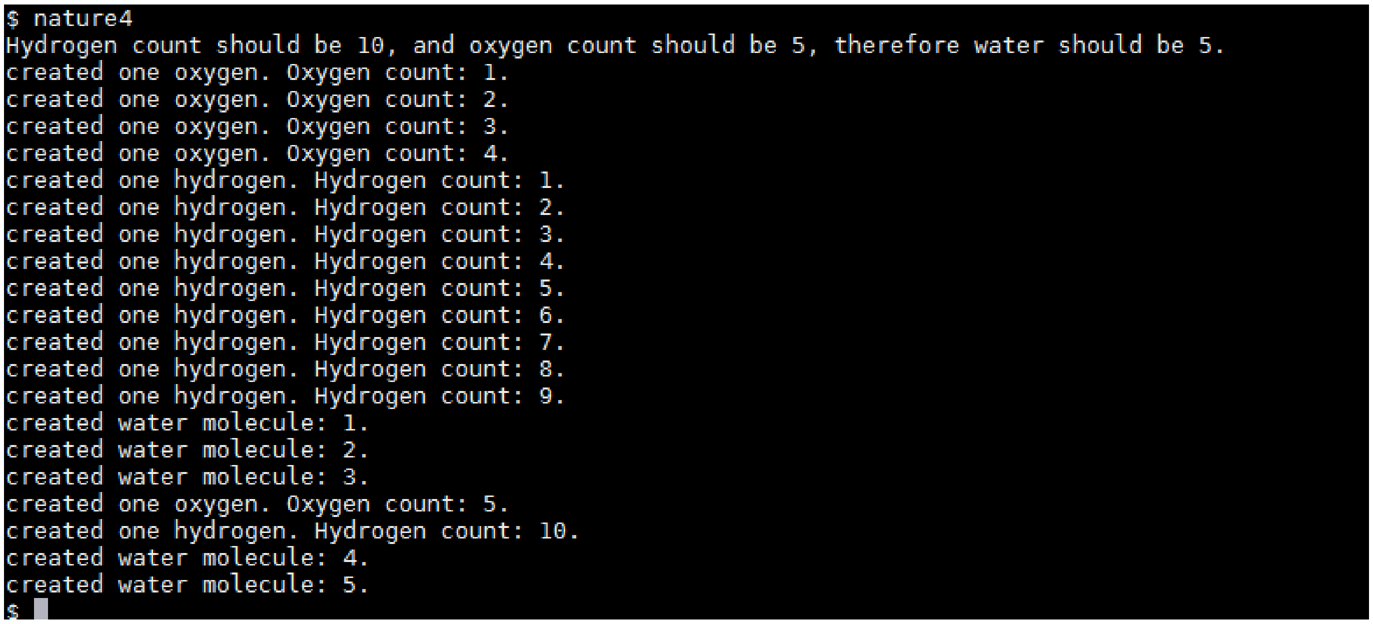
```
$ nature1
Hydrogen count should be 0, and oxygen count should be 1, therefore water should be 0, or infinite loop.
created one oxygen. Oxygen count: 1.
```

Here is a snapshot of our test nature2.c where we created 1 hydrogen only.



```
$ nature2
Hydrogen count should be 1, and oxygen count should be 0, therefore water should be 0, or infinite loop.
created one hydrogen. Hydrogen count: 1.
```

