FAQAS2: Fault-based Automated Quality Assurance Assessment for Space 2

IMPROVE MUTATION TESTING IN SPACE SOFTWARE SYSTEMS

Fabrizio Pastore
University of Luxembourg
fabrizio.pastore@uni.lu

ADCSS

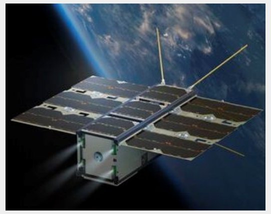
October 23rd, 2024

Fault-based, Automated Quality Assurance Assessment and Augmentation for Space Software 2

SnT/University of Luxembourg

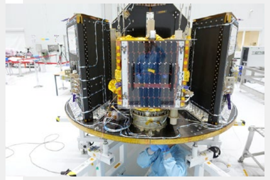
- Technology provider

Gomspace Luxembourg (GSL)



- Develop Nanosatellites
- Case study provider
- Technology validator

LuxSpace



- Develop Microsatelites
- Case study provider
- Technology validator

Huld - Finland

- Develop SW solutions
- ISVV supplier
- Case study provider
- Technology validator

Dr. Fabrizio Pastore

Dr. Jaekwon Lee

Prof. Lionel Briand

Enrico Viganò

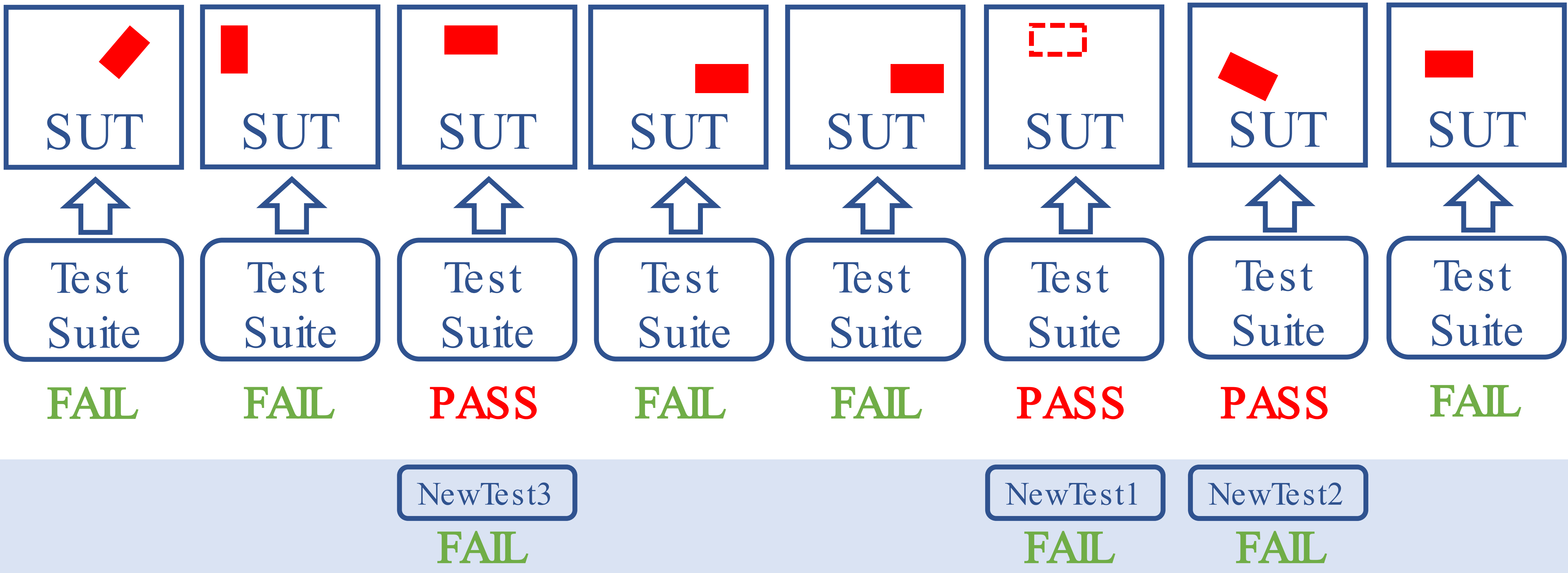
https://faqas.uni.lu/

Software testing is prevalent in V&V

Word cloud for
ECSS-Q-ST-80C standard

How to ensure thorough testing?

Mutation Analysis and Testing



Improve with Automatically Generated Test Cases

FAQAS-2 toolset

Mutate C/C++ source code

Mutate the data exchanged by C/C++ components

Generate test cases in C/C++ that detect the injected faults

Generate test cases in C to exercise more “data partitions”

FAQAS-2 toolset

Mutate C/C++ source code

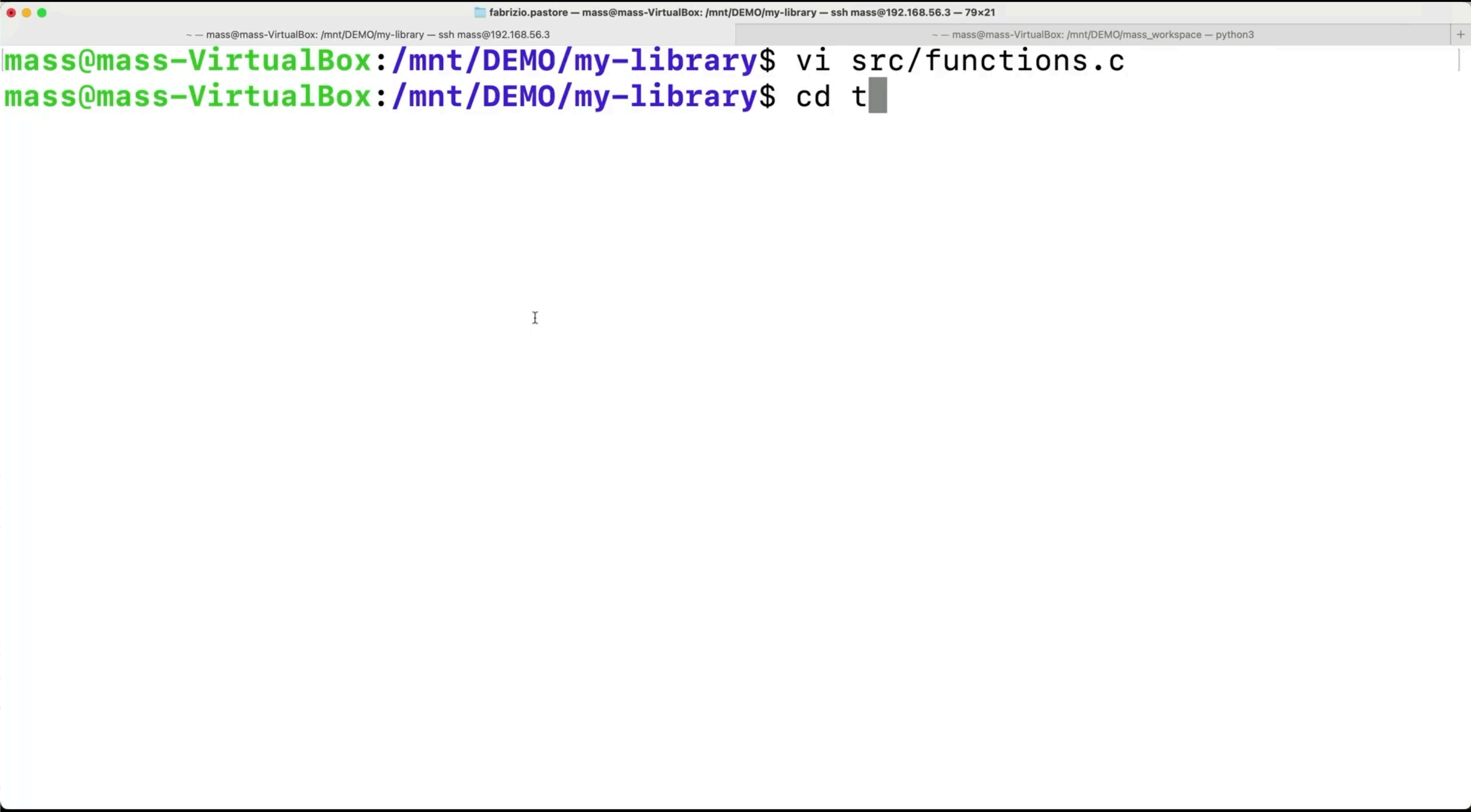
Generate
test cases in C/C++
that detect the injected fault

