



Software Testing Laboratory
(CS6474)
Assignment 05 :TCASES TOOL

Tapas Manna
223CS3152
Master of Technology
223cs3152@nitrkl.ac.in

Department of Computer Science Engineering
NIT, Rourkela
February 08, 2024

Contents

1	Annotations-Test	3
1.1	Specification	3
1.2	XML Code	3
1.3	Test cases screenshot	4
2	Find-Input	5
2.1	Specification	5
2.2	XML Code	5
2.3	Test cases screenshot	6
3	Tcases-Input	6
3.1	Specification	6
3.2	XML Code	8
3.3	Test cases screenshot	13
4	Ice-Cream Input	13
4.1	Specification	13
4.2	XML Code	13
4.3	Test cases screenshot	15

1 Annotations-Test

1.1 Specification

System: Examples Usage: ADD or Create different geometrical shape We can add shape by using addShape function by passing parameter to 1) Type: CIRCLE RECTANGLE,LINE etc 2) Size: In cm 3) Colour: RED ,GREEN etc Function: addShape

1.2 XML Code

```
1 <!--
2   System: Examples
3   Usage: ADD or Create different geometrical shape
4   We can add shape by using addShape function by passing parameter to
5   1) Type: CIRCLE RECTANGLE,LINE etc
6   2) Size: In cm
7   3) Colour: RED ,GREEN etc
8   Function: addShape
9 -->
10
11 <System name="Examples">
12
13   <Function name="addShape">
14     <!-- Test case annotations -->
15     <Has name="pageType" value="Page"/>
16     <Has name="pageName" value="page"/>
17     <Has name="pageValue" value="new Page()"/>
18
19     <Input>
20
21       <Var name="Type">
22         <!-- Variable binding annotations -->
23         <Has name="varType" value="Shape"/>
24         <Has name="varName" value="shape"/>
25         <Has name="varEval" value="new Shape"/>
26
27         <Value name="SQUARE"/>
28         <Value name="CIRCLE"/>
29         <Value name="LINE" property="1D"/>
30       </Var>
31
32       <Var name="Size">
33         <!-- Variable binding annotations -->
34         <Has name="varType" value="int"/>
35         <Has name="varName" value="size"/>
36         <Has name="varApply" value="setSize"/>
37
38         <Value name="1"/>
39         <Value name="10"/>
40         <Value name="100" property="Large"/>
41       </Var>
42
43       <Var name="Color">
44         <!-- Variable binding annotations -->
45         <Has name="varType" value="String"/>
46         <Has name="varName" value="color"/>
47         <Has name="varApply" value="setColor"/>
```

```
48         <Value name="red"/>
49         <Value name="green"/>
50         <Value name="blue"/>
51     </Var>
52 </Input>
53 </Function>
54 </System>
```

1.3 Test cases screenshot

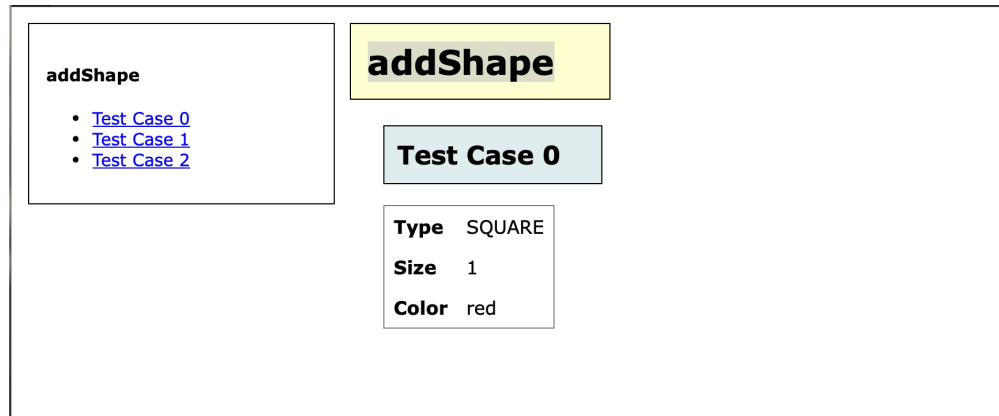


Figure 1: TCases Html View

2 Find-Input

2.1 Specification

Usage: find pattern file

Locates one or more instances of a given pattern in a text file.