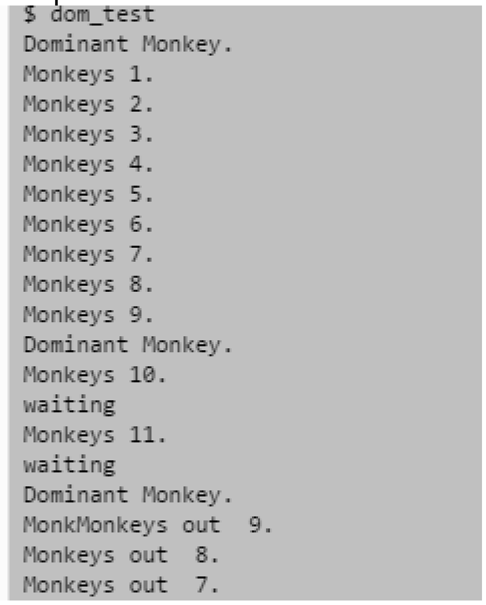
Here is a snapshot of our test nature4.c where we created 5 oxygen and 10 hydrogen in order to make 5 water molecules.



```
$ nature4
Hydrogen count should be 10, and oxygen count should be 5, therefore water should be 5.
created one oxygen. Oxygen count: 1.
created one oxygen. Oxygen count: 2.
created one oxygen. Oxygen count: 3.
created one oxygen. Oxygen count: 4.
created one hydrogen. Hydrogen count: 1.
created one hydrogen. Hydrogen count: 2.
created one hydrogen. Hydrogen count: 3.
created one hydrogen. Hydrogen count: 4.
created one hydrogen. Hydrogen count: 5.
created one hydrogen. Hydrogen count: 6.
created one hydrogen. Hydrogen count: 7.
created one hydrogen. Hydrogen count: 8.
created one hydrogen. Hydrogen count: 9.
created water molecule: 1.
created water molecule: 2.
created water molecule: 3.
created one oxygen. Oxygen count: 5.
created one hydrogen. Hydrogen count: 10.
created water molecule: 4.
created water molecule: 5.
$
```

Part 1.3 Dominant Monkey Problem 2

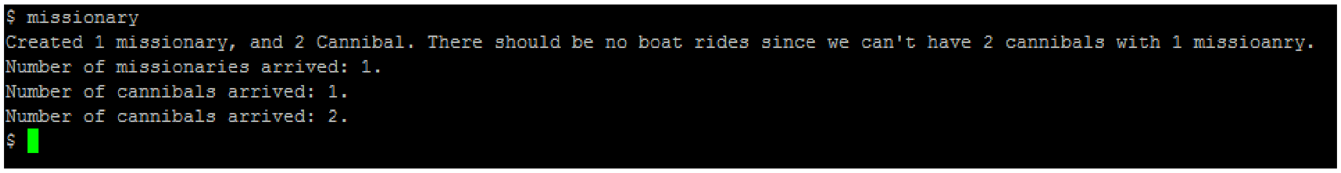
Here is a snapshot of the dominant monkey problem test case. Anytime waiting is printed to the screen, the monkeys are waiting for a dominant monkey to finish. There are a total of two in this screenshot. It is difficult to cause this to occur and sometimes the code does not run to completion.



```
$ dom_test
Dominant Monkey.
Monkeys 1.
Monkeys 2.
Monkeys 3.
Monkeys 4.
Monkeys 5.
Monkeys 6.
Monkeys 7.
Monkeys 8.
Monkeys 9.
Dominant Monkey.
Monkeys 10.
waiting
Monkeys 11.
waiting
Dominant Monkey.
MonkMonkeys out  9.
Monkeys out  8.
Monkeys out  7.
```

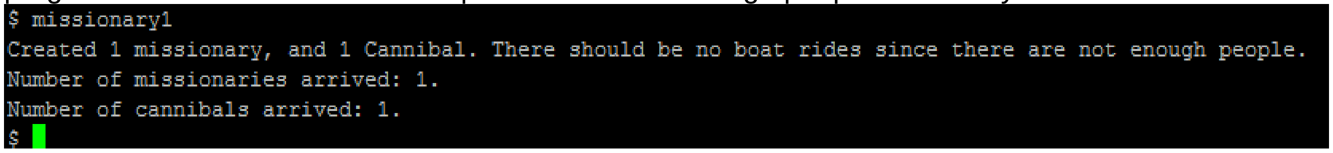
Part 1.3 Missionary Problem 3

Here is a snapshot of missionary.c, our base case where we made 1 missionaries, and two cannibals. No boats are set off since we cannot have such a combination.