DEMO

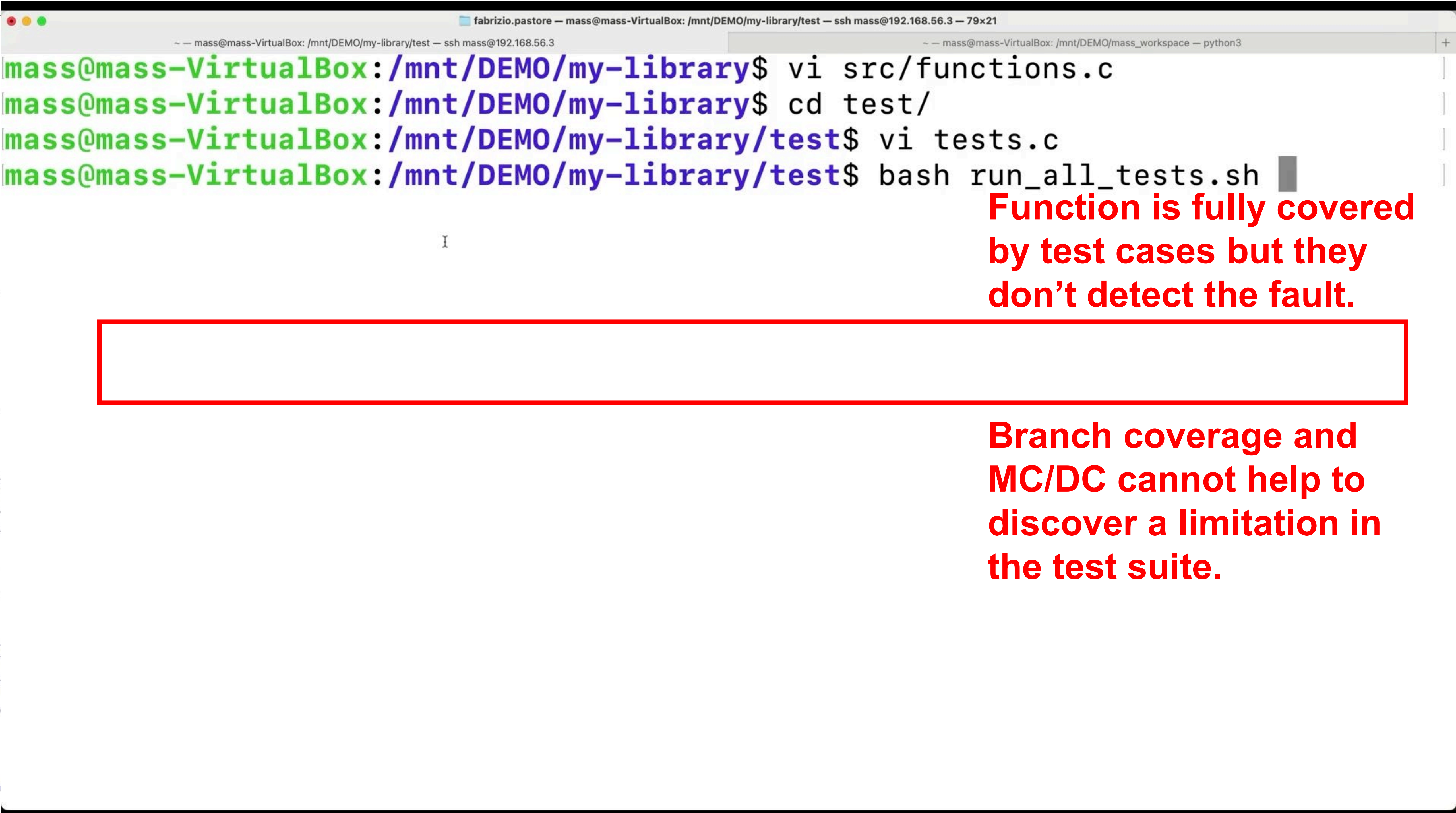
fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/my-library — ssh mass@192.168.56.3 — 79×21
~ — mass@mass-VirtualBox: /mnt/DEMO/my-library — ssh mass@192.168.56.3
~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
mass@mass-VirtualBox:/mnt/DEMO/my-library$ v
```

Faulty implementation



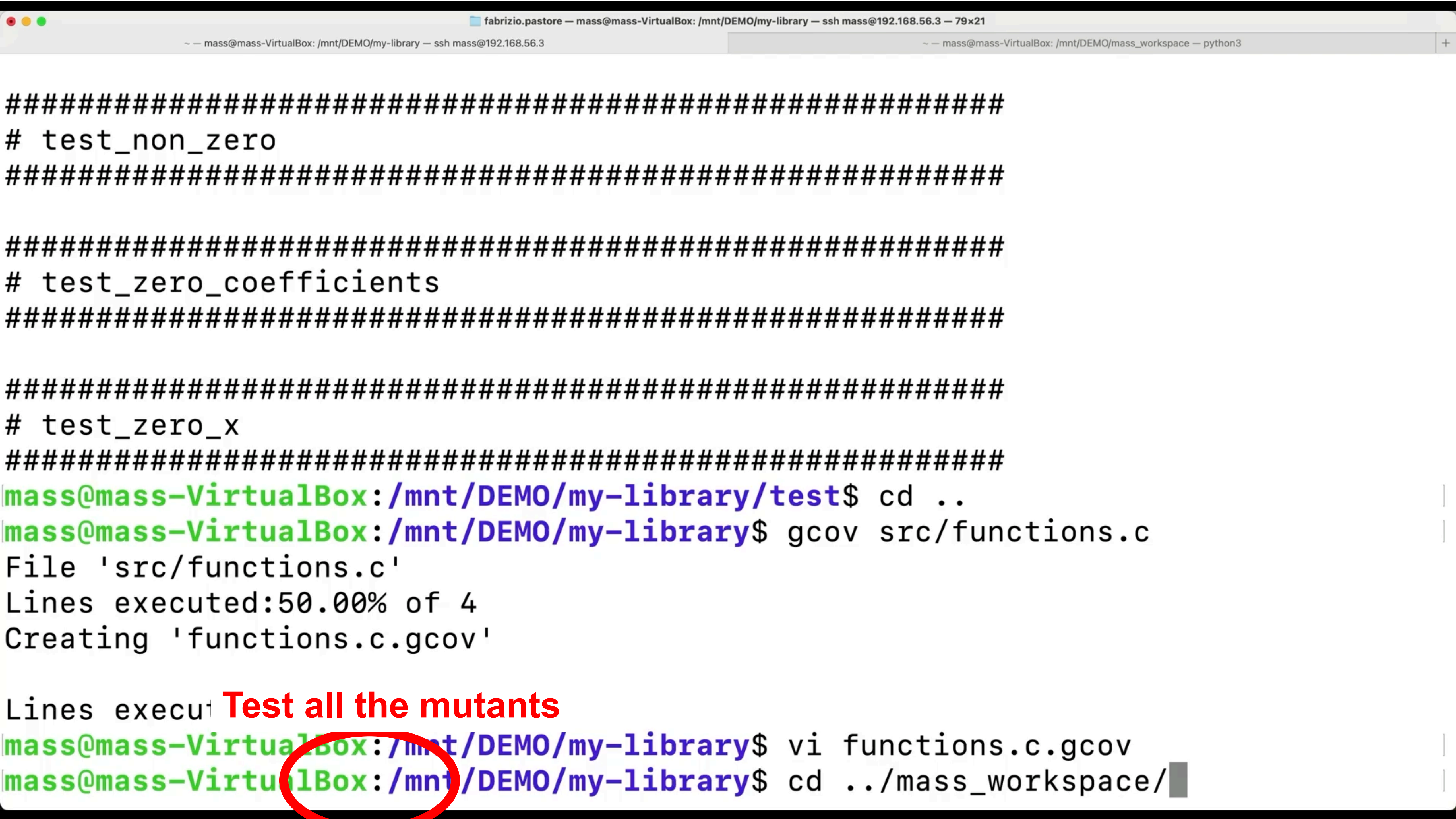
```
mass@mass-VirtualBox:/mnt/DEMO/my-library$ vi src/functions.c
mass@mass-VirtualBox:/mnt/DEMO/my-library$ cd t
```



```
fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/my-library/test — ssh mass@192.168.56.3 — 79×21
~ — mass@mass-VirtualBox: /mnt/DEMO/my-library/test — ssh mass@192.168.56.3
~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3
mass@mass-VirtualBox:/mnt/DEMO/my-library$ vi src/functions.c
mass@mass-VirtualBox:/mnt/DEMO/my-library$ cd test/
mass@mass-VirtualBox:/mnt/DEMO/my-library/test$ vi tests.c
mass@mass-VirtualBox:/mnt/DEMO/my-library/test$ bash run_all_tests.sh
```

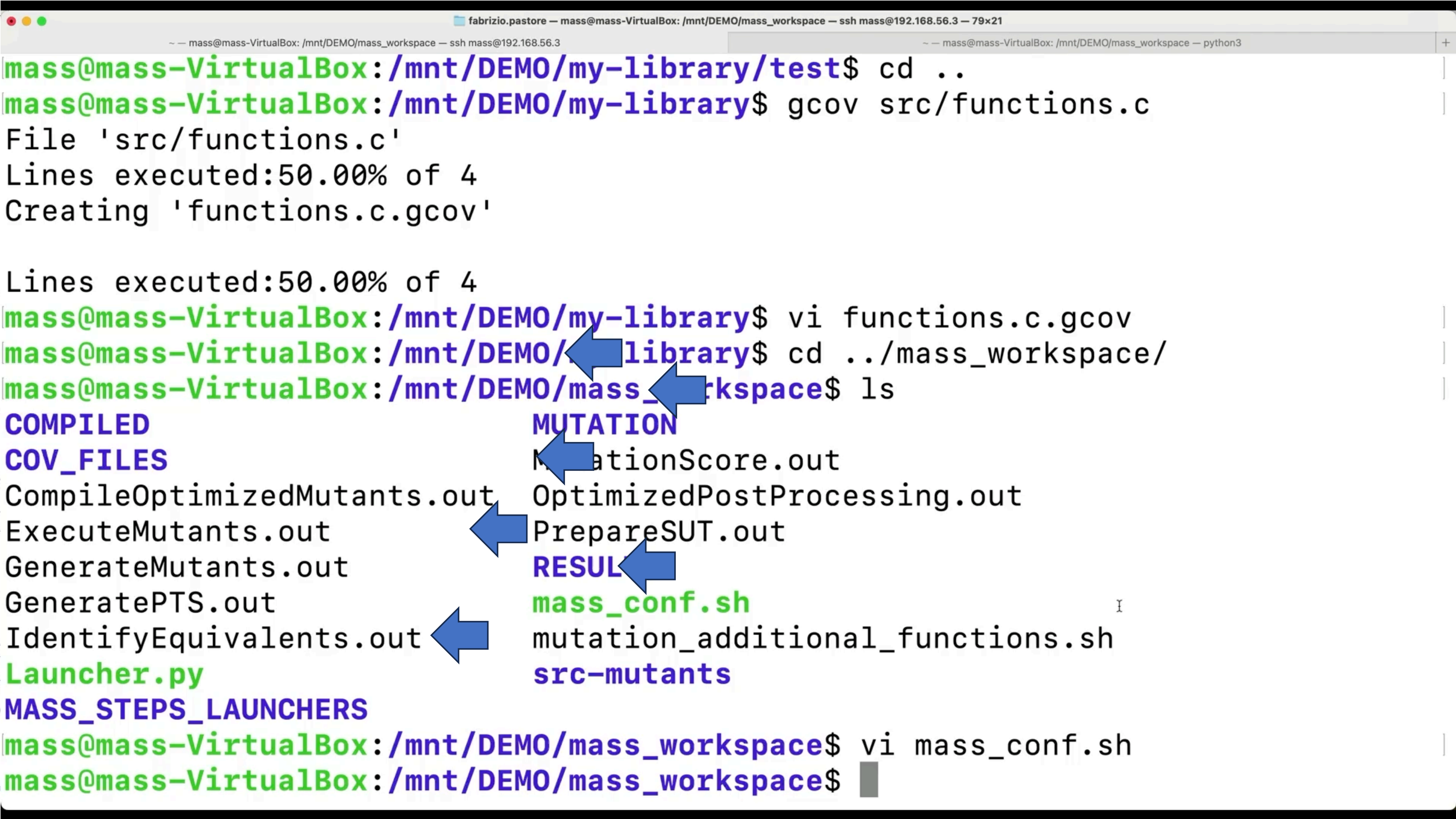
Function is fully covered by test cases but they don't detect the fault.

Branch coverage and MC/DC cannot help to discover a limitation in the test suite.


fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/my-library — ssh mass@192.168.56.3 — 79×21
~ — mass@mass-VirtualBox: /mnt/DEMO/my-library — ssh mass@192.168.56.3
~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
#######################################################
# test_non_zero
#######################################################

#######################################################
# test_zero_coefficients
#######################################################

#######################################################
# test_zero_x
#######################################################
mass@mass-VirtualBox:/mnt/DEMO/my-library/test$ cd ..
mass@mass-VirtualBox:/mnt/DEMO/my-library$ gcov src/functions.c
File 'src/functions.c'
Lines executed:50.00% of 4
Creating 'functions.c.gcov'

Lines execu
mass@mass-VirtualBox:/mnt/DEMO/my-library$ vi functions.c.gcov
mass@mass-VirtualBox:/mnt/DEMO/my-library$ cd ../mass_workspace/
```

