National Institute of Technology, Warangal

End Lab Exam – MCA 5th Sem

Software Testing

Name – Divyanshu Shrivastava

Roll No. – 187909

MCA 3rd Year

Reg. No. – MC18111

Task 1 – TCASES

Definition –

Tcases is a tool for designing tests. It doesn't matter what kind of system you are testing. Nor does it matter what level of the system you are testing — unit, subsystem, or full system. You can use Tcases to design your tests in any of these situations. With Tcases, you define the input space for your system-under-test and the level of coverage that you want. Then Tcases generates a minimal set of test cases that meets your requirements.

Tcases is primarily a tool for black-box test design. For such tests, the concept of "coverage" is different from structural testing critieria such as line coverage, branch converage, etc. Instead, Tcases is guided by coverage of the input space of your system.

How Tcases Work?

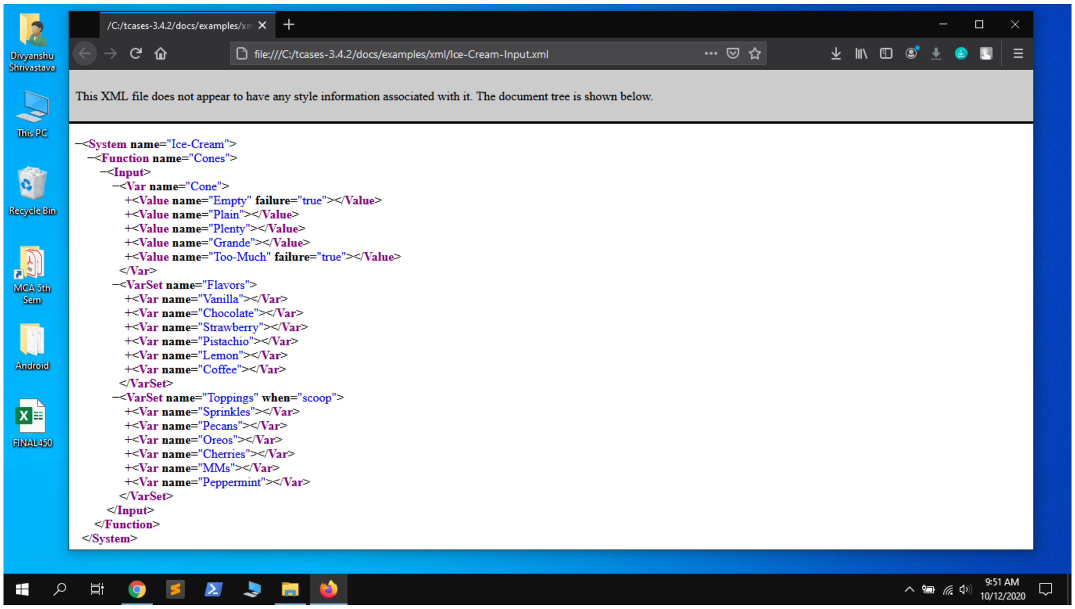
First, you create a system input definition, a document that defines your system as a set of functions. For each system function, the system input definition defines the variables that characterize the function input space.

Then, you can create a generator definition. That's another document that defines the coverage you want for each system function. The generator definition is optional. You can skip this step and still get a basic level of coverage.

Finally, you run Tcases. Tcases is a Java program that you can run from the command line or from your favorite IDE. Tcases comes with built-in support for running using a shell script or an **ant** target. You can also run Tcases with Maven using the Tcases Maven Plugin. Using your input definition and your generator definition, Tcases generates a system test definition. The system test definition is a document that lists, for each system function, a set of test cases that provides the specified level of coverage. Each test case defines a specific value for every function input variable. Tcases generates not only valid input values that define successful test cases but also invalid values for the tests cases that are needed to verify expected error handling.

