Fish Machines:

Intel has donated a cluster of 14 machines for our use in this course. These machines all have identical software and hardware, so if your code runs on one machine, it'll run when we grade your lab! Because of this, all of your code must compile and run on the provided fish machines: no exceptions!

You can access these machines by ssh'ing into fish.ics.cs.cmu.edu . For more information please see http://www.cs.cmu.edu/~213/labmachines.html .

Autolab:

Autolab (https://autolab.cs.cmu.edu) is a automated lab handout/handin website that will keep track of your progress on the labs this semester and provide all your grade information to you. When you submit your assignment to autolab, it will grade and immediately provide you with your score on the lab. With few exceptions, this will be the exact grade you will receive for that assignment!

->All work must be handed into autolab, we cannot accept assignments turned in any other way

->The tests that autolab runs against your code are given to you in the lab handout. Please do not use autolab as a compiler or as a grader, you can do that yourself on the fish machines!

->After you submit to autolab, please check your score! If autolab does not give you a score, you need to determine what went wrong and fix it!

->Autolab will not accept ANY submissions after the final end date of the lab, regardless of any late days you might have.

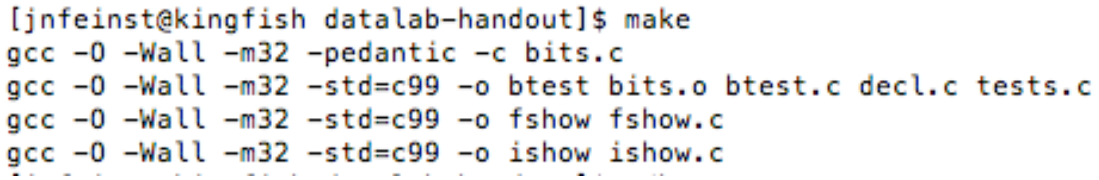
Testing Code:

Per the datalab writeup available on Autolab, you have three methods available to test your code.

btest:

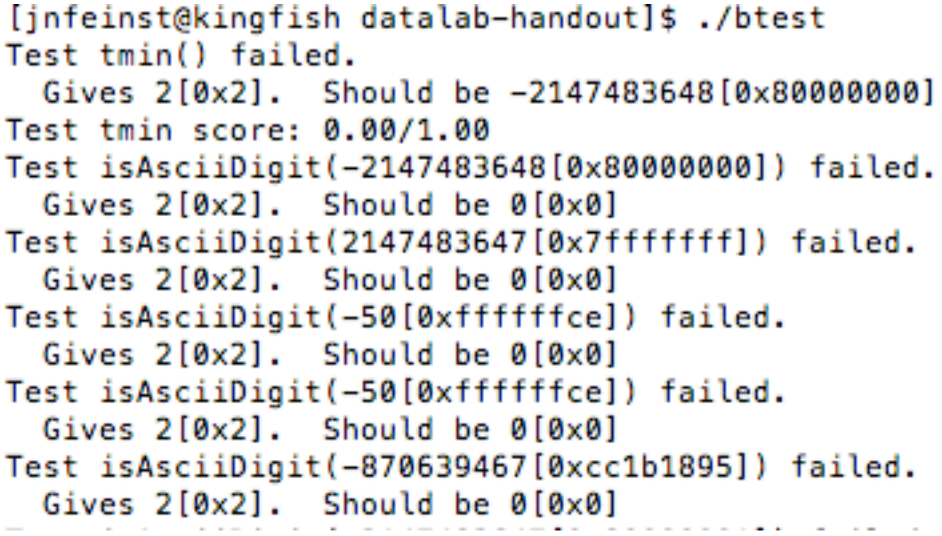
A simple, non-exhaustive function that tests the correction of your code.