Test all the mutants

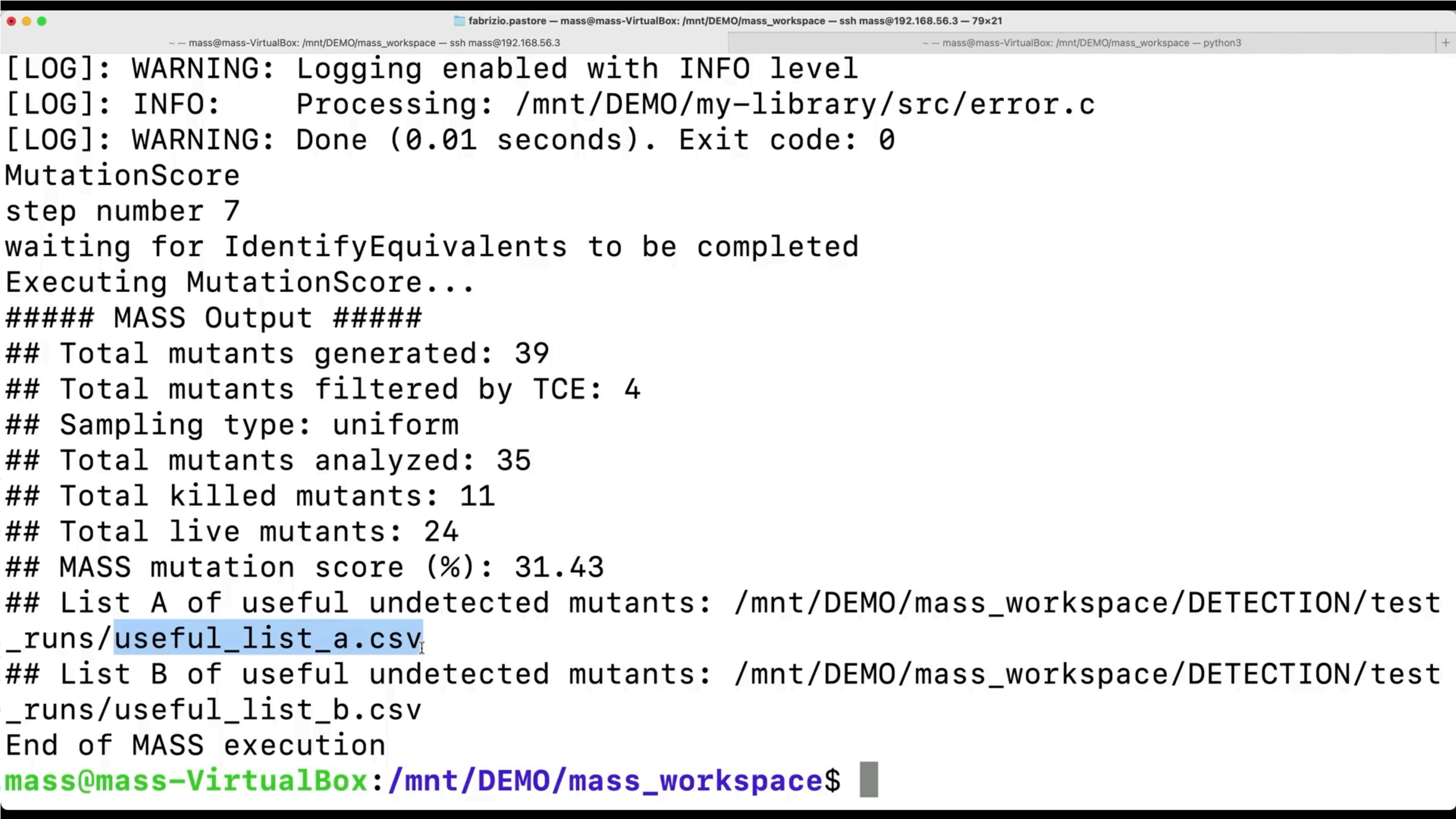


fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
mass@mass-VirtualBox:/mnt/DEMO/my-library/test$ cd ..
mass@mass-VirtualBox:/mnt/DEMO/my-library$ gcov src/functions.c
File 'src/functions.c'
Lines executed:50.00% of 4
Creating 'functions.c.gcov'

Lines executed:50.00% of 4
mass@mass-VirtualBox:/mnt/DEMO/my-library$ vi functions.c.gcov
mass@mass-VirtualBox:/mnt/DEMO/my-library$ cd ../mass_workspace/
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ ls
COMPILED                        MUTATION
COV_FILES                       MutationScore.out
CompileOptimizedMutants.out     OptimizedPostProcessing.out
ExecuteMutants.out              PrepareSUT.out
GenerateMutants.out             RESULTS
GeneratePTS.out                 mass_conf.sh
IdentifyEquivalents.out         mutation_additional_functions.sh
Launcher.py                     src-mutants
MASS_STEPS_LAUNCHERS
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ vi mass_conf.sh
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ 
```

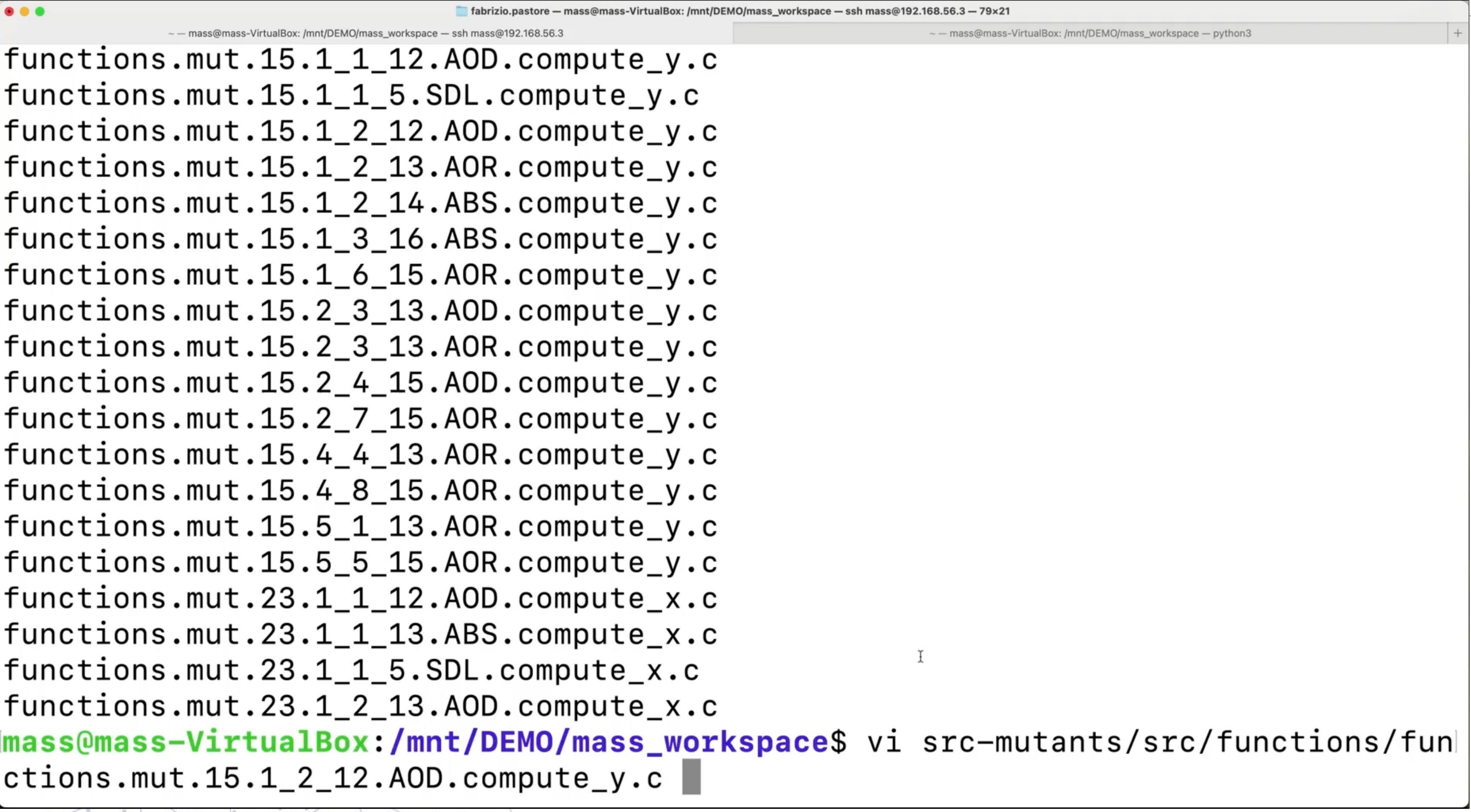


fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
[LOG]: WARNING: Logging enabled with INFO level
[LOG]: INFO:    Processing: /mnt/DEMO/my-library/src/error.c
[LOG]: WARNING: Done (0.01 seconds). Exit code: 0
MutationScore
step number 7
waiting for IdentifyEquivalents to be completed
Executing MutationScore...
##### MASS Output #####
## Total mutants generated: 39
## Total mutants filtered by TCE: 4
## Sampling type: uniform
## Total mutants analyzed: 35
## Total killed mutants: 11
## Total live mutants: 24
## MASS mutation score (%): 31.43
## List A of useful undetected mutants: /mnt/DEMO/mass_workspace/DETECTION/test
_runs/useful_list_a.csv
## List B of useful undetected mutants: /mnt/DEMO/mass_workspace/DETECTION/test
_runs/useful_list_b.csv
End of MASS execution
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$
```