Running XML File

1. Input File – **Ice-Cream-Input.xml**



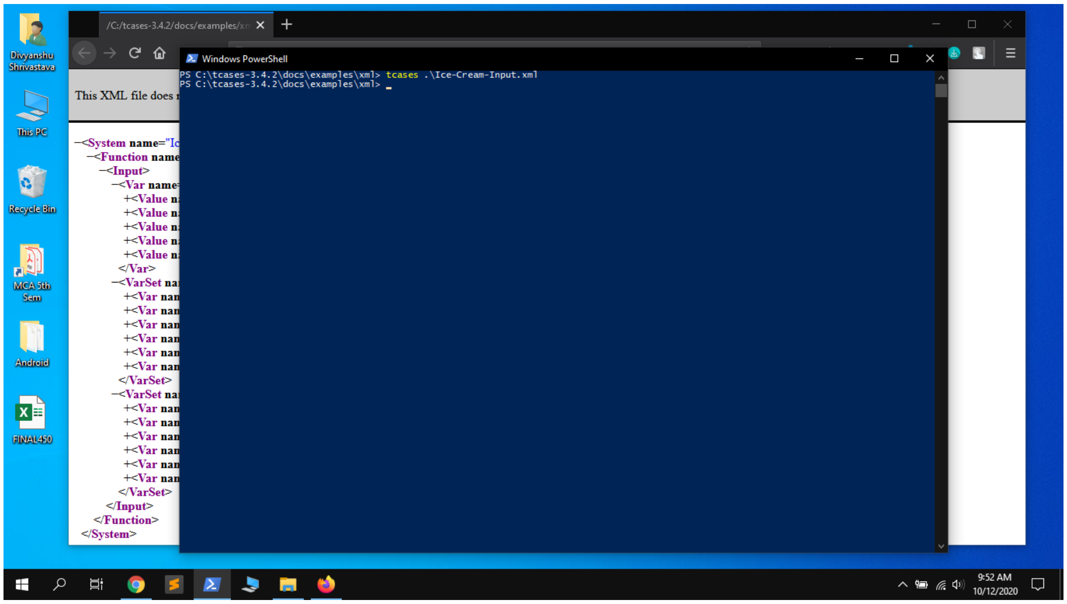
/C:/tcases-3.4.2/docs/examples/xm
file:///C:/tcases-3.4.2/docs/examples/xml/Ice-Cream-Input.xml
Divyanshu Shrivastava
This PC
Recycle Bin
MCA 5th Sem
Android
FINAL450
9:51 AM
10/12/2020

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
−<System name="Ice-Cream">
  −<Function name="Cones">
    −<Input>
      −<Var name="Cone">
        +<Value name="Empty" failure="true"></Value>
        +<Value name="Plain"></Value>
        +<Value name="Plenty"></Value>
        +<Value name="Grande"></Value>
        +<Value name="Too-Much" failure="true"></Value>
       </Var>
      −<VarSet name="Flavors">
        +<Var name="Vanilla"></Var>
        +<Var name="Chocolate"></Var>
        +<Var name="Strawberry"></Var>
        +<Var name="Pistachio"></Var>
        +<Var name="Lemon"></Var>
        +<Var name="Coffee"></Var>
       </VarSet>
      −<VarSet name="Toppings" when="scoop">
        +<Var name="Sprinkles"></Var>
        +<Var name="Pecans"></Var>
        +<Var name="Oreos"></Var>
        +<Var name="Cherries"></Var>
        +<Var name="MMs"></Var>
        +<Var name="Peppermint"></Var>
       </VarSet>
     </Input>
   </Function>
 </System>
```

The above image is showing the input file that we are using for Tcases which mainly three types of input Cone, Flavors and Toppings.

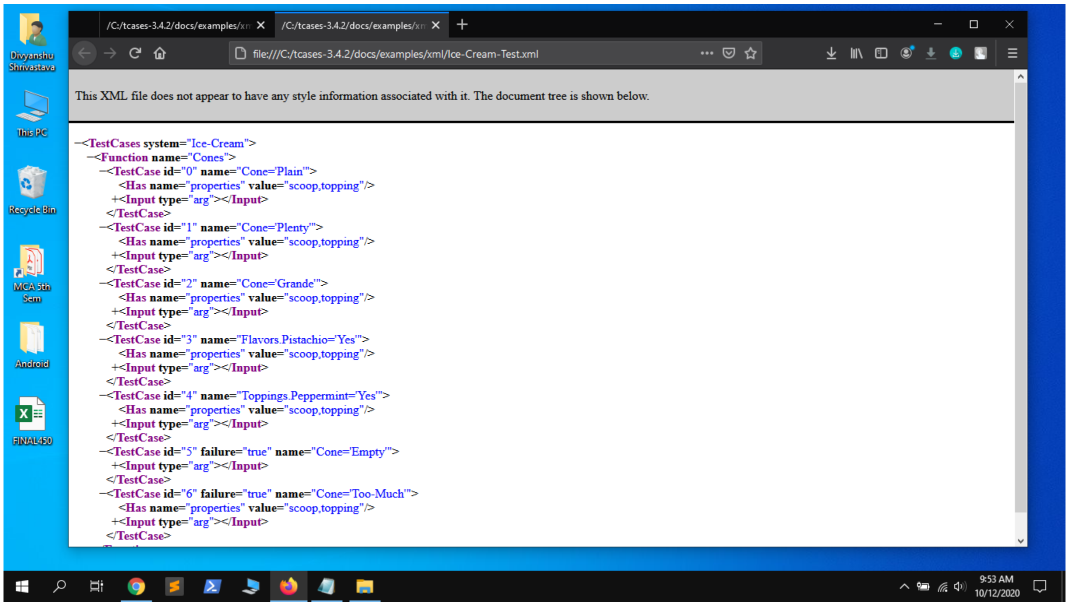
2. **Running Input File.**