All lines in the file that contain the pattern are written to standard output. A line containing the pattern is written only once, regardless of the number of times the pattern occurs in it.

The pattern is any sequence of characters whose length does not exceed the maximum length of a line in the file. To include a blank in the pattern, the entire pattern must be enclosed in quotes ("). To include a quotation mark in the pattern, two quotes (") must be used.

2.2 XML Code

```
1 <System name="Examples">
2   <Function name="find">
3     <!--
4       Usage: find pattern file
5
6       Locates one or more instances of a given pattern in a text file.
7
8       All lines in the file that contain the pattern are written to standard
9       output. A
10      line containing the pattern is written only once, regardless of the
11      number of
12      times the pattern occurs in it.
13
14      The pattern is any sequence of characters whose length does not exceed
15      the
16      maximum length of a line in the file. To include a blank in the pattern,
17      the
18      entire pattern must be enclosed in quotes ("). To include a quotation
19      mark in the
20      pattern, two quotes in a row (") must be used.
21    -->
22    <Input type="arg">
23      <VarSet name="pattern" when="fileExists">
24        <Var name="size">
25          <Value name="empty" property="empty"/>
26          <Value name="singleChar" property="singleChar"/>
27          <Value name="manyChars"/>
28        </Var>
29        <Var name="quoted">
30          <Value name="yes" property="quoted"/>
31          <Value name="no" whenNot="empty"/>
32          <Value name="unterminated" failure="true"/>
33        </Var>
34        <Var name="blanks" whenNot="empty">
35          <Value name="none"/>
36          <Value name="one" when="quoted, singleChar"/>
37          <Value name="many">
38            <When>
39              <AllOf property="quoted">
40                <Not property="singleChar"/>
41              </AllOf>
42            </When>
43          </Value>
44        </Var>
45      </VarSet>
46    </Input>
47  </Function>
48 </System>
```

```

38         </Value>
39     </Var>
40     <Var name="embeddedQuotes" whenNot="empty, singleChar">
41         <Value name="none"/>
42         <Value name="one"/>
43         <Value name="many" once="true"/>
44     </Var>
45 </VarSet>
46
47 <Var name="fileName">
48     <Value name="defined" property="fileName"/>
49     <Value name="missing" failure="true"/>
50 </Var>
51 </Input>
52
53 <Input type="env">
54     <VarSet name="file" when="fileName">
55         <Var name="exists">
56             <Value name="yes" property="fileExists"/>
57             <Value name="no" failure="true"/>
58         </Var>
59         <VarSet name="contents" when="fileExists" whenNot="empty">
60             <Var name="linesLongerThanPattern">
61                 <Value name="one" property="matchable" once="true"/>
62                 <Value name="many" property="matchable"/>
63                 <Value name="none" failure="true"/>
64             </Var>
65             <Var name="patterns" when="matchable" whenNot="empty">
66                 <Value name="none" once="true"/>
67                 <Value name="one" property="match"/>
68                 <Value name="many" property="match, many"/>
69             </Var>
70             <Var name="patternsInLine" when="match">
71                 <Value name="one"/>
72                 <Value name="many" once="true" when="many"/>
73             </Var>
74         </VarSet>
75     </VarSet>
76 </Input>
77
78 </Function>
79 </System>

```

2.3 Test cases screenshot

3 Tcases-Input

3.1 Specification

System: Tcases

Function: run

Specification: This function is responsible for running test cases with various options.