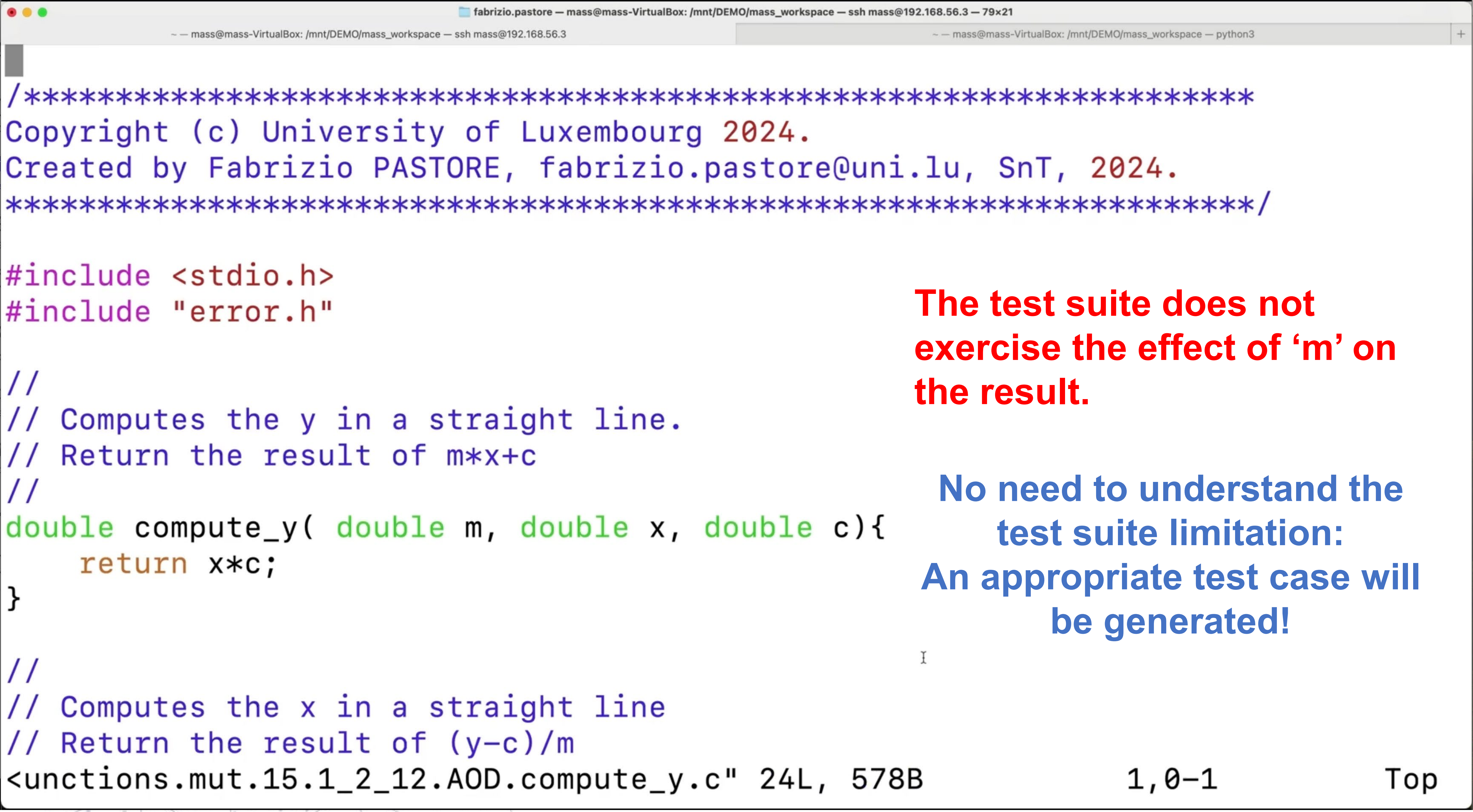


fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
functions.mut.15.1_1_12.AOD.compute_y.c
functions.mut.15.1_1_5.SDL.compute_y.c
functions.mut.15.1_2_12.AOD.compute_y.c
functions.mut.15.1_2_13.AOR.compute_y.c
functions.mut.15.1_2_14.ABS.compute_y.c
functions.mut.15.1_3_16.ABS.compute_y.c
functions.mut.15.1_6_15.AOR.compute_y.c
functions.mut.15.2_3_13.AOD.compute_y.c
functions.mut.15.2_3_13.AOR.compute_y.c
functions.mut.15.2_4_15.AOD.compute_y.c
functions.mut.15.2_7_15.AOR.compute_y.c
functions.mut.15.4_4_13.AOR.compute_y.c
functions.mut.15.4_8_15.AOR.compute_y.c
functions.mut.15.5_1_13.AOR.compute_y.c
functions.mut.15.5_5_15.AOR.compute_y.c
functions.mut.23.1_1_12.AOD.compute_x.c
functions.mut.23.1_1_13.ABS.compute_x.c
functions.mut.23.1_1_5.SDL.compute_x.c
functions.mut.23.1_2_13.AOD.compute_x.c
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ vi src-mutants/src/functions/fun
ctions.mut.15.1_2_12.AOD.compute_y.c
```



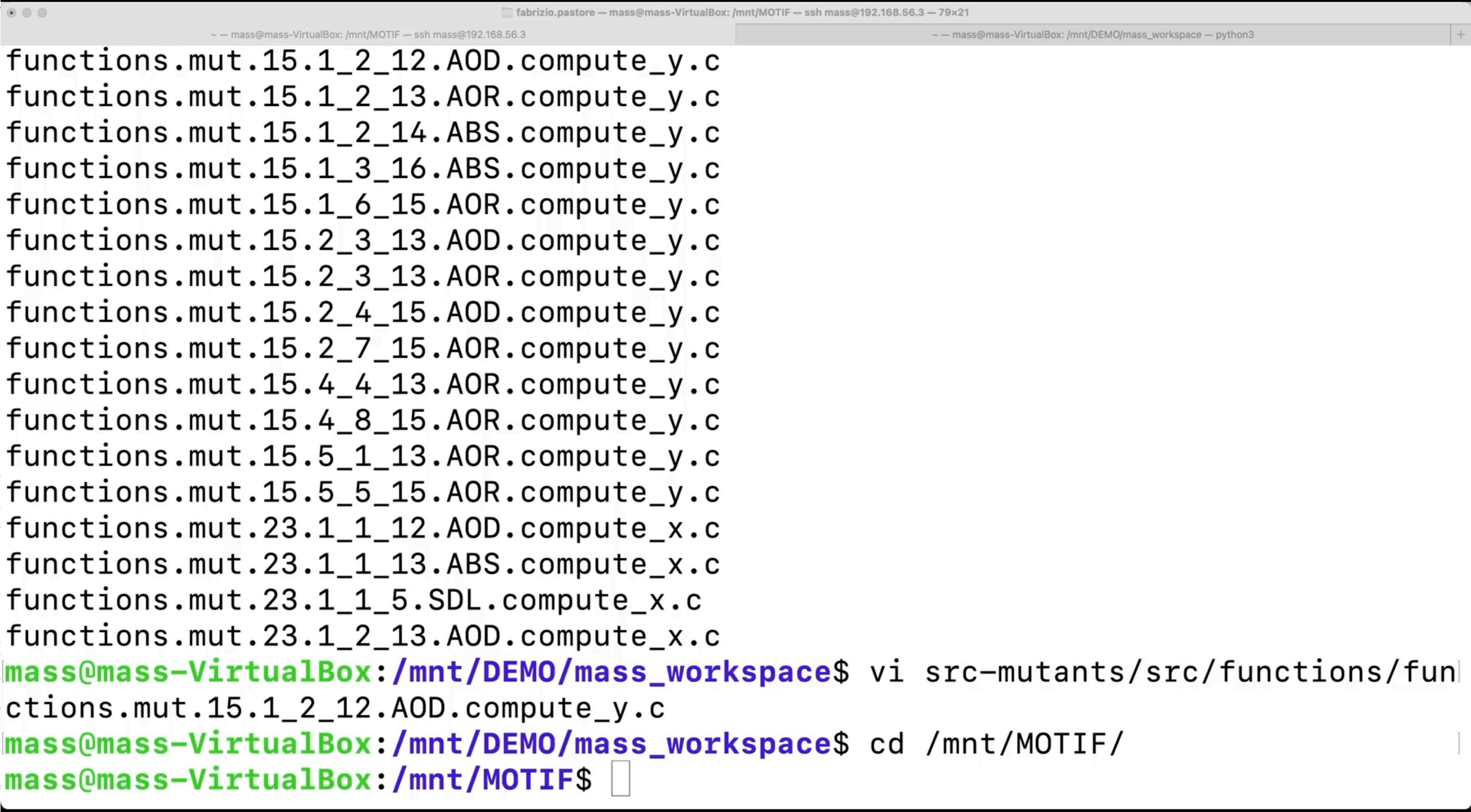
fabrizio.pastore — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — ssh mass@192.168.56.3 | ~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
/*****************************************************************************
Copyright (c) University of Luxembourg 2024.
Created by Fabrizio PASTORE, fabrizio.pastore@uni.lu, SnT, 2024.
*****************************************************************************/

#include <stdio.h>
#include "error.h"

//
// Computes the y in a straight line.
// Return the result of m*x+c
//
double compute_y( double m, double x, double c){
    return x*c;
}

//
// Computes the x in a straight line
// Return the result of (y-c)/m
<unctions.mut.15.1_2_12.AOD.compute_y.c" 24L, 578B          1,0-1          Top
```

The test suite does not exercise the effect of ‘m’ on the result.

