Brielan Beamon: beamonbh@g.cofc.edu
Logan Smith: smithla1@g.cofc.edu
Scott White: whites2@g.cofc.edu

Dr. Bowring
CSCI 362
November 29, 2016

Automated Testing Framework

Introduction

Our project is centered around Sugar Labs, a free and open source software project that focuses on allowing people around the world to learn more about the world around them for free. Sugar Labs is a spin off of One Laptop Per Child, constituting a simple Linux-based operating system. Due to the complexity of emulating and then testing an entire operating system, we have opted to test individual components of Sugar Labs — which are referred to as activities.

Faults

For this deliverable, we inserted five faults into our source code. The goal of this was to make some, but hopefully not all, of our test cases fail. In this, we succeeded. For the following five faults, we say what file the fault is in, what line(s) the fault is at, and what effect the fault has.

1st Fault:
TestAutomation/project/sugar-calculate/functions.py
Line 351
Replace

```
return long(x) ** int(y)
```

With

```
return long(x) * int(y)
```

With this fault, we made it so that some test cases would fail depending on whether or not they reached this line inside a conditional statement. Specifically, this will occur if the first argument and second arguments off the add function are integers.

2nd Fault:
TestAutomation/project/sugar-calculate/functions.py
Line 101
Replace

```
angle_scaling = ClassValue(1.0)
```

With

```
angle_scaling = ClassValue(2.0)
```

We added a fault to the variable angle_scaling in functions.py. This variable is assigned a value of 1.0 normally and it is used in the calculations of sine, cosine, and tangent values. Changing this value cased the calculation of our cosine values to be incorrect.

3rd Fault:
TestAutomation/project/sugar-calculate/rational.py
Lines 34 and 35
Replace

```
if n is not None:
        self.set(n, d)
```

With

```
 if n is None:
        self.set(n, d)
```

We added a fault to the rational class so that it would skip some of the steps it takes upon initialization based on an erroneous boolean expression.

4th Fault:
TestAutomation/project/sugar-calculate/functions.py
Line 413
Replace

```
return math.sqrt(float(x))
```

With

```
return math.sqrt(int(x))
```

This fault results in a loss of precision when using the square root function. Thus non-integers (that are not equivalent to whole numbers) will not calculate correctly as precision is lost in the conversion.

5th Fault:
TestAutomation/project/sugar-calculate/functions.py
Lines 132-135
Replace

```
if isinstance(x, _Decimal) or isinstance(y, _Decimal):
x = _d(x)
y = _d(y)
 return x + y
```

With

```
if isinstance(x, _Decimal) or isinstance(y, _Decimal):
x = _d(x)
y = _d(y)
 return x + y
```

This fault results in the addition being performed only if the parameters for the function are instances of a decimal object.