Inputs: - defaultTupleSize: Specifies the default tuple size for test cases. - Yes: Indicates that a default tuple size is defined. - No: Indicates that no default tuple size is defined. - isNumber: Indicates whether the default tuple size is a number or not. Failure occurs if it's not a number.

- outFile: Specifies the output file option. - Yes: Indicates that an output file is defined. - No: Indicates that no output file is defined. - TransformOutputUndefined: Failure occurs if transformed

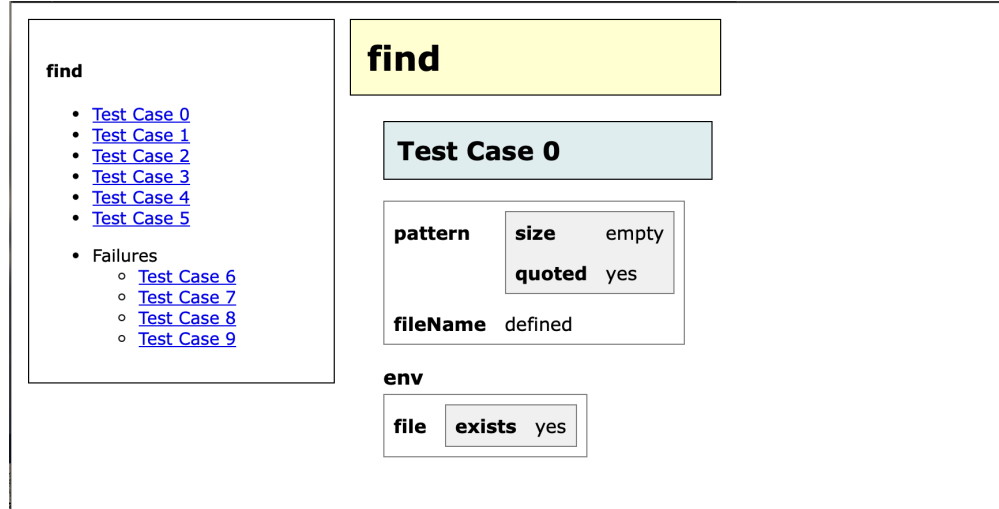


Figure 2: TCases Html View

output is undefined and the test file exists.

- genFile: Specifies the generated file option. - Yes: Indicates that a generated file is defined. - No: Indicates that no generated file is defined. - path: Specifies the path of the generated file. - isAbsolute: Indicates whether the path is absolute or not. - exists: Indicates whether the generated file exists or not.

- isJUnit: Specifies whether the output format is JUnit or not. - Yes: Indicates the output format is JUnit. It also marks transformed output if transform is not specified. - No: Indicates the output format is not JUnit if transform is specified. - NotAllowed: Failure occurs if transform is specified and isJUnit is not allowed.

- newTests: Specifies whether new tests are defined or not. - Yes: Indicates new tests are defined when the test file exists. - No: Indicates no new tests are defined.

- outDir: Specifies the output directory option. - Yes: Indicates that an output directory is defined. - No: Indicates that no output directory is defined. - path: Specifies the path of the output directory. - exists: Indicates whether the output directory exists or not. - isDirectory: Indicates whether the path is a directory or not. Failure occurs if it's not a directory.

- transformParams: Specifies the transformation parameters option. - count: Specifies the count of transformation parameters. - assignsValue: Indicates whether the transformation parameter assigns a value or not. Failure occurs if it doesn't assign a value. - nameDefined: Indicates whether the name of the transformation parameter is defined or not. Failure occurs if it's not defined. - valueDefined: Indicates whether the value of the transformation parameter is defined or not.

- seed: Specifies the seed option for generating random values. - Yes: Indicates that a seed is defined for generating random values. - No: Indicates that no seed is defined. - isNumber: Indicates whether the seed is a number or not. Failure occurs if it's not a number.

- testFile: Specifies the test file option. - Yes: Indicates that a test file is defined. - No: Indicates that no test file is defined. - path: Specifies the path of the test file. - isAbsolute: Indicates whether the path is absolute or not. - exists: Indicates whether the test file exists or not.