No need to understand the test suite limitation: An appropriate test case will be generated!

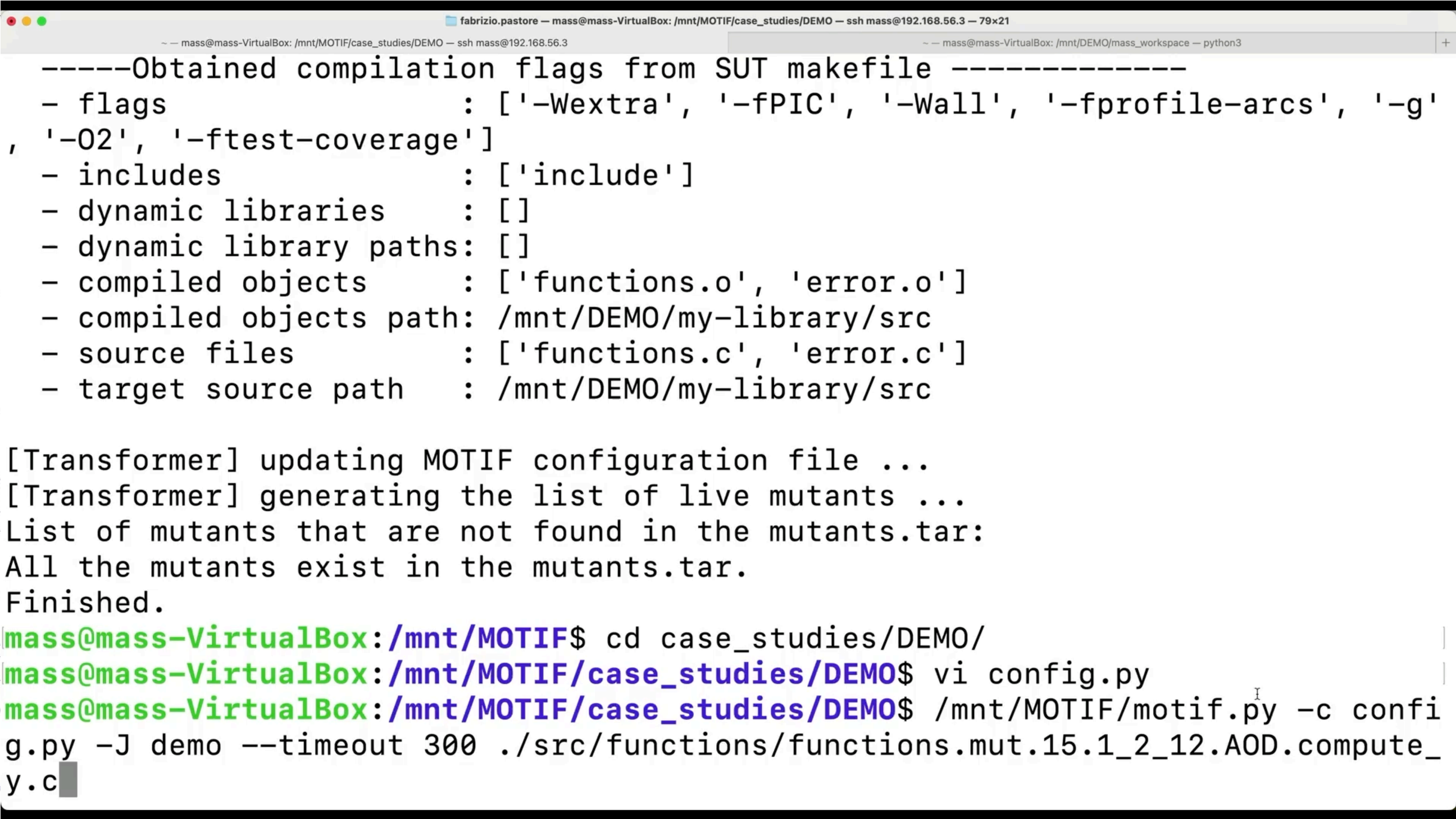


fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/MOTIF — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
functions.mut.15.1_2_12.AOD.compute_y.c
functions.mut.15.1_2_13.AOR.compute_y.c
functions.mut.15.1_2_14.ABS.compute_y.c
functions.mut.15.1_3_16.ABS.compute_y.c
functions.mut.15.1_6_15.AOR.compute_y.c
functions.mut.15.2_3_13.AOD.compute_y.c
functions.mut.15.2_3_13.AOR.compute_y.c
functions.mut.15.2_4_15.AOD.compute_y.c
functions.mut.15.2_7_15.AOR.compute_y.c
functions.mut.15.4_4_13.AOR.compute_y.c
functions.mut.15.4_8_15.AOR.compute_y.c
functions.mut.15.5_1_13.AOR.compute_y.c
functions.mut.15.5_5_15.AOR.compute_y.c
functions.mut.23.1_1_12.AOD.compute_x.c
functions.mut.23.1_1_13.ABS.compute_x.c
functions.mut.23.1_1_5.SDL.compute_x.c
functions.mut.23.1_2_13.AOD.compute_x.c
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ vi src-mutants/src/functions/fun
ctions.mut.15.1_2_12.AOD.compute_y.c
mass@mass-VirtualBox:/mnt/DEMO/mass_workspace$ cd /mnt/MOTIF/
mass@mass-VirtualBox:/mnt/MOTIF$ 
```


fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO — ssh mass@192.168.56.3 — 79×21
~ — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO — ssh mass@192.168.56.3
~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

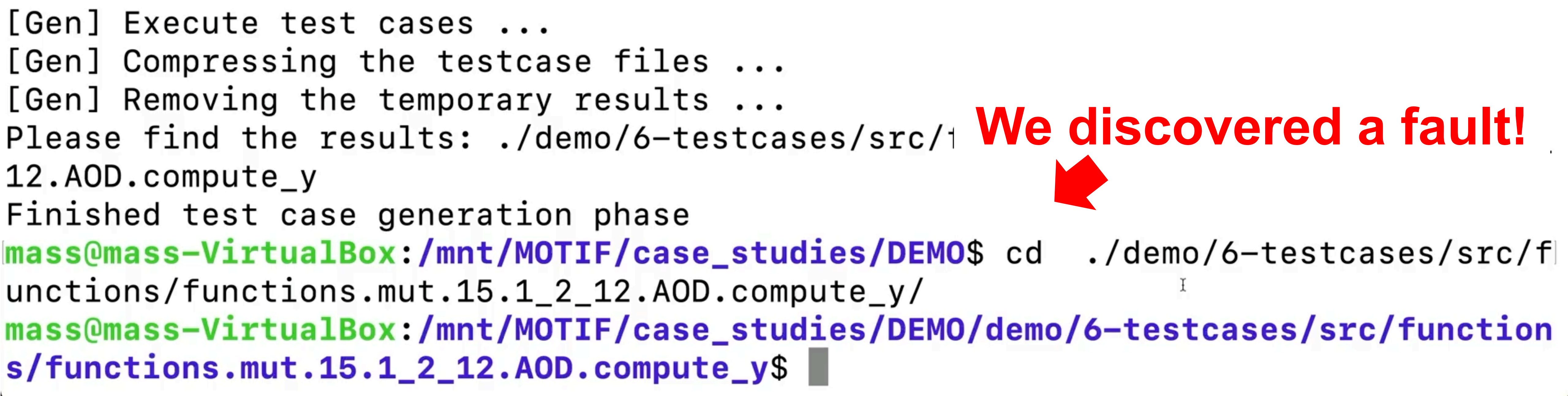
```
  -----Obtained compilation flags from SUT makefile -------------
  - flags                   : ['-Wextra', '-fPIC', '-Wall', '-fprofile-arcs', '-g'
, '-O2', '-ftest-coverage']
  - includes                : ['include']
  - dynamic libraries       : []
  - dynamic library paths: []
  - compiled objects        : ['functions.o', 'error.o']
  - compiled objects path: /mnt/DEMO/my-library/src
  - source files            : ['functions.c', 'error.c']
  - target source path      : /mnt/DEMO/my-library/src