First, run the “make” command within your datalab-handout folder to build the btest function:



```
[jnfeinst@kingfish datalab-handout]$ make
gcc -O -Wall -m32 -pedantic -c bits.c
gcc -O -Wall -m32 -std=c99 -o btest bits.o btest.c decl.c tests.c
gcc -O -Wall -m32 -std=c99 -o fshow fshow.c
gcc -O -Wall -m32 -std=c99 -o ishow ishow.c
```

Next, run ./btest:



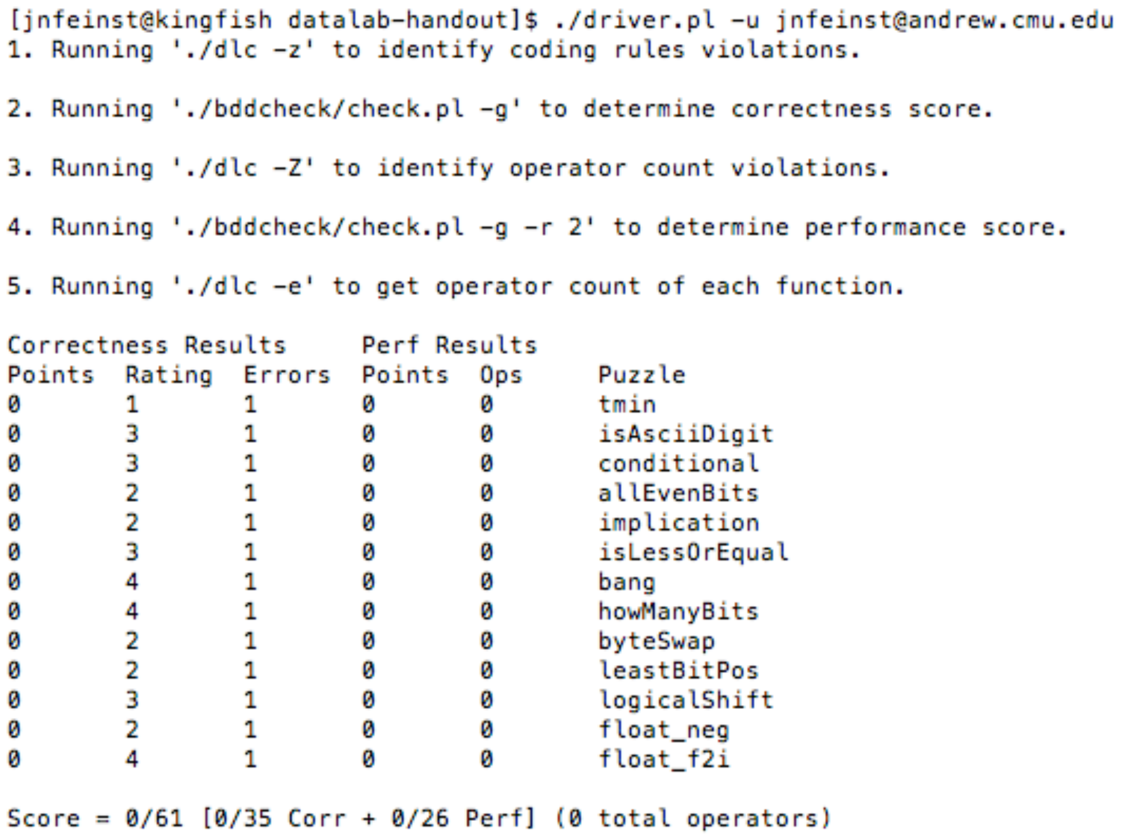
```
[jnfeinst@kingfish datalab-handout]$ ./btest
Test tmin() failed.
  Gives 2[0x2].  Should be -2147483648[0x80000000]
Test tmin score: 0.00/1.00
Test isAsciiDigit(-2147483648[0x80000000]) failed.
  Gives 2[0x2].  Should be 0[0x0]
Test isAsciiDigit(2147483647[0x7fffffff]) failed.
  Gives 2[0x2].  Should be 0[0x0]
Test isAsciiDigit(-50[0xffffffce]) failed.
  Gives 2[0x2].  Should be 0[0x0]
Test isAsciiDigit(-50[0xffffffce]) failed.
  Gives 2[0x2].  Should be 0[0x0]
Test isAsciiDigit(-870639467[0xcc1b1895]) failed.
  Gives 2[0x2].  Should be 0[0x0]
```

The output will include whether a test succeeded or failed. This does NOT test all cases, while bddcheck does.

driver:

The driver script will create an unofficial grade report on Autolab so you can get an idea as to what grade you will get. The driver script does NOT submit your code to Autolab! It is only an unofficial submission so you can see what grade you will get.