- transform: Specifies the transformation option. - Yes: Indicates that a transformation is defined. - No: Indicates that no transformation is defined. - path: Specifies the path of the transformation. - isAbsolute: Indicates whether the path is absolute or not. - exists: Indicates whether the transformation exists or not.

- inFile: Specifies the input definition file option. - Yes: Indicates that an input definition file is defined. - No: Indicates that no input definition file is defined. - path: Specifies the path of the input definition file. - isAbsolute: Indicates whether the path is absolute or not. - exists: Indicates whether the input definition file exists or not.

—i

3.2 XML Code

```

1 <!--
2   System: Tcases
3
4   Function: run
5
6   Specification:
7   This function is responsible for running test cases with various options.
8
9   Inputs:
10  - defaultTupleSize: Specifies the default tuple size for test cases.
11    - Yes: Indicates that a default tuple size is defined.
12    - No: Indicates that no default tuple size is defined.
13    - isNumber: Indicates whether the default tuple size is a number or not.
14    Failure occurs if it's not a number.
15
16  - outFile: Specifies the output file option.
17    - Yes: Indicates that an output file is defined.
18    - No: Indicates that no output file is defined.
19    - TransformOutputUndefined: Failure occurs if transformed output is
20      undefined and the test file exists.
21
22  - genFile: Specifies the generated file option.
23    - Yes: Indicates that a generated file is defined.
24    - No: Indicates that no generated file is defined.
25    - path: Specifies the path of the generated file.
26    - isAbsolute: Indicates whether the path is absolute or not.
27    - exists: Indicates whether the generated file exists or not.
28
29  - isJUnit: Specifies whether the output format is JUnit or not.
30    - Yes: Indicates the output format is JUnit. It also marks transformed
31      output if transform is not specified.
32    - No: Indicates the output format is not JUnit if transform is specified.
33    - NotAllowed: Failure occurs if transform is specified and isJUnit is not
34      allowed.
35
36  - newTests: Specifies whether new tests are defined or not.
37    - Yes: Indicates new tests are defined when the test file exists.
38    - No: Indicates no new tests are defined.
39
40  - outDir: Specifies the output directory option.
41    - Yes: Indicates that an output directory is defined.
42    - No: Indicates that no output directory is defined.
43    - path: Specifies the path of the output directory.
44    - exists: Indicates whether the output directory exists or not.
45    - isDirectory: Indicates whether the path is a directory or not. Failure
46      occurs if it's not a directory.
47
48  - transformParams: Specifies the transformation parameters option.
49    - count: Specifies the count of transformation parameters.

```



```

45     - assignsValue: Indicates whether the transformation parameter assigns a
value or not. Failure occurs if it doesn't assign a value.
46     - nameDefined: Indicates whether the name of the transformation parameter
is defined or not. Failure occurs if it's not defined.
47     - valueDefined: Indicates whether the value of the transformation
parameter is defined or not.
48
49     - seed: Specifies the seed option for generating random values.
50         - Yes: Indicates that a seed is defined for generating random values.
51         - No: Indicates that no seed is defined.
52         - isNumber: Indicates whether the seed is a number or not. Failure occurs
if it's not a number.
53
54     - testFile: Specifies the test file option.
55         - Yes: Indicates that a test file is defined.
56         - No: Indicates that no test file is defined.
57         - path: Specifies the path of the test file.
58         - isAbsolute: Indicates whether the path is absolute or not.
59         - exists: Indicates whether the test file exists or not.
60
61     - transform: Specifies the transformation option.
62         - Yes: Indicates that a transformation is defined.
63         - No: Indicates that no transformation is defined.
64         - path: Specifies the path of the transformation.
65         - isAbsolute: Indicates whether the path is absolute or not.
66         - exists: Indicates whether the transformation exists or not.
67
68     - inFile: Specifies the input definition file option.
69         - Yes: Indicates that an input definition file is defined.
70         - No: Indicates that no input definition file is defined.
71         - path: Specifies the path of the input definition file.
72         - isAbsolute: Indicates whether the path is absolute or not.
73         - exists: Indicates whether the input definition file exists or not.
74
75 -->
76
77 <System name="Tcases">
78   <Function name="run">
79     <Input>
80
81       <!-- Option: -c -->
82       <VarSet name="defaultTupleSize">
83         <Var name="defined">
84           <Value name="Yes" property="defaultTupleSize"/>
85           <Value name="No"/>
86         </Var>
87
88         <Var name="isNumber" when="defaultTupleSize">
89           <Value name="Yes"/>
90           <Value name="No" failure="true"/>
91         </Var>
92       </VarSet>
93
94       <!-- Option: -f -->
95       <VarSet name="outFile">
96         <Var name="defined">
97           <Value name="Yes" property="outFile"/>
98           <Value name="No"/>
99           <Value name="TransformOutputUndefined" failure="true" when="

