Seneca College

Applied Arts & Technology

SCHOOL OF COMPUTER STUDIES

Workshop - 1

Due Date: 24th Sep 2025

INSTRUCTIONS

- *This workshop must be completed individually without any outside collaboration. All work must be your own. Copying or reproducing the work done by others (in part or in full) or letting others to copy or reproduce your own work is subject to significant grade reduction or getting no grade at all and/or being treated as academic dishonesty under the College's Academic Dishonesty Policy.*
- *This is an out of class workshop and you are required to complete this workshop on your own time, unless otherwise specified by your instructor.*
- *Your application must compile and run upon download to receive any mark. If the corresponding program fails to compile and run it will receive ZERO grade.*
- *To submit the workshop, please follow the Submission Guideline provided at the end of this document.*
- *You must submit the workshop by the due date mentioned above. Late submissions policy is specified in the Academic Procedures for Evaluations document available through the class plan on blackboard.*
- *Total mark = 6% of the final grade.*

- *Make sure you go through/ read carefully all the requirements given below and understand them very clearly before starting coding you solutions.*

Electronic Devices

Design and implement the following interface and class hierarchy (shown in Figure 1) in java to demonstrate the relationships and functions among various electronic devices in a typical electronic shop.

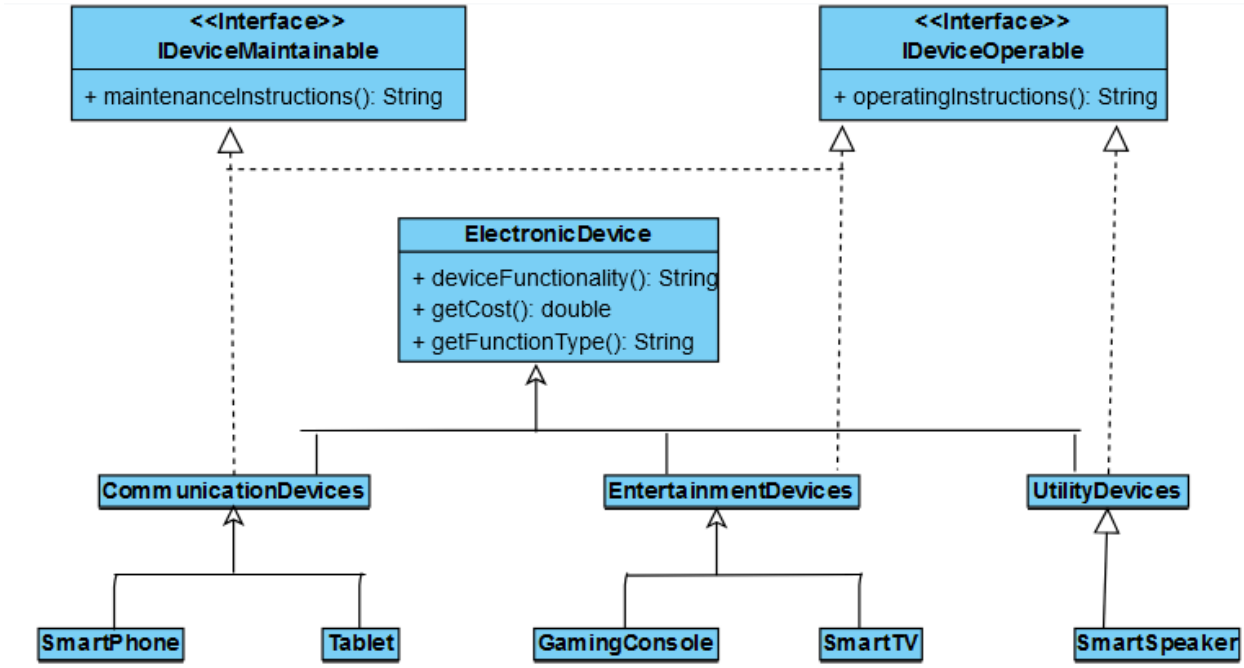

Figure 1: *`IDeviceMaintanable` and `IDeviceOperable` are two interfaces. `ElectronicDevice`, `CommunicationDevices`, EntertainmentDevices, and `UtilityDevices` are all abstract classes. SmartPhone, Tablet, GamingConsole, SmartTV and SmartSpeaker are concrete classes.*

Most information handled and/or returned by interface and abstract class methods are presented in Table 1. Use this table for information to be returned by your method implementations within respective class.