```
PS C:\tcases-3.4.2\docs\examples\xml> tcases .\Ice-Cream-Input.xml
PS C:\tcases-3.4.2\docs\examples\xml> _
```

We have executed the input file by using the command –
tcases Ice-Cream-Input.xml
which when completed successfully generates output file Ice-Cream-Test.xml within the same directory.

3. Output File – Ice-Cream-Test.xml



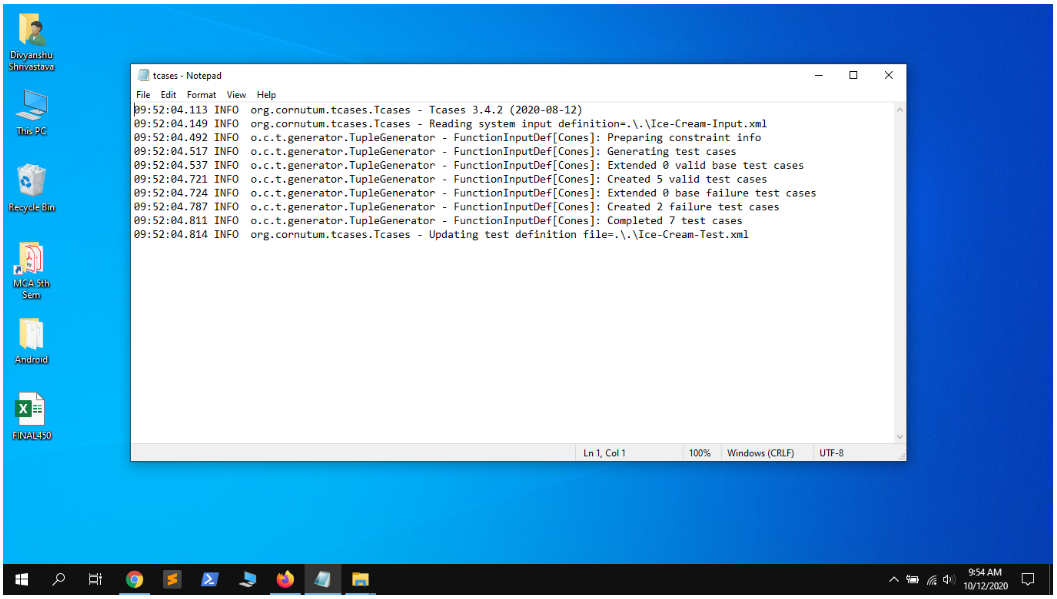
/C:/tcases-3.4.2/docs/examples/xn
/C:/tcases-3.4.2/docs/examples/xn
file:///C:/tcases-3.4.2/docs/examples/xml/Ice-Cream-Test.xml
Divyanshu Shrivastava
This PC
Recycle Bin
MCA 5th Sem
Android
FINAL450
9:53 AM
10/12/2020

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
−<TestCases system="Ice-Cream">
  −<Function name="Cones">
    −<TestCase id="0" name="Cone='Plain'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="1" name="Cone='Plenty'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="2" name="Cone='Grande'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="3" name="Flavors.Pistachio='Yes'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="4" name="Toppings.Peppermint='Yes'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="5" failure="true" name="Cone='Empty'">
      +<Input type="arg"></Input>
     </TestCase>
    −<TestCase id="6" failure="true" name="Cone='Too-Much'">
        <Has name="properties" value="scoop,topping"/>
      +<Input type="arg"></Input>
     </TestCase>
```

From the output file we can observe that it gives 7 different test ids by the combination of values present.