```

```

transformedOut, testFileExists"/>
100     </Var>
101
102     <VarSet name="path" when="outFile">
103         <Var name="isAbsolute">
104             <Value name="Yes"/>
105             <Value name="No"/>
106         </Var>
107         <Var name="exists">
108             <Value name="Yes" property="outFileExists"/>
109             <Value name="No"/>
110         </Var>
111     </VarSet>
112 </VarSet>
113
114 <!-- Option: -g -->
115 <VarSet name="genFile">
116     <Var name="defined">
117         <Value name="Yes" property="genFile"/>
118         <Value name="No"/>
119     </Var>
120
121     <VarSet name="path" when="genFile">
122         <Var name="isAbsolute">
123             <Value name="Yes"/>
124             <Value name="No"/>
125         </Var>
126         <Var name="exists">
127             <Value name="Yes"/>
128             <Value name="No" failure="true"/>
129         </Var>
130     </VarSet>
131
132     <Var name="default" whenNot="genFile">
133         <Value name="ForInputExists" when="inFile"/>
134         <Value name="ForInputNone" when="inFile"/>
135         <Value name="Standard" whenNot="inFile"/>
136     </Var>
137 </VarSet>
138
139 <!-- Option: -J -->
140 <Var name="isJUnit">
141     <Value name="Yes" property="isJUnit, transformedOut" whenNot="transform"/>
142     <Value name="No" when="transform"/>
143     <Value name="NotAllowed" when="transform" failure="true"/>
144 </Var>
145
146 <!-- Option: -n -->
147 <VarSet name="newTests">
148     <Var name="defined" when="testFileExists">
149         <Value name="Yes"/>
150         <Value name="No"/>
151     </Var>
152 </VarSet>
153
154 <!-- Option: -o -->
155 <VarSet name="outDir">
156     <Var name="defined">
157         <Value name="Yes" property="outDir"/>

```

```

158         <Value name="No"/>
159     </Var>
160
161     <VarSet name="path" when="outDir">
162         <Var name="exists">
163             <Value name="Yes" property="outDirExists"/>
164             <Value name="No"/>
165         </Var>
166         <Var name="isDirectory" when="outDirExists">
167             <Value name="Yes"/>
168             <Value name="No" failure="true"/>
169         </Var>
170     </VarSet>
171 </VarSet>
172
173 <!-- Option: -p -->
174 <VarSet name="transformParams">
175     <When>
176         <AnyOf property="transform, isJUnit"/>
177     </When>
178     <Var name="count">
179         <Value name="One" property="params"/>
180         <Value name="Many" property="params"/>
181         <Value name="None"/>
182     </Var>
183     <Var name="assignsValue" when="params">
184         <Value name="Yes"/>
185         <Value name="No" failure="true"/>
186     </Var>
187     <Var name="nameDefined" when="params">
188         <Value name="Yes"/>
189         <Value name="No" failure="true"/>
190     </Var>
191     <Var name="valueDefined" when="params">
192         <Value name="Yes"/>
193         <Value name="No"/>
194     </Var>
195 </VarSet>
196
197 <!-- Option: -r -->
198 <VarSet name="seed">
199     <Var name="defined">
200         <Value name="Yes" property="random"/>
201         <Value name="No"/>
202     </Var>
203
204     <Var name="isNumber" when="random">
205         <Value name="Yes"/>
206         <Value name="No" failure="true"/>
207     </Var>
208 </VarSet>
209
210 <!-- Option: -t -->
211 <VarSet name="testFile">
212     <Var name="defined">
213         <Value name="Yes" property="testFile"/>
214         <Value name="No"/>
215     </Var>
216