Table 1: Information on electronic devices

Device	Functionality	Price	How to Operate	How to Maintain	Function Type
Smartphone	Communication and apps	$699.00	By using touchscreen	Regular software updates	Multi-functional
Tablet	Larger screen communication	$499.00	By using touchscreen	Regular software updates	Multi-functional
GamingConsole	Video gaming	$399.00	By using game controllers	Clean vents, software updates	Interactive entertainment

SmartTV	Streaming and media viewing	$799.00	By using remote control	Update firmware, clean screen	Visual entertainment
SmartSpeaker	Voice-controlled assistance	$149.00	By using voice commands	N/A	Audio assistance

Your implementation should:

i) **[*Requirement 1*]** Receive the price value for each device from the user to create an object of that device. (Other information for that device object e.g., functionality, how to operate, how to maintain, and function type can be hard coded as per the table 1).

ii) **[*Requirement 2*]** Display the How to Operate, How to Maintain, Function Type, and Price information for the most expensive device by comparing objects of all devices.

iii) **[*Requirement 3*]** Display the names of all devices (Smartphone, Tablet, GamingConsole, SmartTV, SmartSpeaker) in descending order of price. Override the ToString() method to display the name of a device.

iv) **[*Requirement 4*]** Receive a device category name (e.g., CommunicationDevices, EntertainmentDevices, or UtilityDevices) from the user, and display how each device in that given category functions.

Your implementation must also satisfy following requirements:

v) **[*Requirement 5*]** You must demonstrate the use of `Comparable` interface and implement its `compareTo()` method (or `Comparator` interface and implement its `compare()` method) to find the device object with the highest price and sort them.

vi) **[*Requirement 6*]** You must demonstrate the use of an array of objects (e.g., an array of type `ElectronicDevice` to hold objects of each concrete type of electronic device). Use the `compareTo()` method that you may should overridden in `ElectronicDevice` class (or `compare()` method that you might have implemented in a separate helper class) to find the instrument object with the highest price and sort them in the collection. No operation on the instruments should be hard coded – meaning all operations must be done programmatically.

vii) **[*Requirement 7*]** Your program must demonstrate proper modularization of code, java industry standard with proper comments, indentations, and naming convention.

viii) **[*Requirement 8*]** You must include the following information as commented at the beginning of **each** class file/ interface.

/***
Workshop #
Course:*<subject type> - Semester*
Last Name:*<student last name>*
First Name:*<student first name>*
ID:*<student ID>*
Section:*<section name>*
This assignment represents my own work in accordance with Seneca Academic Policy.
Signature
Date:*<submission date>*
***/

ix) **[*Requirement 9*]** You must test your application for multiple runs with different price values for all instruments. Change the price value for each instrument to show that your application works even if the price of instruments is updated.

x) ***[Requirement 10]*** Your code must follow the MVC design pattern.

Sample Output: On the above information of musical instruments in Table 1, the output of a test run should be as follows:

```
--: Requirement 1 :--
Enter the price for SmartPhone: 699
Enter the price for Tablet: 499
Enter the price for GamingConsole: 399
Enter the price for SmartTV: 799
Enter the price for SmartSpeaker: 149

--: Requirement 2 :--
The most expensive device is: SmartTV
SmartTV's cost is: $799.0
SmartTV is operated: By using remote control
SmartTV maintenance: Update firmware, clean screen
SmartTV function type: Visual entertainment
--: Requirement 3 :--

Devices in Descending Order of Price:
SmartTV
SmartPhone
```

Tablet
GamingConsole
SmartSpeaker

--: Requirement 4 :--
Enter a device category (CommunicationDevices, EntertainmentDevices, UtilityDevices): CommunicationDevices

Functionality of CommunicationDevices:
SmartPhone: Communication and apps
Tablet: Larger screen communication

Code Submission Criteria:

Please note that you should have:

- Appropriate indentation.
- Proper package names
- Proper file structure
- Follow java naming convention.
- Do Not have any debug/ useless code and/ or files in the assignment.

Deliverables and Important Notes:

All these deliverables are supposed to be uploaded on the blackboard once done.

- Complete the code behind for the requirements plus if any other helper classes or functions required . 60%
- Submit a **reflect.txt** file with the submission. 20%
  - Reflect on how MVC helped you in code organization.
- Video submission explaining core code pointers and showing the full working application (3 - 5 minutes max). 20%
- All submission goes to Black Board.
- Your submission should include
  - Video file with audio
  - Reflect.txt file
  - Complete zipped project.
- Late submissions would result in additional 10% penalties for each day or part of it.

Remember that you are encouraged to talk to each other, to the instructor, or to anyone else about any of the workshops, but the final solution may not be copied from any source.