4. Tcases log file

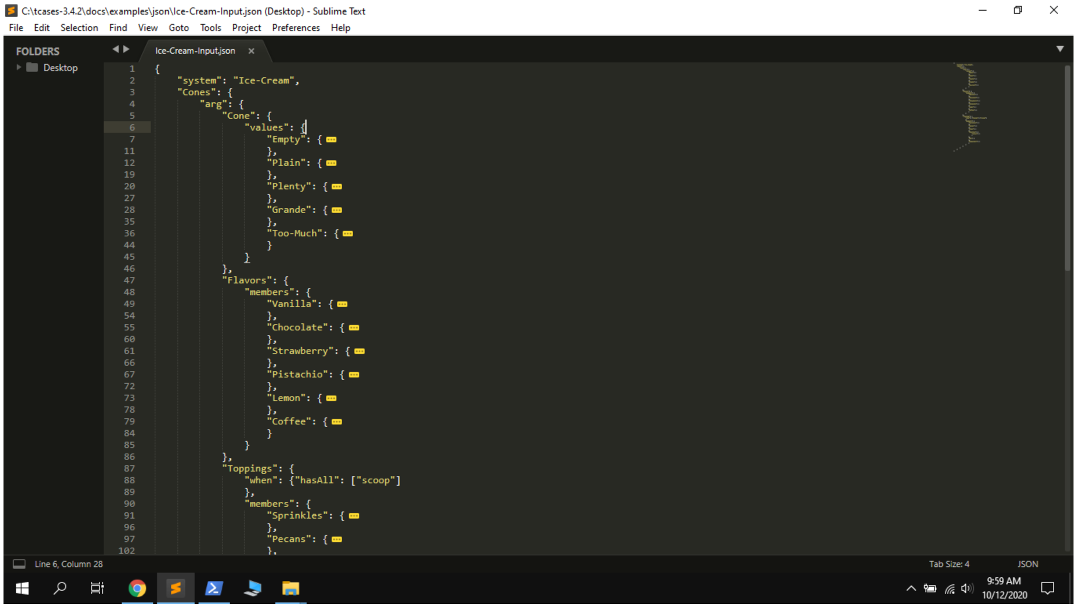


```
09:52:04.113 INFO  org.cornutum.tcases.Tcases - Tcases 3.4.2 (2020-08-12)
09:52:04.149 INFO  org.cornutum.tcases.Tcases - Reading system input definition=.\.\Ice-Cream-Input.xml
09:52:04.492 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Preparing constraint info
09:52:04.517 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Generating test cases
09:52:04.537 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Extended 0 valid base test cases
09:52:04.721 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Created 5 valid test cases
09:52:04.724 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Extended 0 base failure test cases
09:52:04.787 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Created 2 failure test cases
09:52:04.811 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Completed 7 test cases
09:52:04.814 INFO  org.cornutum.tcases.Tcases - Updating test definition file=.\.\Ice-Cream-Test.xml
```

The tcases.log file clearly shows that it created 5 valid test cases and 2 failure or invalid test Cases out of total 7 test cases.

Running JSON File

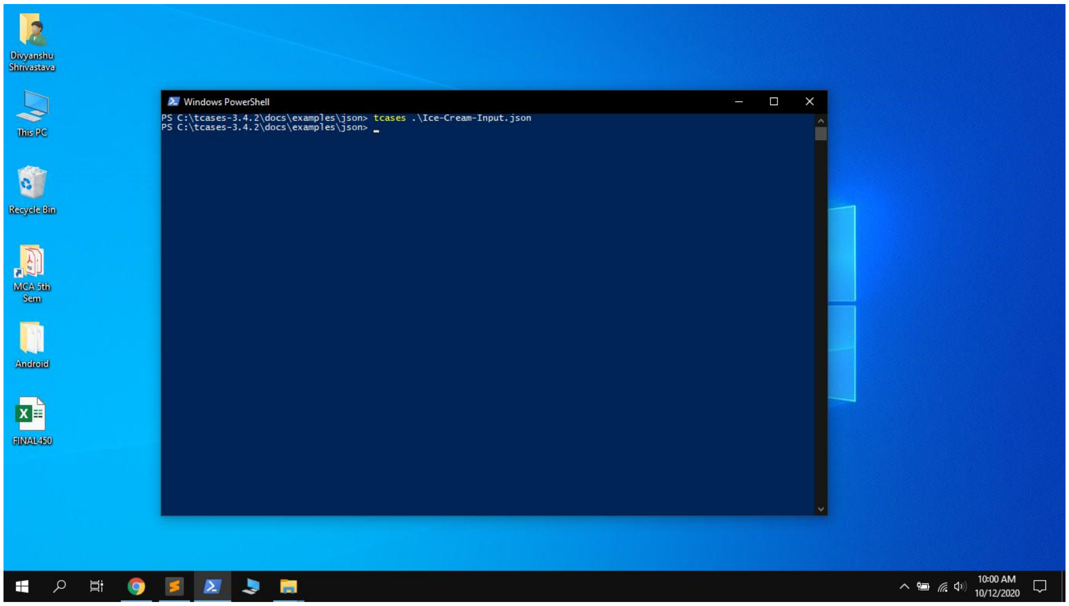
1. Input File – **Ice-Cream-Input.json**