```

```

217     <VarSet name="path" when="testFile">
218         <Var name="isAbsolute">
219             <Value name="Yes"/>
220             <Value name="No"/>
221         </Var>
222         <Var name="exists">
223             <Value name="Yes" property="testFileExists"/>
224             <Value name="No"/>
225         </Var>
226     </VarSet>
227
228     <VarSet name="default" whenNot="testFile">
229         <Var name="exists">
230             <Value name="Yes" property="testFileExists"/>
231             <Value name="No"/>
232         </Var>
233     </VarSet>
234 </VarSet>
235
236 <!-- Option: -x -->
237 <VarSet name="transform">
238     <Var name="defined">
239         <Value name="Yes" property="transform, transformedOut"/>
240         <Value name="No"/>
241     </Var>
242     <VarSet name="path" when="transform">
243         <Var name="isAbsolute">
244             <Value name="Yes"/>
245             <Value name="No"/>
246         </Var>
247         <Var name="exists">
248             <Value name="Yes"/>
249             <Value name="No" failure="true"/>
250         </Var>
251     </VarSet>
252 </VarSet>
253
254 <!-- Input definition file -->
255 <VarSet name="inFile">
256     <Var name="defined">
257         <Value name="Yes" property="inFile"/>
258         <Value name="No"/>
259     </Var>
260     <VarSet name="path" when="inFile">
261         <Var name="isAbsolute">
262             <Value name="Yes"/>
263             <Value name="No"/>
264         </Var>
265         <Var name="exists">
266             <Value name="asDefined"/>
267             <Value name="withInputXml"/>
268             <Value name="withXml"/>
269             <Value name="No" failure="true"/>
270         </Var>
271     </VarSet>
272 </VarSet>
273
274 </Input>
275 </Function>