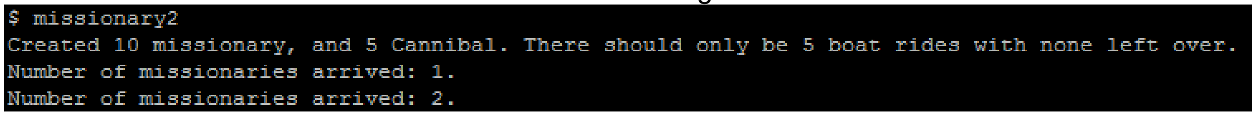
```
$ missionary
Created 1 missionary, and 2 Cannibal. There should be no boat rides since we can't have 2 cannibals with 1 missioanry.
Number of missionaries arrived: 1.
Number of cannibals arrived: 1.
Number of cannibals arrived: 2.
$
```

Here is a snapshot of missionary1.c where we created only 1 missionary and 1 cannibal. The program should be in an infinite loop as there is not enough people to set off yet.



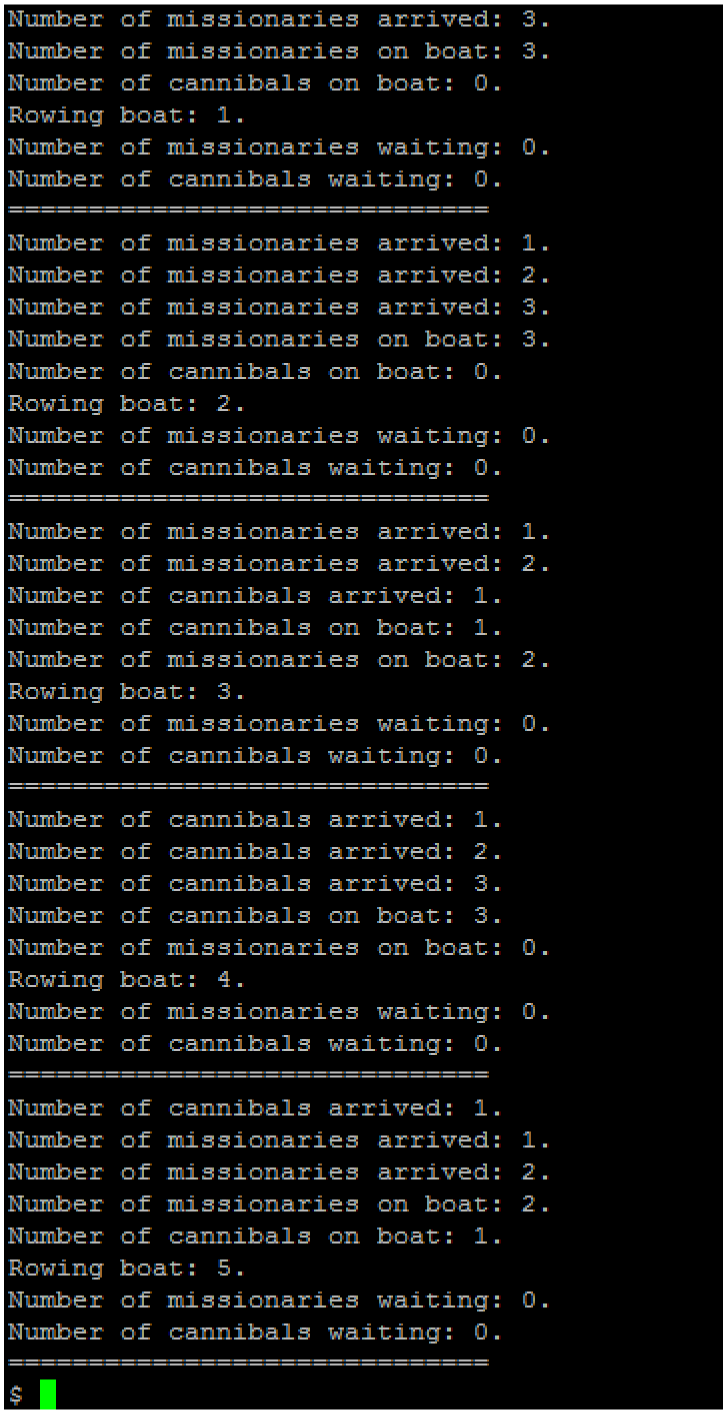
```
$ missionary1
Created 1 missionary, and 1 Cannibal. There should be no boat rides since there are not enough people.
Number of missionaries arrived: 1.
Number of cannibals arrived: 1.
$ 
```