C:\tcases-3.4.2\docs\examples\json\Ice-Cream-Input.json (Desktop) - Sublime Text
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
Desktop
Ice-Cream-Input.json
Line 6, Column 28
Tab Size: 4
JSON
9:59 AM
10/12/2020

```
{
    "system": "Ice-Cream",
    "Cones": {
        "arg": {
            "Cone": {
                "values": {
                    "Empty": { ⋯
                    },
                    "Plain": { ⋯
                    },
                    "Plenty": { ⋯
                    },
                    "Grande": { ⋯
                    },
                    "Too-Much": { ⋯
                    }
                }
            },
            "Flavors": {
                "members": {
                    "Vanilla": { ⋯
                    },
                    "Chocolate": { ⋯
                    },
                    "Strawberry": { ⋯
                    },
                    "Pistachio": { ⋯
                    },
                    "Lemon": { ⋯
                    },
                    "Coffee": { ⋯
                    }
                }
            },
            "Toppings": {
                "when": {"hasAll": ["scoop"]
                },
                "members": {
                    "Sprinkles": { ⋯
                    },
                    "Pecans": { ⋯
                    },
```

The above image is showing the input file that we are using for Tcases which mainly three types of input Cone, Flavors and Toppings and each of these having subparts.

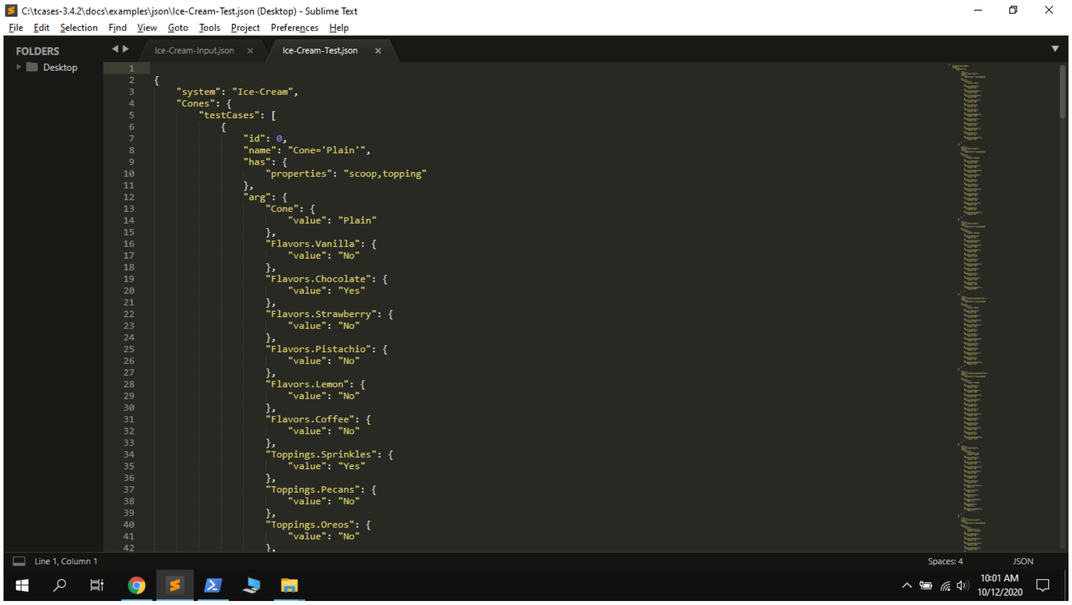
2. **Running Input File.**



```
PS C:\tcases-3.4.2\docs\examples\json> tcases .\Ice-Cream-Input.json
PS C:\tcases-3.4.2\docs\examples\json> _
```

We have executed the input file by using the command –
tcases Ice-Cream-Input.json
which when completed successfully generates output file Ice-Cream-Test.json within the same directory.