[Transformer] updating MOTIF configuration file ...
[Transformer] generating the list of live mutants ...
List of mutants that are not found in the mutants.tar:
All the mutants exist in the mutants.tar.
Finished.
mass@mass-VirtualBox:/mnt/MOTIF$ cd case_studies/DEMO/
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO$ vi config.py
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO$ /mnt/MOTIF/motif.py -c confi
g.py -J demo --timeout 300 ./src/functions/functions.mut.15.1_2_12.AOD.compute_
y.c
```

- The new test case exercises the software with “corner case” inputs never tested before.

- We shall manually verify the correctness of results.

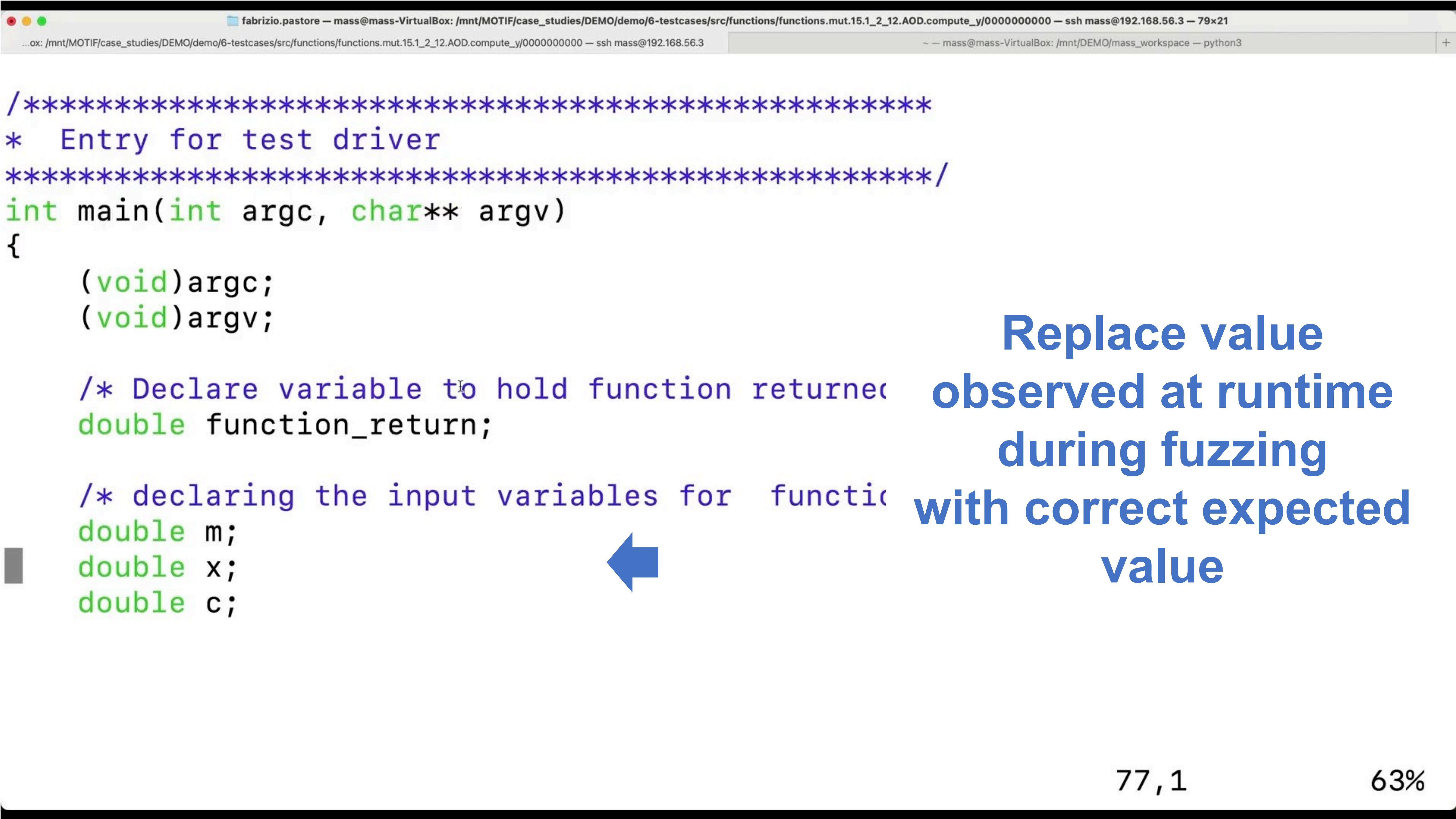
- Expected result based on specification docs:

$$m*x+c = (-1)*(-1)+(-1) = 0$$



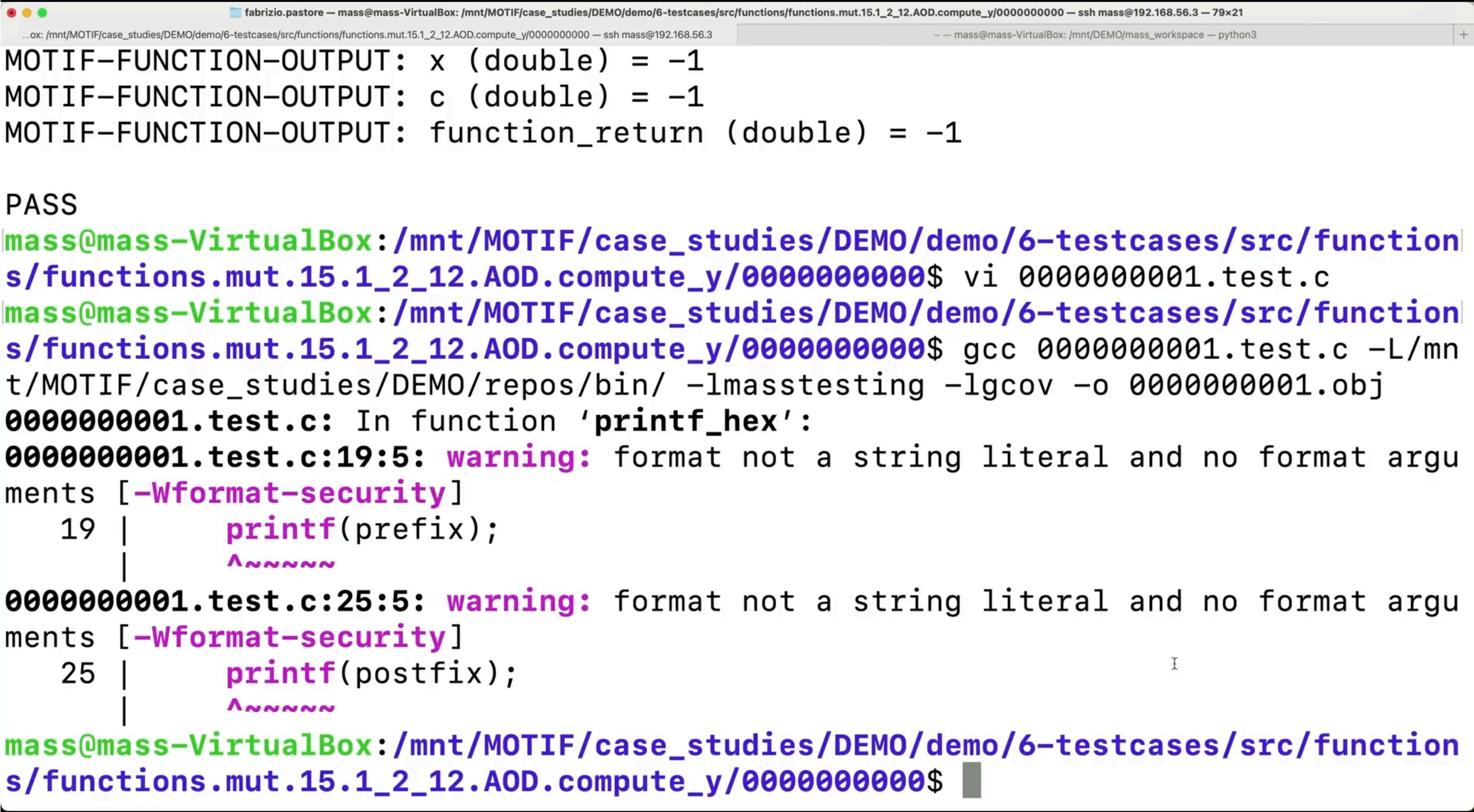
```
[Gen] Execute test cases ...
[Gen] Compressing the testcase files ...
[Gen] Removing the temporary results ...
Please find the results: ./demo/6-testcases/src/f
12.AOD.compute_y
Finished test case generation phase
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO$ cd  ./demo/6-testcases/src/f
unctions/functions.mut.15.1_2_12.AOD.compute_y/
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y$
```

We discovered a fault!

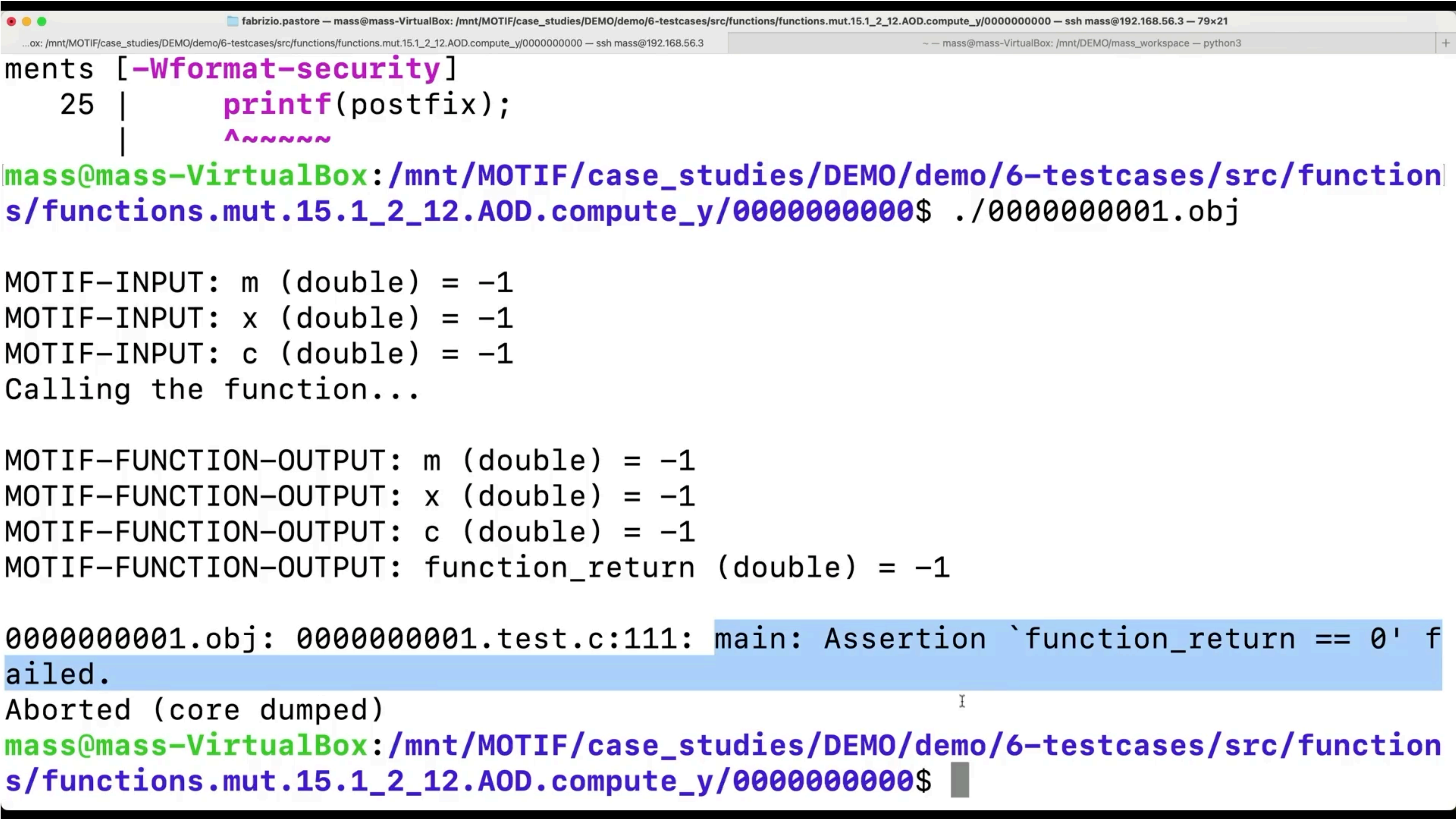

fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/functions/functions.mut.15.1_2_12.AOD.compute_y/0000000000 — ssh mass@192.168.56.3 — 79×21
...ox: /mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/functions/functions.mut.15.1_2_12.AOD.compute_y/0000000000 — ssh mass@192.168.56.3
~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
/**************************************************
*  Entry for test driver
**************************************************/
int main(int argc, char** argv)
{
    (void)argc;
    (void)argv;

    /* Declare variable to hold function returnec
    double function_return;

    /* declaring the input variables for  functic
    double m;
    double x;
    double c;

                                                    77,1          63%
```

