Think-piece

You are given a class, Container and the following methods signatures:

```
public Container();/* a constructor returns Container object*/
public int put(int n)/* returns 1 and adds n to the Container if n not
  in the Container, otherwise returns 0*/
public int get(int n);/* returns 1 if n is in the Container, 0
  otherwise */
Public int remove(int n); /* returns 1 if n was in the Container;
  after return n is not in the Container! */
public int size()/* returns the number of elements in this Container.*/
```

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. 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.

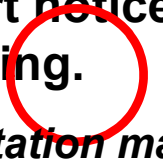
Think-piece

You are given a class, Container and the following methods signatures:

```
public Container();/* a constructor returns Container object*/
public int put(int n)/* returns 1 and adds n to the Container if n not
  in the Container, otherwise returns 0*/
public int get(int n);/* returns 1 if n is in the Container, 0
  otherwise */
Public int remove(int n); /* returns 1 if n was in the Container;
  after return n is not in the Container! */
public int size()/* returns the number of elements in this Container.*/
```

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. 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 class, Container and the following methods signatures:

```
public Container();/* a constructor returns Container object*/
public int put(int n)/* returns 1 and adds n to the Container if n not
  in the Container, otherwise returns 0*/
public int get(int n);/* returns 1 if n is in the Container, 0
  otherwise */
Public int remove(int n); /* returns 1 if n was in the Container;
  after return n is not in the Container! */
public int size()/* returns the number of elements in this Container.*/
```

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. 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:

```
public Container();
       returns Container object

public int put(int n);
      returns 1 and adds n to the Container if n not in the Container,
  otherwise returns 0

public int get(int n);
        returns 1 if n is in the Container, 0 otherwise

Public int remove(int n);
        returns 1 if n was in the Container; after return n is not in
  the Container!
public int size()
        returns the number of elements in this Container.
```

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?

```
public Container();
        returns Container object

public int put(int n);
      returns 1 and adds to the Container if n not in the
Container, otherwise returns 0

public int get(int n);
        returns 1 if n is in the Container, 0 otherwise

Public int remove(int n);
        returns 1 if n was in the Container; after return
n is not in the Container!

public int size()
        returns the number of elements in this Container.
```

Test Data Generation

- Given a function/method to test, how do we derive/generate inputs to the function/method that test its beahviour?

1. Manual Testing (**Think**)
2. Random Testing (**Guess**)
3. Search-based Testing (**Search**)
4. Symbolic execution (**Deduce**)

Manual Unit Tests (Think)

- Manual testing includes testing a software manually, i.e., without using any automated tool or any script.
- The tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug.
- There are different stages for manual testing such as unit testing, integration testing, system testing, and user acceptance testing.

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 = new Container();

 r = c.put(0);
assertEquals(1, r);

 r = c.put(0);
 assertEquals(0, r);

 r = c.get(0);
assertEquals(1, r);

 r = c.remove(0);
assertEquals(1, r);

 r = c.get(0);
 assertEquals(0, r);
```

Manual Unit Tests (Think)

- 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 = new Container();

r = c.put(0);
assertEquals(1, r);

r = c.put(0);
assertEquals(0, r);

r = c.get(0);
assertEquals(1, r);

r = c.remove(0);
assertEquals(1, r);

r = c.get(0);
assertEquals(0, r);
```

Manual Unit Tests (Think)

- 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 (Guess)

- Randomly generate inputs to feed in a software.
- (Totally) uninformed search.
- Do not require any preparation & easy to implement
- Works pretty well in many cases
- Problems
  - Semantically redundant inputs
  - E.g., for a simple program 10/x, providing any input except 0 means the same

Random Testing (Guess)

- Here's an attempt:

```
NUM_TESTS=100;
MAX_VALUE=10;
c = new Container();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(4);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = c.put(v);
 if (op == 1)
   r1 = c.get(v);
 if (op == 2)
   r1 = c.remove(v);
 if (op == 3)
   r1 = c.size();
}
```

Random Testing (Guess)

- What kind of bugs can this testing find?

```
NUM_TESTS=100;
MAX_VALUE=10;

c = new Container();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(4);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = c.put(v);
 if (op == 1)
   r1 = c.get(v);
 if (op == 2)
   r1 = c.remove(v);
 if (op == 3)
   r1 = c.size();
}
```

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

- 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?

any idea!

Differential Testing

```
c = new Container();
ref = new HashSet();
for (int i = 0; i < NUM_TESTS; i++) {
 op = random(4);
 v = random(MAX_VALUE);
 if (op == 0) {
   r1 = c.put(v);
   r2 = ref.add(v)?1:0;
 } else if (op == 1) {
   r1 = c.get(v);
   r2 = ref.contains(v)?1:0;
 } else if (op == 2) {
   r1 = c.remove(v);
   r2 = ref.remove(v)?1:0;
 } else if (op == 3) {
   r1 = c.size();
   r2 = ref.size();
 }
  assert (r1 == r2);
}
```

Search-based Software Testing (SBST)

- Search Based Software Testing (**SBST**) is a branch of Search Based Software Engineering (**SBSE**).
- **SBST** is the process of generating the inputs of test cases (i.e., test cases) using search-based optimisation algorithms, guided by a fitness function that captures the current test objective.
- The **fitness function** is used to guide a search-based optimisation algorithm, which searches the space of test inputs to find those that meet the test objectives.
- There are many different search-based optimisation algorithms, such as Genetic algorithm (GA), and Hill Climbing (HC).
- **Problems**
  - Might not always find the best solution.
  - The **effectiveness** of the search algorithms is improved as long as the **fitness function** distinguishes between better and worse solutions.

Symbolic execution (deduce)

- Symbolic execution is a program analysis technique that analyzes a program's code to automatically generate test data for the program.
- Symbolic execution analyzes the code structure to find out a path to go to a certain statement.
- Symbolic execution analyzes the code structure to find out the constraint of the inputs to let the program follow the path.
- Symbolic execution uses **constraint solver** to provide a value set that satisfies the constraint.
- **Problems:**
  - **Path explosion.**
    - It is difficult to symbolically execute a significantly large subset of all program paths because most real world software have an extremely large number of paths.
  - **Too complex constraint.**
    - It may not always be possible to solve path constraints because solving the general class of constraints is undecidable.

References:

http://classes.engr.oregonstate.edu/eecs/summer2015/cs362-002/

Anand, Saswat, et al. "An orchestrated survey of methodologies for automated software test case generation." Journal of Systems and Software 86.8 (2013): 1978-2001.