3. Output File – Ice-Cream-Test.json

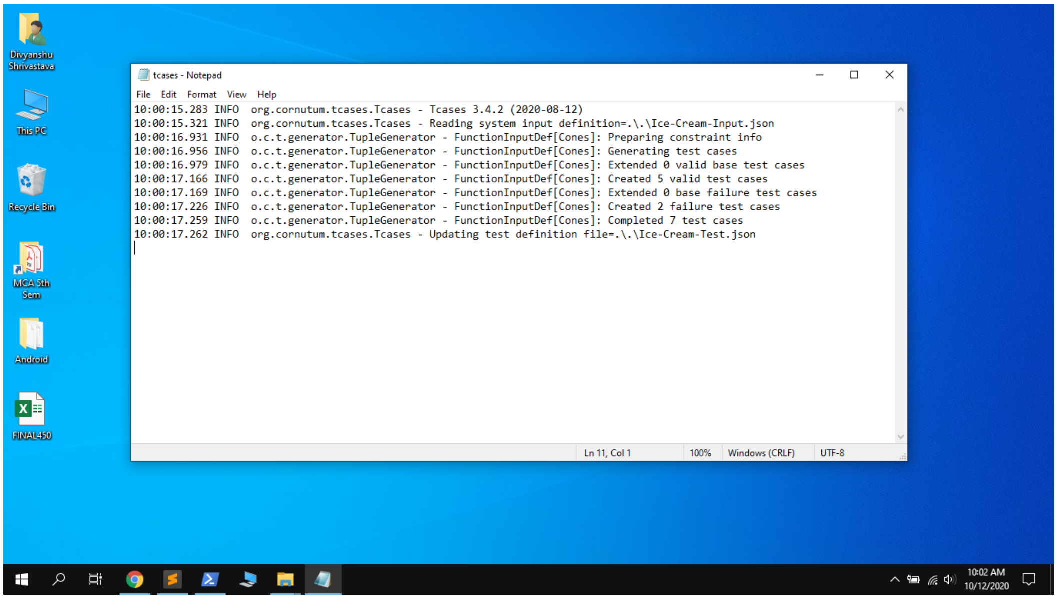

C:\tcases-3.4.2\docs\examples\json\Ice-Cream-Test.json (Desktop) - Sublime Text
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
Desktop
Ice-Cream-Input.json
Ice-Cream-Test.json
Line 1, Column 1
Spaces: 4
JSON
10:01 AM
10/12/2020

```

{
    "system": "Ice-Cream",
    "Cones": {
        "testCases": [
            {
                "id": 0,
                "name": "Cone='Plain'",
                "has": {
                    "properties": "scoop,topping"
                },
                "arg": {
                    "Cone": {
                        "value": "Plain"
                    },
                    "Flavors.Vanilla": {
                        "value": "No"
                    },
                    "Flavors.Chocolate": {
                        "value": "Yes"
                    },
                    "Flavors.Strawberry": {
                        "value": "No"
                    },
                    "Flavors.Pistachio": {
                        "value": "No"
                    },
                    "Flavors.Lemon": {
                        "value": "No"
                    },
                    "Flavors.Coffee": {
                        "value": "No"
                    },
                    "Toppings.Sprinkles": {
                        "value": "Yes"
                    },
                    "Toppings.Pecans": {
                        "value": "No"
                    },
                    "Toppings.Oreos": {
                        "value": "No"
                    },
```

From the output file we can observe that it gives 7 different test ids by the combination of values present.

4. Tcases log file



```
10:00:15.283 INFO  org.cornutum.tcases.Tcases - Tcases 3.4.2 (2020-08-12)
10:00:15.321 INFO  org.cornutum.tcases.Tcases - Reading system input definition=.\.\Ice-Cream-Input.json
10:00:16.931 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Preparing constraint info
10:00:16.956 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Generating test cases
10:00:16.979 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Extended 0 valid base test cases
10:00:17.166 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Created 5 valid test cases
10:00:17.169 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Extended 0 base failure test cases
10:00:17.226 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Created 2 failure test cases
10:00:17.259 INFO  o.c.t.generator.TupleGenerator - FunctionInputDef[Cones]: Completed 7 test cases
10:00:17.262 INFO  org.cornutum.tcases.Tcases - Updating test definition file=.\.\Ice-Cream-Test.json
```

The tcases.log file clearly shows that it created 5 valid test cases and 2 failure or invalid test Cases out of total 7 test cases.