**Product and CellPhone Object Class**

**Test Plan**

**I. Introduction**

The tests in this plan are designed to ensure the functionality of two object classes and their methods. The parent class, Product, contains a default constructor and an initialization constructor, as well as mutator and accessor methods for each of the Product class’s three properties. The class also has a compareTo method, and equals method, and toString method. The CellPhone class is a subclass of the Product. It inherits Product’s methods and properties, but it also has additional method and overrides some of the Product’s methods.

**II. Items to Be Tested**

Because the CellPhone class inherits many of its properties and methods from the parent class Product, the testing methods will be very similar between the two classes. For organizational purposes, the following methods will be listed together because the constructs, requirements, and testing methods will be similar.

**Constructor Methods**: Product() and CellPhone()

Do the constructor methods in both classes create the desired object with the correct properties? Does it differentiate from the default and initialization constructor when loaded with different parameters? What happens if invalid parameters are loaded in the constructors?

**Mutator Methods:** setProductNumber(), setName(), setPrice(), setModel(), setMacNumber(), and setScreenSize()

Do these mutator methods change the value of the right property? What happens if invalid parameters are loaded?

**Accessor Methods:** getProductNumber(), getName(), getPrice(), getModel(), getMacNumber(), and getScreenSize()

Do these accessor methods return the right value from the object’s property?

**Other Methods:**

The toCompare() method in both classes – Does the method return the correct integer value? Is comparing the Product number property or something else?

The equals() method in both classes - Does the method return the correct Boolean value?

**III. Items Not Being Tested**

The toString method will not be tested because it was rigorously tested and validated prior to see if it returns the correct string value. The toString method was tested by loading the properties with random values and see if it returned the expected value. It’s a crucial method used to return all properties of the object and needed to clarify if the constructor methods work. It is also easier to validate that method on its own. In addition, since the constructor method and the toString method rely on each to create and present the data, testing the constructor methods will at the same time validate the toString.

There will not be any tests that explicitly check if the CellPhone class has inherited its properties and methods from the Product class. If the other tests pass, then it will mean that the CellPhone class inherited the methods and properties.

There also will be no test regarding whether the parameter inputted into the methods are valid or not. Since these are object classes, the right implementation depends on the programmer using these methods. An invalid parameter will cause a compile error. The programmer must include Exceptions for end-user input to catch invalid parameters.

**IV. Tests to Be Conducted:**   
**Constructor Methods**: Product() and CellPhone()

1. **Input test 1** – This test will check if the default constructors in both classes are using the default properties and properly storing the data. The test will be conducted using a driver program with no data, and the toString method will be used to obtain the results. The method will pass if the objects are created with the default values.
2. **Input test 2-3** – The test will check if the initialization constructors in both classes are using the parameters for their properties and properly storing the data. The test will be conducted using a driver with correctly formatted data to create objects and the toString method will be used to obtain the results. The method will pass if the objects are created with the parameter as their property values.

**Mutator Methods:** setProductNumber(), setName(), setPrice(), setModel(), setMacNumber(), and setScreenSize()

1. **Input test** **4-9**– Each method will be tested to see if it changes the desired property through the CellPhone class, as stated before, due to the fact it inherits some methods from the Product class. The driver program will load a correctly formatted data file as the parameters and creates objects. Then it will call the set methods and compare the changes made with the toString method. The method will pass if the change is made to the expected value.

**Accessor Methods:** getProductNumber(), getName(), getPrice(), getModel(), getMacNumber(), and getScreenSize()

1. **Output test 1**– Each get method will be tested in one test module to see if the method returns the correct property value through the CellPhone class, as stated before, due to the fact it inherits some methods from the Product class. The driver program will load data from a file as parameters for a new object. Then it will call each get method to get the desired property and compare the expected value and the returned value. The method will pass if the both are the same.

**Other Methods:**

1. **toCompare() correctness test 1 - 2** - This method will be tested to see if it returns the correct value when comparing two Product object’s product number and two Cell Phones object’s model. The test driver will load a data file to create several different objects and call the method to test the different scenarios. The returned value will then be compared to the expected value. The method will pass the test if the two are equal.
2. **equals() correctness test 3 - 4**- This will test if the method returns the correct Boolean value when two object’s properties are compared. The test driver will load a data file to create several different objects and call the method to test different scenarios. The returned value will then be compared to the expected value. The method will pass the test if the two matches.

**V. Environment Needed for the Test:**   
**Hardware –** The software will be tested using the same hardware used for development.

**Software –** The software will be tested using the same software used for development.

* A data file with both correctly and incorrectly formatted data
* Test drivers for each method as described before
* An output file that reads the results of the automated test process

**VI. Test Deliverables:**

An automated generated pass/fail report will be provided with a description of each test and data file used for the test. If a test failed, the report will indicate why and where. The source code for the tests will also be provided.

**VII. Responsibility for the Tests:**

All test will be conducted by the software developer internally.