Chapter 9: Syntax-Based Testing

Name: Devansh Amin

Complete the problems below and submit this word document with your answers.

1. Consider the following BNF:

```
Y ::= W S F | S M F | S M
W ::= "d" | "j" | "f"
S ::= "m" | "a" | "y"
M ::= "y" | "n" | "u" | "g"
F ::= "p" | "o" | "n"
```

   a) Write two strings that are valid according to the BNF.
   b) For each of your two strings, give two valid mutants of the string.
   c) For each of your two strings, give two invalid mutants of the string.

Solution:

a) dmp, myp

b) Valid Mutants

   dmp -> jmp, fmp

   myp -> ayp, mnp

c) Invalid Mutants

   dmp -> dma, djm

   myp -> myn, myg

2. Answer questions (a) through (d) for the mutant on line 6 in the method **`sum()`**.

```
/**
 * Sum values in an array
 *
 * @param x array to sum
 *
 * @return sum of values in x
 * @throws NullPointerException if x is null
 */
1. public static int sum(int[] x)
2. {
3.    int s = 0;
4.    for (int i=0; i < x.length; i++) }
5.    {
6.       s = s + x[i];
Δ6'.   // s = s - x[i]; //AOR
7.    }
8.    return s;
9.  }
```

a) If possible find test inputs that do not reach the mutant.
b) If possible, find test inputs that satisfy reachability but not infection for the mutant.
c) If possible, find test inputs that satisfy reachability and infection, but not propagation for the mutant.
d) If possible, find test inputs that **strongly** kill the mutants.

Solution:

a) If x is null or empty array, x = null or x = [], then the mutant is never reached.
b) Any input with all zeros will reach but not infect. x = [0] or x = [0, 0]
c) Any input with nonzero entries, but with a sum of zero is fine. x = [1, -1] or x = [1, -3, 2]
d) Any input with a nonzero sum works. x = [4, 5, 6]

Extra credit:

Answer the following questions for the following XML schema for CCSU CS course codes (You can visit the following page at the link below to see the list of available CS course codes (enter CS into the search box and hit search button)).
https://www2.ccsu.edu/course/

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:simpleType name='coursecode'>
    <xsd:restriction base='xsd:string'>
      <xsd:pattern value='Type your answer here'/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

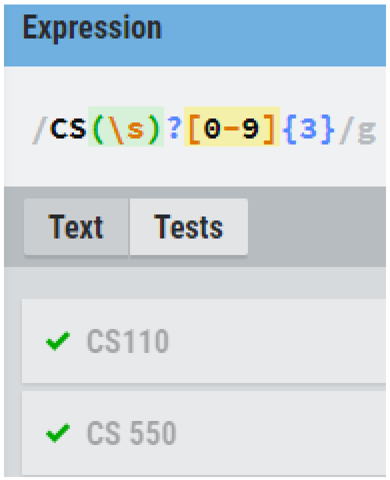
a) Come up with a pattern accepting both undergraduate and graduate level courses (e.g., CS 418 Principles of Software Testing and Quality Assurance and CS 506 Software Testing and Quality Assurance) (You can use a website like this: https://regexr.com/ to test your pattern accuracy)

b) Write 2 test strings of different lengths that satisfy the schema
c) Give 2 mutants
d) Change the schema's pattern to only allow undergraduate CS course codes (you can assume the undergraduate course codes begin with CS 110 and end with CS 499).
e) Change the schema's pattern to only allow graduate CS course codes (i.e., you can assume the graduate course codes begin with CS 500 and end with CS 599).

Solution:

a) `<xsd:pattern value = 'CS(\s)?[0-9]{3}' />`

b) CS110, CS 550



Expression

```
/CS(\s)?[0-9]{3}/g
```

✓ CS110

✓ CS 550

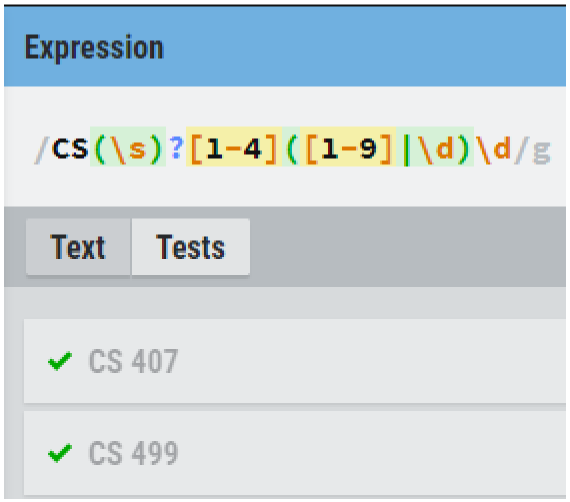
c) Mutants

`<xsd:pattern value = 'CS(\s)?[0-8]{3}' />`

`<xsd:pattern value = 'CS(\s)?[0-9]{4}' />`

d) Undergraduate

`<xsd:pattern value = 'CS(\s)?[1-4]([1-9]|\d)\d' />`



Expression

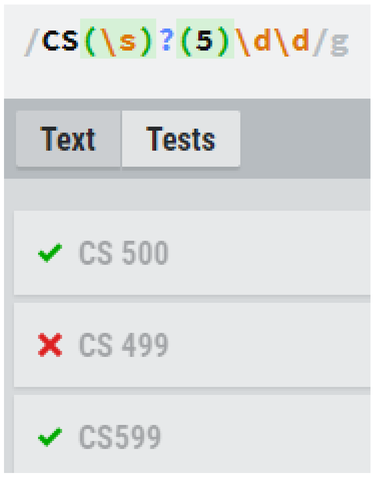
```
/CS(\s)?[1-4]([1-9]|\d)\d/g
```

✓ CS 407

✓ CS 499

e) Graduate

`<xsd:pattern value = 'CS(\s)?(5)\d\d' />`



```
/CS(\s)?(5)\d\d/g
```

✓ CS 500

✗ CS 499

✓ CS599