```

3.3 Test cases screenshot

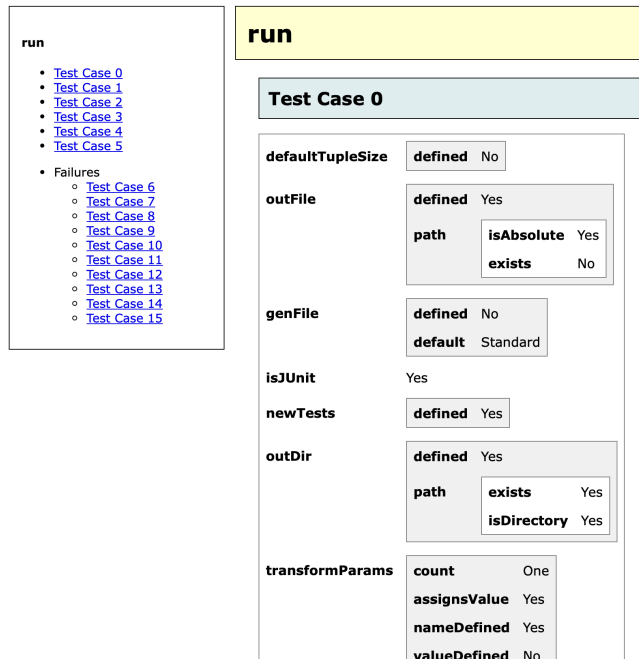


Figure 3: TCases Html View

4 Ice-Cream Input

4.1 Specification

Usage: Checking different Scoop level toppings of ice-cream We can add different levels of scoops in different condition of cone functions for example 1) If cone is not there then we will not add any scoop of ice-cream. 2) If cone is there and customer want Plain ice-cream then we can add maximum of 1 scoop and 1 topping in ice-cream. 3) If cone is there and customer want Plenty ice-cream then we can add min of 1 and maximum of 2 scoop and 2 topping in ice-cream. 4) If cone is there and customer want Grande ice-cream then we can add maximum of 4 scoop and 3 maximum and 1 minimum topping in ice-cream. 5) If cone is there and customer want Too-much ice-cream then we can add maximum of 3 scoop and 4 minimum topping in ice-cream. 6) If cone is there and customer want Too-much ice-cream then we can add maximum of 4 scoop and 5 minimum topping in ice-cream.

4.2 XML Code

```

1 <System name="Ice-Cream">
2   <Function name="Cones">
3     <!--
4     Usage: Checking different Scoop level & toppings of ice-cream
5     We can add different levels of scoops in different condition of cone functions
      for example

```

```

6      1) If cone is not there then we will not add any scoop of ice-cream.
7      2) If cone is there and customer want Plain ice-cream then we can add maximum
      of 1 scoop and 1 topping in ice-cream.
8      3) If cone is there and customer want Plenty ice-cream then we can add min of
      1 and maximum of 2 scoop and 2 topping in ice-cream.
9      4) If cone is there and customer want Grande ice-cream then we can add maximum
      of 4 scoop and 3 maximum and 1 minimum topping in ice-cream.
10     5) If cone is there and customer want Too-much ice-cream then we can add
      maximum of 3 scoop and 4 minimum topping in ice-cream.
11     6) If cone is there and customer want Too-much ice-cream then we can add
      maximum of 4 scoop and 5 minimum topping in ice-cream.
12 -->
13
14     <Input>
15         <Var name="Cone">
16             <Value name="Empty" failure="true">
17                 <When>
18                     <LessThan property="scoop" max="1"/>
19                 </When>
20             </Value>
21             <Value name="Plain">
22                 <When>
23                     <AllOf>
24                         <Equals property="scoop" count="1"/>
25                         <NotMoreThan property="topping" max="1"/>
26                     </AllOf>
27                 </When>
28             </Value>
29             <Value name="Plenty">
30                 <When>
31                     <AllOf>
32                         <Between property="scoop" min="1" max="2"/>
33                         <NotMoreThan property="topping" max="2"/>
34                     </AllOf>
35                 </When>
36             </Value>
37             <Value name="Grande">
38                 <When>
39                     <AllOf>
40                         <Between property="scoop" exclusiveMin="0"
exclusiveMax="4"/>
41                         <Between property="topping" min="1" max="3"/>
42                     </AllOf>
43                 </When>
44             </Value>
45             <Value name="Too-Much" failure="true">
46                 <When>
47                     <AnyOf>
48                         <MoreThan property="scoop" min="3"/>
49                         <NotLessThan property="topping" min="4"/>
50                     </AnyOf>
51                 </When>
52             </Value>
53         </Var>
54
55         <VarSet name="Flavors">
56             <Var name="Vanilla">
57                 <Value name="Yes" property="scoop"/>
58                 <Value name="No"/>