Here is a snapshot of missionary2.c where we created 5 cannibals and 10 missionaries. There should be a total of five boat rides with none remaining.

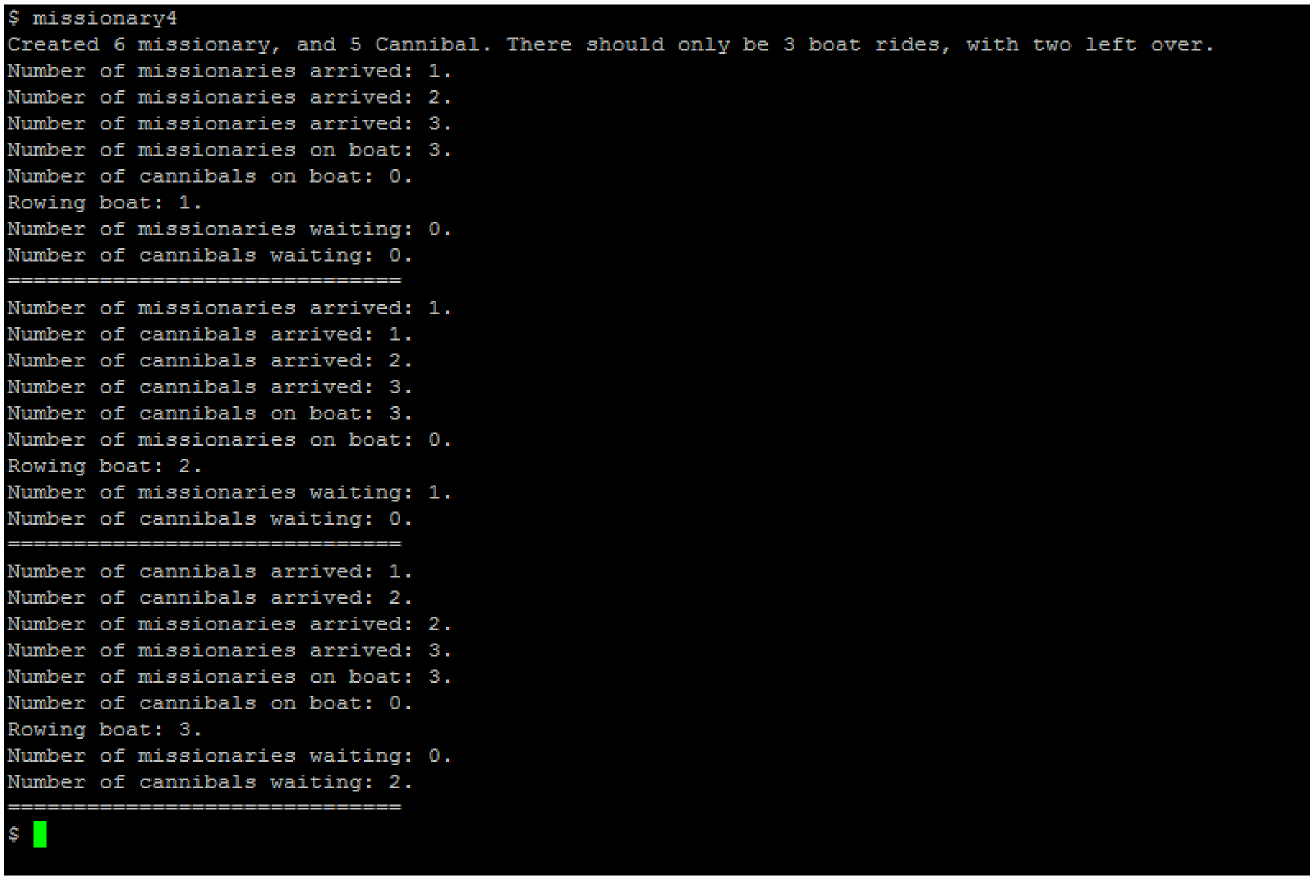


```
$ missionary2
Created 10 missionary, and 5 Cannibal. There should only be 5 boat rides with none left over.
Number of missionaries arrived: 1.
Number of missionaries arrived: 2.
```