Replace value observed at runtime during fuzzing with correct expected value


fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/functions/functions.mut.15.1_2_12.AOD.compute_y/0000000000 — ssh mass@192.168.56.3 — 79×21

```
MOTIF-FUNCTION-OUTPUT: x (double) = -1
MOTIF-FUNCTION-OUTPUT: c (double) = -1
MOTIF-FUNCTION-OUTPUT: function_return (double) = -1

PASS
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y/0000000000$ vi 0000000001.test.c
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y/0000000000$ gcc 0000000001.test.c -L/mn
t/MOTIF/case_studies/DEMO/repos/bin/ -lmasstesting -lgcov -o 0000000001.obj
0000000001.test.c: In function 'printf_hex':
0000000001.test.c:19:5: warning: format not a string literal and no format argu
ments [-Wformat-security]
   19 |     printf(prefix);
      |     ^~~~~~
0000000001.test.c:25:5: warning: format not a string literal and no format argu
ments [-Wformat-security]
   25 |     printf(postfix);
      |     ^~~~~~
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y/0000000000$
```

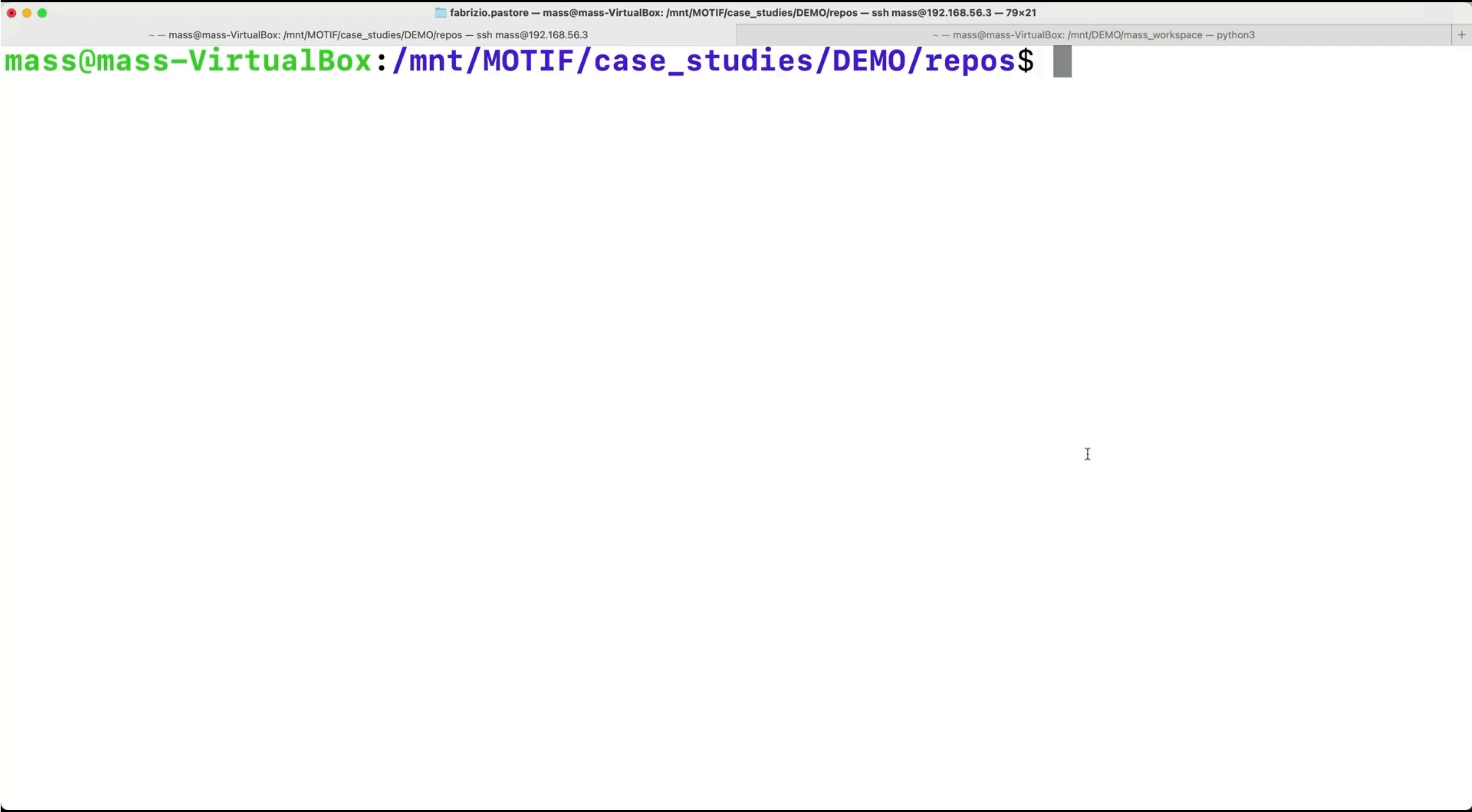


fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/functions/functions.mut.15.1_2_12.AOD.compute_y/0000000000 — ssh mass@192.168.56.3 — 79×21

...ox: /mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/functions/functions.mut.15.1_2_12.AOD.compute_y/0000000000 — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
ments [-Wformat-security]
   25 |     printf(postfix);
      |     ^~~~~~
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y/0000000000$ ./0000000001.obj

MOTIF-INPUT: m (double) = -1
MOTIF-INPUT: x (double) = -1
MOTIF-INPUT: c (double) = -1
Calling the function...

MOTIF-FUNCTION-OUTPUT: m (double) = -1
MOTIF-FUNCTION-OUTPUT: x (double) = -1
MOTIF-FUNCTION-OUTPUT: c (double) = -1
MOTIF-FUNCTION-OUTPUT: function_return (double) = -1

0000000001.obj: 0000000001.test.c:111: main: Assertion `function_return == 0' f
ailed.
Aborted (core dumped)
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/demo/6-testcases/src/function
s/functions.mut.15.1_2_12.AOD.compute_y/0000000000$ 
```



fabrizio.pastore — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO/repos — ssh mass@192.168.56.3 — 79×21

~ — mass@mass-VirtualBox: /mnt/MOTIF/case_studies/DEMO/repos — ssh mass@192.168.56.3

~ — mass@mass-VirtualBox: /mnt/DEMO/mass_workspace — python3

```
mass@mass-VirtualBox:/mnt/MOTIF/case_studies/DEMO/repos$ 
```

Methodology

Test adequacy determination guidelines

- Derived from the empirical assessment of the capability of detecting real faults of multiple test suites for **ESAIL**, **S5 L1PP**, and **BepiColombo**
- Test suites shall have a code-driven mutation score **above 70%** (65% for data-driven)
    - Only above those thresholds, test suites have **significantly higher fault detection rate** than test suites not derived with mutation analysis (i.e., lower mutation score)
- Poor test suites have a code-driven mutation score below 40% (20% data driven)
    - Below such threshold, they **do not detect any real-world fault**

Assessment

Empirical assessment

				Assessed Tools			
	Subject	**LOC**	**Test suite**	**MASS**	**MOTIF**	**DAMAT**	**DAMTEF**
LuxSpace	ESAIL-CSW	74,155	System + Unit	Yes	Yes	Yes	
	Z80 emulator	1999	Unit	Yes	Yes	Yes	
	Zynq 7000 SW	20	Unit	Yes			
GomSpace	LibGCSP	9,836	Integration	Yes	Yes		
	LibParam	3,179	Integration	Yes		Yes	Yes
	LibUtil	10,576	Unit	Yes	Yes		
ESA	MLFS	5,402	Unit	Yes	Yes		
	ASN1.CC	4,338	Unit	Yes	Yes		
Huld	BepiColombo SIXS/MISX	30,000	System + Unit	Yes	Yes	Yes	
	S5 L1bPP	23,000	Integration + Unit	Yes	Yes	Yes	
	ExoMars RSI	11,500	System + Unit	Yes			

Verification reports

Independent feedback

- Code-driven and data-driven mutation are complimentary
- Test suite pitfalls reported by code-driven and data-driven mutation are correct
- FAQAS2 tools enabled the detection of real faults
- Personnel training costs are non-negligible
- Require parallel execution for large software
- A small subset of mutants belonging to complex functions enables a quick application of the methodology (e.g., for ISVV)

FAQAS2: Fault-based Automated Quality Assurance Assessment for Space 2

IMPROVE MUTATION TESTING IN SPACE SOFTWARE SYSTEMS

http://faqas.uni.lu

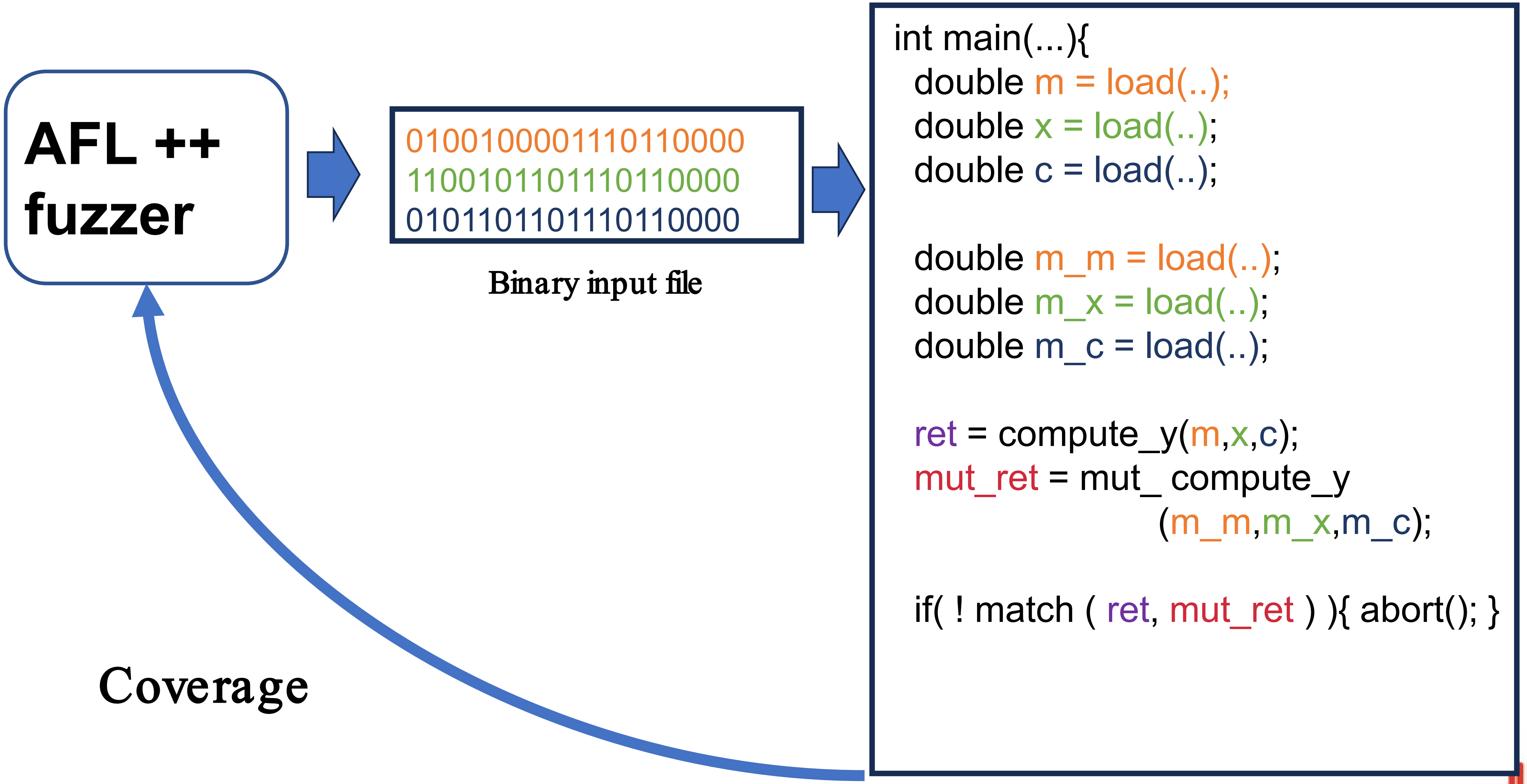
SnT

uni.lu UNIVERSITY OF LUXEMBOURG

Backup

How test cases are generated?

Test generation