```

```

59         </Var>
60         <Var name="Chocolate">
61             <Value name="Yes" property="scoop"/>
62             <Value name="No"/>
63         </Var>
64         <Var name="Strawberry">
65             <Value name="Yes" property="scoop"/>
66             <Value name="No"/>
67         </Var>
68         <Var name="Pistachio">
69             <Value name="Yes" property="scoop"/>
70             <Value name="No"/>
71         </Var>
72         <Var name="Lemon">
73             <Value name="Yes" property="scoop"/>
74             <Value name="No"/>
75         </Var>
76         <Var name="Coffee">
77             <Value name="Yes" property="scoop"/>
78             <Value name="No"/>
79         </Var>
80     </VarSet>
81
82     <VarSet name="Toppings" when="scoop">
83         <Var name="Sprinkles">
84             <Value name="Yes" property="topping"/>
85             <Value name="No"/>
86         </Var>
87         <Var name="Pecans">
88             <Value name="Yes" property="topping"/>
89             <Value name="No"/>
90         </Var>
91         <Var name="Oreos">
92             <Value name="Yes" property="topping"/>
93             <Value name="No"/>
94         </Var>
95         <Var name="Cherries">
96             <Value name="Yes" property="topping"/>
97             <Value name="No"/>
98         </Var>
99         <Var name="MMs">
100             <Value name="Yes" property="topping"/>
101             <Value name="No"/>
102         </Var>
103         <Var name="Peppermint">
104             <Value name="Yes" property="topping"/>
105             <Value name="No"/>
106         </Var>
107     </VarSet>
108 </Input>
109 </Function>
110 </System>

```

4.3 Test cases screenshot

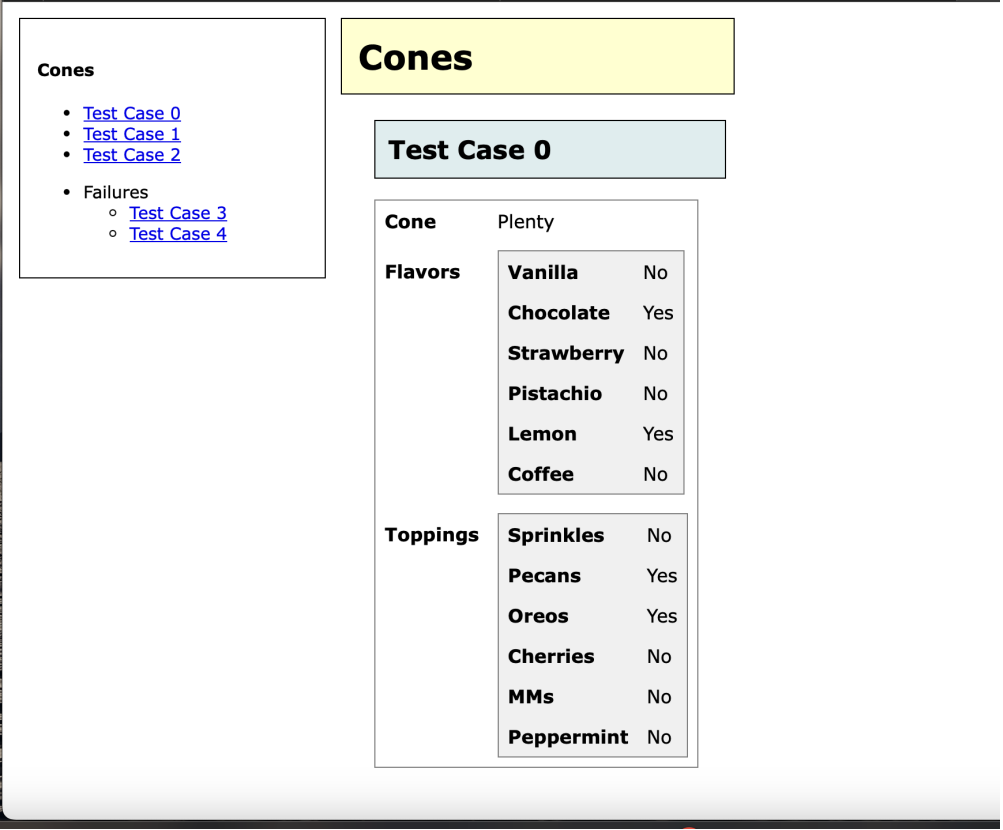


Figure 4: TCases Html View