CSE 1321L: Programming and Problem Solving I Lab

Lab 13

Introduction to Java

What students will learn:

* Installing IntelliJ
* Creating a new IntelliJ project
* Creating new files inside an IntelliJ project

Overview: This week, we’ll be writing our code in Java instead of Python. To do this, you will need an IDE which can compile and run java code. For the purposes of this class, **we recommend that you use IntelliJ**, given that it is developed by the same company that developed PyCharm, JetBrains. However, feel free to use any IDEs you are familiar with which can run Java (even notepad and the command prompt if you know how).

Installing IntelliJ: First, you must install IntelliJ. To do so, go to the IntelliJ section of JetBrains’ website. The link can be found below:

<https://www.jetbrains.com/idea/download/>

Select your operating system at the top and then download IntelliJ. Be sure to **download and install the Community Edition, not the Ultimate Edition:**



Once you are done installing, run IntelliJ. In the screen that appears, select the option to create a new project. Select the following options.

* New Project (on the left menu)
* Name: Lab13
* Location: (leave as is)
* Language: Java
* Build system: IntelliJ

Next, click on the JDK option. Select “Add JDK” then “Download JDK”.



In the window that appears, select the highest version available, “Oracle OpenJDK” as the vendor, and leave the Location as is.



Wait for the installation to be done. If everything’s gone through, your screen should look like this:



**(Note that you only have to download the JDK once. For all future projects, the JDK should be automatically selected)**

Press “Create” to create your project.

Using IntelliJ: As has been mentioned, IntelliJ is maintained by the same company which developed PyCharm. As such, the interface should be largely familiar to you. On the top left, you can click the little folder icon to reveal all the files inside of your project.

Whenever you create a new file, be sure to create it inside the “src” folder, to keep things managed. You can create a new file by right clicking the “src” folder, and selecting “New > Java Class”. Enter your class name, and the file will be created.



You are now all set and ready to code in Java. Be sure to refer to your lecture material for all Java syntax.

A few things to keep in mind:

* All code in Java must go inside of a class (except for classes themselves).
* Most code you write will need to go inside of a method.
* Files can only be run if they have a main() method.
	+ The main method always has the header below:

public static void main(String[] args)

* Scope in Java is determined by the presence of curly braces, not of tabs.
* Like in Python, be mindful of the capitalization of words (“Word” is different than “word”).
* Remember that all variables, method parameters, and method return types must have an explicitly declared type.

**Lab13A**: Write the three static methods below inside of your driver class (this is the class that contains your main method):

* min(): Receives an integer array as a parameter and returns an integer. This method traverses the array in the parameters and returns the smallest number in the array.
* max(): Receives an integer array as a parameter and returns an integer. This method traverses the array in the parameters and returns the largest number in the array.
* avg(): Receives an integer array as a parameter and returns a double. This method traverses the array in the parameters and returns the mean average of the numbers in the array.

Inside your main method, prompt the user for a size for the array you are about to create; keep prompting the user if they enter a size that’s less than 1. Once the user enters a valid number, create an integer array of that size.

Next, prompt the user for enough numbers to fill the whole array. These numbers can be positive, negative, or 0.

Next, ask the user if they want to find the smallest, the largest, or the mean average of all the numbers they’ve entered (keep prompting the user until they enter a valid choice). Pass the array to the appropriate method and print the results.

Keep asking the user for an option until they enter the option to quit.

**You can assume that the user will not enter doubles or strings when prompted for a number.**

***Sample input on the next page.***

Sample output (user input in **bold**):

How many numbers would you like to enter? **-30**

You must enter a value that is greater than 0.

How many numbers would you like to enter? **0**

You must enter a value that is greater than 0.

How many numbers would you like to enter? **5**

Enter number 0: **6**

Enter number 1: **-99**

Enter number 2: **37**

Enter number 3: **2000**

Enter number 4: **81**

1 - Find the smallest of the numbers entered

2 - Find the largest of the numbers entered

3 - Find the average of the numbers entered

4 - Quit

Enter your option: **1**

The smallest of the numbers you entered is -99

1 - Find the smallest of the numbers entered

2 - Find the largest of the numbers entered

3 - Find the average of the numbers entered

4 - Quit

Enter your option: **2**

The largest of the numbers you entered is 2000

1 - Find the smallest of the numbers entered

2 - Find the largest of the numbers entered

3 - Find the average of the numbers entered

4 - Quit

Enter your option: **3**

The average of the numbers you entered is 405.0

1 - Find the smallest of the numbers entered

2 - Find the largest of the numbers entered

3 - Find the average of the numbers entered

4 - Quit

Enter your option: **5**

That is not a valid option!

1 - Find the smallest of the numbers entered

2 - Find the largest of the numbers entered

3 - Find the average of the numbers entered

4 - Quit

Enter your option: **4**

Shutting off...

**Lab13B**: Create a Book class which contains the following:

* Fields:
	+ name: string
	+ price: double
	+ yearOfRelease: int
* Constructor:
	+ Takes in 2 parameters: a string, and an integer.
	+ Sets the 2 parameters to the appropriate fields.
* setPrice():
	+ Takes in a double as a parameter and returns nothing.
	+ Sets the price field with the value in the parameter.
* toString():
	+ Takes in no parameters and returns a string.
	+ The string returned is in the following format:
		- name (yearOfRelease): $price

In your main method, prompt the user for the name of a book and a release date. Create a Book object with that information. Then, prompt the user for a price, and use the object’s setPrice() to set the Book object’s price. Finally, call the Book object’s toString() and print what is returned.

Please note the following:

* When asking for a price, keep prompting the user until they enter a price that is greater than 0.
* When asking for a yearOfRelease, keep prompting the user until they enter a number that is between 1500 and 2024.
* When asking for a name, keep prompting the user until they enter a name that isn’t empty.
* You can assume the user will never enter information of the wrong type (e.g.: when asking for a yearOfRelease, you can assume the user will only enter integers).

Sample output (user input in **bold**):

Enter a name for the book:

Name cannot be empty.

Enter a name for the book: **Gulliver's Travels**

Enter a year of release for the book: **1499**

Year of release must be between 1500 and 2024.

Enter a year of release for the book: **1726**

Enter a price for the book: **-200.0**

Price must be a positive number.

Enter a price for the book: **0**

Price must be a positive number.

Enter a price for the book: **42.99**

Gulliver's Travels (1726): $42.99

**Submit your file to Gradescope as Lab13A.java and Lab13B.java**