# Hands-on Experiment # 12: Worksheet

Section\_\_\_\_\_\_\_1\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_17/11/2563\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student ID \_\_\_\_\_\_\_\_\_\_6338110221\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Name\_\_\_\_\_\_\_\_Nonthapat Keawamporn\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

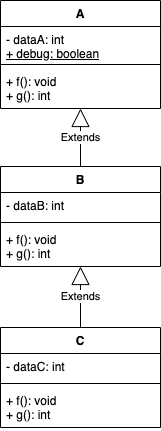
## Part A: Getting Familiar with Inheritance

In part A, we aim to practice the behavior of inheritance class hierarchy. We have the class hierarchy as show in the picture. Class C inherits from class B and B inherits from A.

A program InheritanceDemo is given. Create class A.java, B.java, and C.java to make sure that program InheritanceDemo runs correctly.

**Your** **task**

* When testConstructor() is called, the program must print out the sequence of constructors being called.
* testF() and testG() provide expecting results when f() and g() are called.

Class A

* Fields:
  + private int dataA
  + public static Boolean debug, set to true when want to show the constructor’s calling sequence.su
* constructor A(int a), initializes dataA to a
* f() prints text to show that f() of A is being called.
* g() returns dataA + 1

Class B

* Fields:
  + private int dataB
* constructor B(int a, int b), initializes dataA to a, dataB to b
* f() prints text to show that f() of B is being called.
* g() returns 2 \* super class’s g() + dataB

Class C

* Fields:
  + private int dataC
* constructor C(int a, int b, int c), initializes dataA to a, dataB to b and dataC to c
* f() calls f() of its superclass then prints text to shows the f() of C is being called.
* g() returns super class’s g() - 10 + dataC \* 3

## Part B: Applying Inheritance in Observer Pattern

Observer pattern is used when there is one-to-many relationship between objects such as if one object is modified, its dependent objects are to be notified automatically.

Diagram

Description automatically generated

From the diagram above, when Data is modified, all Display hierarchy will be notified automatically and updated. DataDisplay and it’s subclasses are call observers of Data.

Graphical user interface, text, application

Description automatically generated

Class Data:

* content: the data that this class store, for this lab we store our data as ArrayList<Integer>
* observers: collection of observer class that will be notified when this class is modified

If we want to any DataDisplay to be notified we need to attach that DataDisplay to the collection, and every time the Data is modified, it should call method notifyAllObservers()

A picture containing graphical user interface, application

Description automatically generated

When you run PartB.java the program run and wait for user to enter an integer number. Each time you enter a number, the program show the message that new data is added and show the updated from the DataDisplay with the content. To exit the program enter a negative number. The example of program execution is shown above.

**Your** **task**

Create three classes that inherit from DataDisplay

* MaxDataDisplay: displays max value of content in Data

Text

Description automatically generated

* HistogramDataDisplay: displays content as histogram

Text

Description automatically generated

* BorderDataDisplay: display similar as DataDisplay but show top and bottom border.

Text, letter

Description automatically generated

**Submission Instruction:**

Submit all class into one .zip file name **YourID-Lab12.zip** containing:

* **PartA.java**
* **A.java**
* **B.java**
* **C.java**
* **PartB.java**
* **DataDisplay.java**
* **MaxDataDisplay.java**
* **HistogramDataDisplay.java**
* **BorderDataDisplay.java**

Submit via <http://www.myCourseVille.com> (Assignments > Lab # 12) before deadline specified in MCV.