To run the driver script, execute
“./driver.pl –u YOURUSERNAME@andrew.cmu.edu” within your datalab-handout folder:



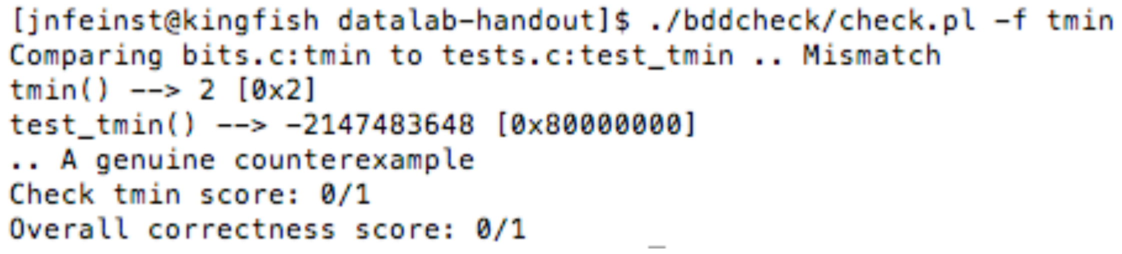
```
[jnfeinst@kingfish datalab-handout]$ ./driver.pl -u jnfeinst@andrew.cmu.edu
1. Running './dlc -z' to identify coding rules violations.

2. Running './bddcheck/check.pl -g' to determine correctness score.

3. Running './dlc -Z' to identify operator count violations.

4. Running './bddcheck/check.pl -g -r 2' to determine performance score.

5. Running './dlc -e' to get operator count of each function.

Correctness Results     Perf Results
Points  Rating  Errors  Points  Ops     Puzzle
0       1       1       0       0       tmin
0       3       1       0       0       isAsciiDigit
0       3       1       0       0       conditional
0       2       1       0       0       allEvenBits
0       2       1       0       0       implication
0       3       1       0       0       isLessOrEqual
0       4       1       0       0       bang
0       4       1       0       0       howManyBits
0       2       1       0       0       byteSwap
0       2       1       0       0       leastBitPos
0       3       1       0       0       logicalShift
0       2       1       0       0       float_neg
0       4       1       0       0       float_f2i

Score = 0/61 [0/35 Corr + 0/26 Perf] (0 total operators)
```

As you can see in this example, I got 0/61 because all of my functions are incorrect. You MUST submit your code to Autolab officially in order to get a real grade on the assignment.

bddcheck:

Sometimes all of your cases in btest will pass but you will not get full credit from the driver script. You can run bddcheck to perform a comprehensive test on your function to determine what is wrong.