Image continued below


```
Number of missionaries arrived: 3.
Number of missionaries on boat: 3.
Number of cannibals on boat: 0.
Rowing boat: 1.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
Number of missionaries arrived: 1.
Number of missionaries arrived: 2.
Number of missionaries arrived: 3.
Number of missionaries on boat: 3.
Number of cannibals on boat: 0.
Rowing boat: 2.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
Number of missionaries arrived: 1.
Number of missionaries arrived: 2.
Number of cannibals arrived: 1.
Number of cannibals on boat: 1.
Number of missionaries on boat: 2.
Rowing boat: 3.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
Number of cannibals arrived: 1.
Number of cannibals arrived: 2.
Number of cannibals arrived: 3.
Number of cannibals on boat: 3.
Number of missionaries on boat: 0.
Rowing boat: 4.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
Number of cannibals arrived: 1.
Number of missionaries arrived: 1.
Number of missionaries arrived: 2.
Number of missionaries on boat: 2.
Number of cannibals on boat: 1.
Rowing boat: 5.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
$ 
```

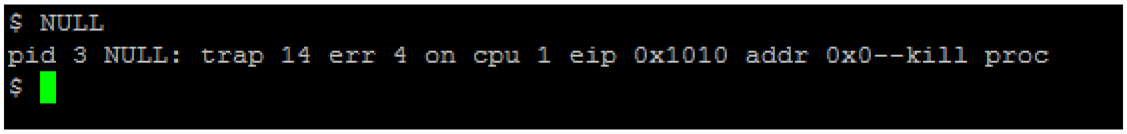
Here is a snapshot of missionary4.c where we created 5 cannibals and 6 missionary. There should be only 3 boats setting off, and two people left over. The options are either 2 cannibals left over, or 1 missionary and 1 cannibal left over.



```
$ missionary4
Created 6 missionary, and 5 Cannibal. There should only be 3 boat rides, with two left over.
Number of missionaries arrived: 1.
Number of missionaries arrived: 2.
Number of missionaries arrived: 3.
Number of missionaries on boat: 3.
Number of cannibals on boat: 0.
Rowing boat: 1.
Number of missionaries waiting: 0.
Number of cannibals waiting: 0.
==============================
Number of missionaries arrived: 1.
Number of cannibals arrived: 1.
Number of cannibals arrived: 2.
Number of cannibals arrived: 3.
Number of cannibals on boat: 3.
Number of missionaries on boat: 0.
Rowing boat: 2.
Number of missionaries waiting: 1.
Number of cannibals waiting: 0.
==============================
Number of cannibals arrived: 1.
Number of cannibals arrived: 2.
Number of missionaries arrived: 2.
Number of missionaries arrived: 3.
Number of missionaries on boat: 3.
Number of cannibals on boat: 0.
Rowing boat: 3.
Number of missionaries waiting: 0.
Number of cannibals waiting: 2.
==============================
$ 
```

<u>Part 2.1</u>

Here, we run NULL.c. Once the program dereferences a null-pointer, it outputs the panic trap 14 error and exits.



```
$ NULL
pid 3 NULL: trap 14 err 4 on cpu 1 eip 0x1010 addr 0x0--kill proc
$ 
```

This concludes our report. Thank you!