# Think-piece

**You are given a library, container.o and the following .h file signatures:**

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c,
  otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after
  return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

**You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something.**

***A bug! Missing quotation mark!***

# Think-piece

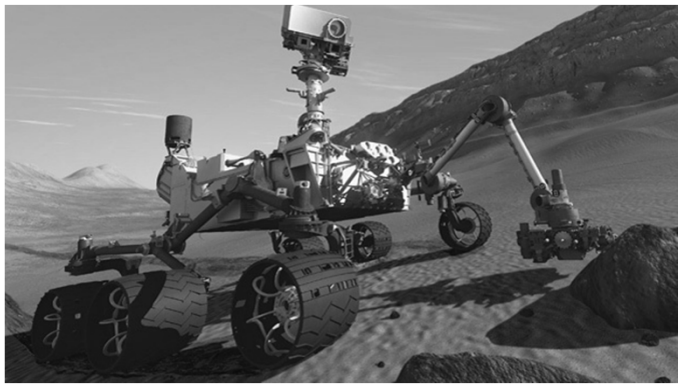
**You are given a library, container.o and the following .h file signatures:**

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c, otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

**You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something."**

# What Do I Know About Testing?

- Who am I?
  - Alex Groce (you can look me up online)
- What do I know about testing?
  - Before coming to the university, I was lead for design/development for software test automation on a project at NASA…
    - **JPL's Curiosity Rover**
- These days I look for (and find) bugs in JavaScript engines and compilers



# Think-piece

**You are given a library, container.o and the following .h file signatures:**

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c, otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

**You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something."**

# What Matters Here?

- Simplify the problem
- What we have:

```
int put (int n, container* c);
  returns 1 and adds to c if n not in c, otherwise returns 0

int get (int n, container* c);
  returns 1 if n is in c, 0 otherwise

int remove (int n, container* c);
  returns 1 if n was in c; after return n is not in c!

container* newContainer();
  returns a new container if memory avail
```

# The API (Interface) We're Testing

- What is the goal of testing?

- We want to see if this code does what it says it does
- How can we do that?

```
int put (int n, container* c);
   returns 1 and adds to c if n not in c, otherwise
   returns 0

int get (int n, container* c);
   returns 1 if n is in c, 0 otherwise

int remove (int n, container* c);
   returns 1 if n was in c; after return n is not in
   c!

container* newContainer();
   returns a new container if memory avail
```

# A Very Simple Test

- This is a typical manual unit test
  - Do something to the software that has a known result
  - *Assert* the result matches what you expect
- How much does this test?

```
c = newContainer();

r = put(0,c);
assert (r == 1);

r = put(0,c);
assert (r == 0);

r = get(0,c);
assert (r == 1);

r = remove(0,c);
assert (r == 1);

r = get(0,c);
assert (r == 0);
```

# Manual Unit Tests

- How much does this test?
  - Probably not very much
  - That's ok, we can write more tests…
    - and more tests…
      - and still more tests…
  - How do we know we're done?

```
c = newContainer();

r = put(0,c);
assert (r == 1);

r = put(0,c);
assert (r == 0);

r = get(0,c);
assert (r == 1);

r = remove(0,c);
assert (r == 1);

r = get(0,c);
assert (r == 0);
```

# Boredom Sets in Quickly

- Writing each sequence of operations we want to try is tedious
  - We’re going to run out of patience before we try very many things
  - Each test takes a long time to write
  - Could we get the computer to do it for us?

# Random Testing

- Here's an attempt:

```
c = newContainer();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = put(v,c);
 if (op == 1)
   r1 = get(v,c);
 if (op == 2)
   r1 = remove(v,c);
}
```

# Random Testing

- What kind of bugs can this testing find?

```
c = newContainer();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = put(v,c);
 if (op == 1)
   r1 = get(v,c);
 if (op == 2)
   r1 = remove(v,c);
}
```

# Differential Testing

- What does the container program act like?
  - A set
  - Do we have any other set implementations?
  - Could we write one that
    - we are pretty sure is correct
    - acts like our tested system is supposed to act?

# Differential Testing

```
c = newContainer();
ref = newBinaryTree();
for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0) {
   r1 = put(v,c);
   r2 = bt_put(v,ref);
 } else if (op == 1) {
   r1 = get(v,c);
   r2 = bt_get(v,ref);
 } else if (op == 2) {
   r1 = remove(v, c);
   r2 = bt_remove(v, ref);
 }
 assert (r1 == r2);
}
```