```
01001000011110110000
11001011011110110000
01011011011110110000
```

```
int main(...){
  double m = load(..);
  double x = load(..);
  double c = load(..);

  double m_m = load(..);
  double m_x = load(..);
  double m_c = load(..);

  ret = compute_y(m,x,c);
  mut_ret = mut_ compute_y
                (m_m,m_x,m_c);

  if( ! match ( ret, mut_ret ) ){ abort(); }
```

Test generation

AFL ++ fuzzer

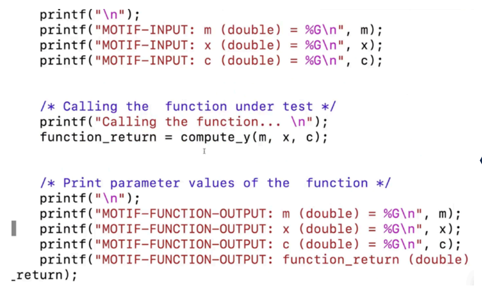
```
0100100001110110000
1100101101110110000
0101101101110110000
```

Binary input file

```
int main(...){
  double m = load(..);
  double x = load(..);
  double c = load(..);

  double m_m = load(..);
  double m_x = load(..);
  double m_c = load(..);

  ret = compute_y(m,x,c);
  mut_ret = mut_ compute_y
                (m_m,m_x,m_c);

  if( ! match ( ret, mut_ret ) ){ abort(); }
```

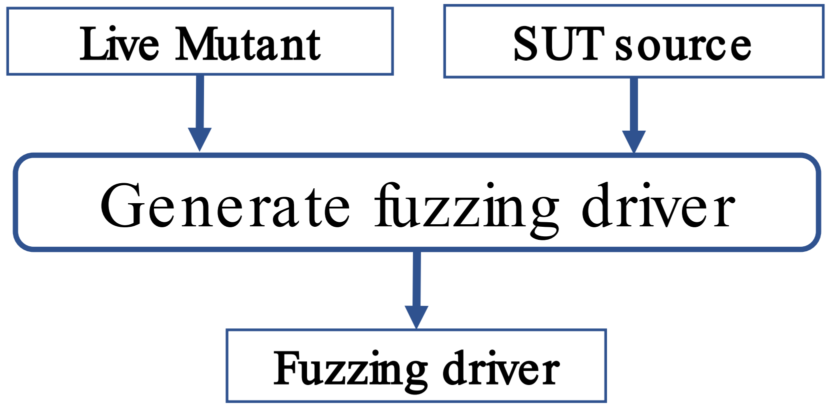
Post-processing

Test case



```
    printf("\n");
    printf("MOTIF-INPUT: m (double) = %G\n", m);
    printf("MOTIF-INPUT: x (double) = %G\n", x);
    printf("MOTIF-INPUT: c (double) = %G\n", c);


    /* Calling the  function under test */
    printf("Calling the function... \n");
    function_return = compute_y(m, x, c);


    /* Print parameter values of the  function */
    printf("\n");
    printf("MOTIF-FUNCTION-OUTPUT: m (double) = %G\n", m);
    printf("MOTIF-FUNCTION-OUTPUT: x (double) = %G\n", x);
    printf("MOTIF-FUNCTION-OUTPUT: c (double) = %G\n", c);
    printf("MOTIF-FUNCTION-OUTPUT: function_return (double)
_return);
```

The MOTIF process

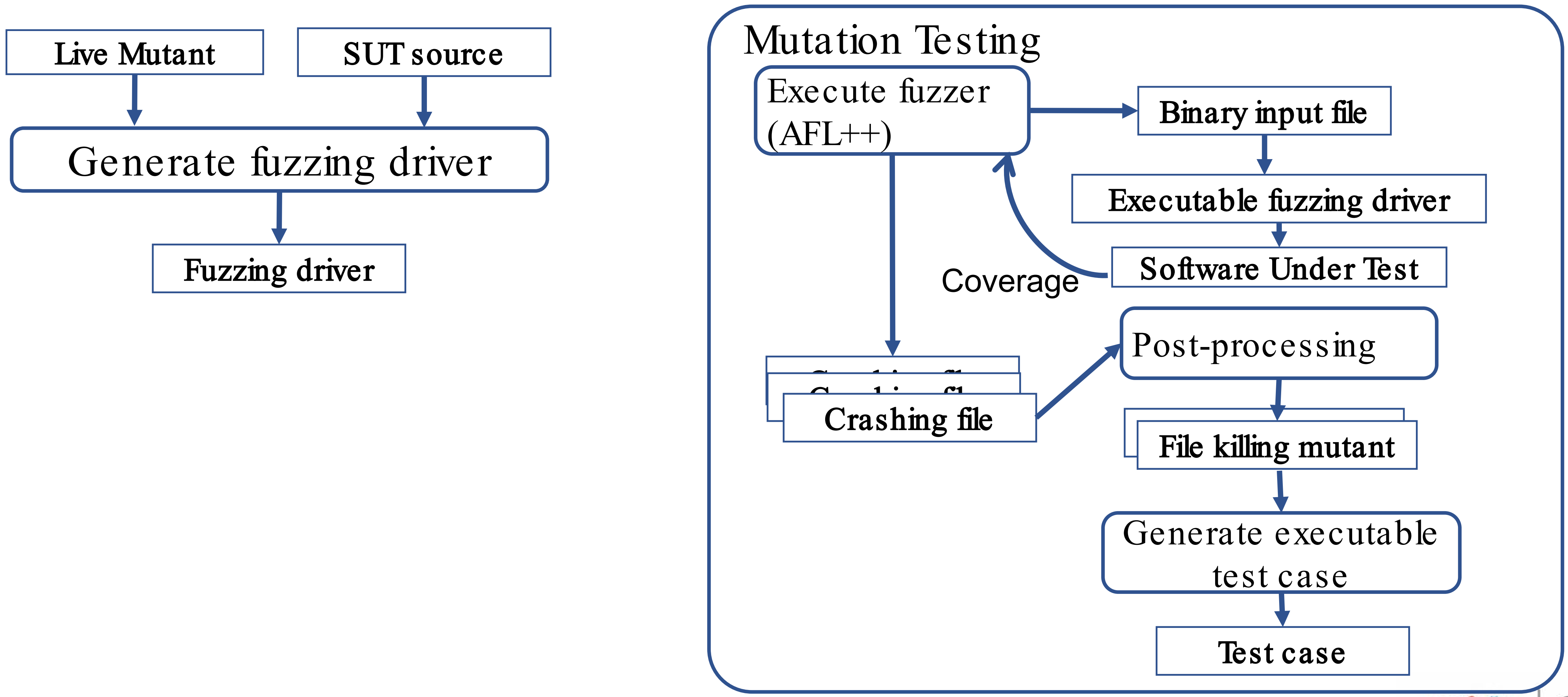


```
01001000011101100000
11001011011101100000
01011011011101100000
```

Binary input file

```
int main(...){
  double m = load(..);
  double x = load(..);
  double c = load(..);

  double m_m = load(..);
  double m_x = load(..);
  double m_c = load(..);

  ret = compute_y(m,x,c);
  mut_ret = mut_ compute_y
                  (m_m,m_x,m_c);

  if( ! match ( ret, mut_ret ) ){ abort(); }
  if( ! match (m, m_m ){ abort() };
  if( ! match (x, m_x) { abort() };
  if( ! match (c, m_c) { abort() };
```

The MOTIF process


Live Mutant
SUT source
Generate fuzzing driver
Fuzzing driver
Mutation Testing
Execute fuzzer (AFL++)
Binary input file
Executable fuzzing driver
Software Under Test
Coverage
Crashing file
Post-processing
File killing mutant
Generate executable test case
Test case