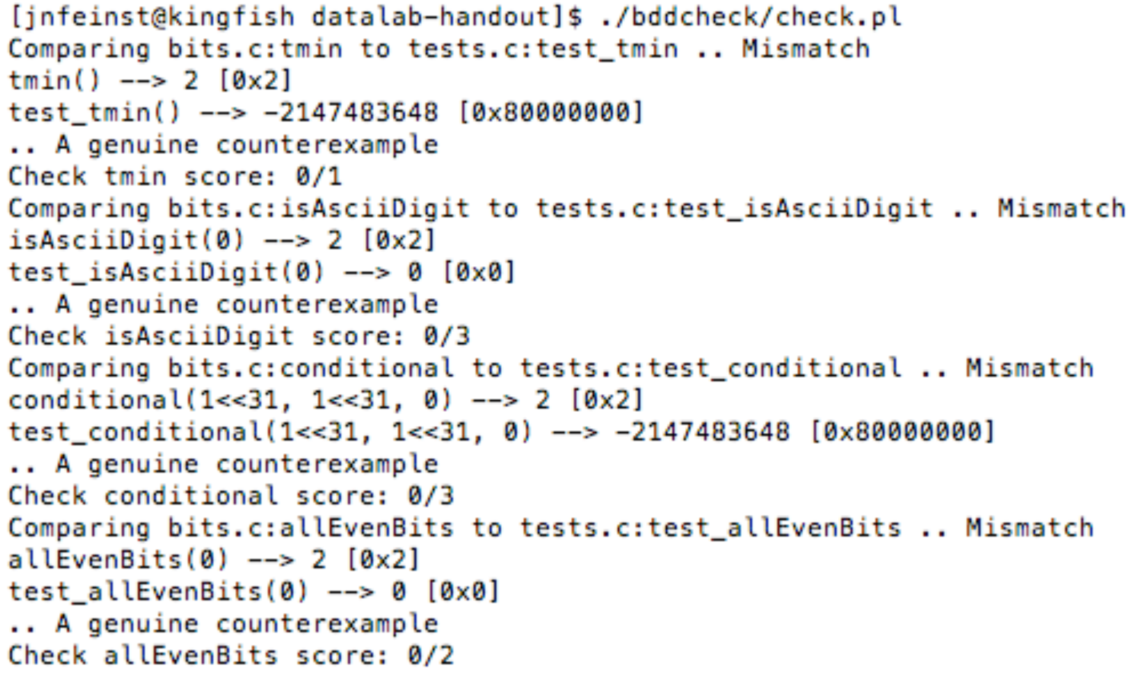
You can run bddcheck on one function or all of your functions. To run it on one function, execute “./bddcheck/check.pl –f FUNCTION”:



```
[jnfeinst@kingfish datalab-handout]$ ./bddcheck/check.pl -f tmin
Comparing bits.c:tmin to tests.c:test_tmin .. Mismatch
tmin() --> 2 [0x2]
test_tmin() --> -2147483648 [0x80000000]
.. A genuine counterexample
Check tmin score: 0/1
Overall correctness score: 0/1
```

FUNCTION (in this case tmin) represents your function output, while test_FUNCTION is the correct output. In this example, the output of my tmin()

was 0x2, while the output of test_tmin() was 0x80000000. Something is seriously wrong with my code!
You can also run bddcheck on all of your functions at once. To do this, execute "./bddcheck/check.pl":



```
[jnfeinst@kingfish datalab-handout]$ ./bddcheck/check.pl
Comparing bits.c:tmin to tests.c:test_tmin .. Mismatch
tmin() --> 2 [0x2]
test_tmin() --> -2147483648 [0x80000000]
.. A genuine counterexample
Check tmin score: 0/1
Comparing bits.c:isAsciiDigit to tests.c:test_isAsciiDigit .. Mismatch
isAsciiDigit(0) --> 2 [0x2]
test_isAsciiDigit(0) --> 0 [0x0]
.. A genuine counterexample
Check isAsciiDigit score: 0/3
Comparing bits.c:conditional to tests.c:test_conditional .. Mismatch
conditional(1<<31, 1<<31, 0) --> 2 [0x2]
test_conditional(1<<31, 1<<31, 0) --> -2147483648 [0x80000000]
.. A genuine counterexample
Check conditional score: 0/3
Comparing bits.c:allEvenBits to tests.c:test_allEvenBits .. Mismatch
allEvenBits(0) --> 2 [0x2]
test_allEvenBits(0) --> 0 [0x0]
.. A genuine counterexample
Check allEvenBits score: 0/2
```

Using these three methods (btest, driver, bddcheck), you should be able to get a good picture as to how your functions work. Driver will let you know what your current grade would be, while btest and bddcheck will let you know which cases work and which do not. When you are debugging, look through your code to figure out why some cases fail and correct them!

Note: Autolab is not intended to be a compiler or test bench. While it can act as such, it does not have the capacity to do so for everyone on a regular basis. You should be using the driver to calculate your grade, and then submit officially when you know that your code works.