AP Computer Science

mr Hanley

Assignment 5/101_2/5_8/5_{16} Version: 3.0

Last Updated: 11/7/2017 1:41 PM

Binary

					🟦		🟦

Ones Comp

🟦	🟦	🟦	🟦	🟦		🟦	

Twos Comp

🟦	🟦	🟦	🟦	🟦		🟦	🟦

BoxPrint and Console Input Verifiers

* Dynamic Memory *Big O Notation*Stacks *Extreme Programming*Selection Sort*Insertion Sort*Waterfall Model

String*Arrays*ArrayList*Client Server*Artificial Intelligence*Inheritance*Files*Video Games*Short circuit evaluation*

Let's create something USEFUL, yes I know I said it, something you can actual USE again!

This project will be spread out across 4 different classes

CWHUtilities or BLPUtilities depending upon your intials	**This guy will have 2 static methods in it!** **public static void outputSquareRoots()** **and public static void outputBoxStr(String message)**
Assign5Tester	**This class is going to have a main method with a menu** **The menu is going to have the following options** **1 = Output Square Roots from 1-100** **2 = BoxPrint Something Nifty** **3 = Use Verifiers for 3 examples** **0 = exit** **This menu is going to call the methods and objects from the other classes**
DoubleVerifier	**This class has a Instance Vars, a Constructor and a readAndVerifyMethod**
IntVerifier	**This class has a Instance Vars, a Constructor and a readAndVerifyMethod**

Let's build your Tester First
public class Assign5 see my website

Parts 1 and 2 will be static methods in a class called "CWHUtilities" if your initials are CWH or "BLPUtilities" if your initials are BLP.

1. Write a loop that will print out the square roots of the numbers from 1 to 1000. Use a tab in between the numbers and make a table

   ```
   1	1
   2	1.4142135623
   …
   1000	31.622776601
   ```

   public static void squareRoots(){ … }

2. Write a method that takes in a String and prints a box around it
   The string can be between 1 and 20 characters
   Scale the box accordingly

For example
public static void outputBoxedStr(String message) { …}

outputBoxedStr("Hi!");
Since there are 3 characters in the message, you will place a row of 3 characters + 1 space before and after(2 total) + 2 leading @'s and 2 following @'s (4 total)= 3 + 2 + 4 = 9 @'s to form the top of the box
Then print out 2 @'s, fill the middle with spaces and then 2 more @'s
Then print 2 @'s + a space followed by the message followed by a space and 2 @'s to finish
Then print out 2 @'s, fill the middle with spaces and then 2 more @'s
Follow up with 9 @'s on a separate line to finish the bottom of the box
(Change in font below to a mono spaced font == each character is same width)

```
@@@@@@@@@
@@     @@
@@ Hi! @@
@@     @@
@@@@@@@@@
```

Another example
outputBoxedStr("Name: Brianna");

```
@@@@@@@@@@@@@@@@@@
@@              @@
@@ Name:Brianna @@
@@              @@
@@@@@@@@@@@@@@@@@@
```

If a String > 20 characters comes in, force only the first 20 characters to get printed. You can take a substring of the existing String by using

message = message.substring(0,20);
emphasis for Manan Jain (2019)

 //first you want, first you don't want

Another example

outputBoxedStr("The Philadelphia Flyers");

```
@@@@@@@@@@@@@@@@@@@@@@@@@@
@@                      @@
@@ The Philadelphia Fly @@
@@                      @@
@@@@@@@@@@@@@@@@@@@@@@@@@@
```

From a separate class, you now can do….

```
public static void main(String[] args) {
   outputBoxStr("Assignment #1");

}
```

3. Make 2 input verifier classes to make sure numeric inputs are in range
   Your first class should be called DoubleVerifier
   Your second class should be called IntVerifier
   Your classes should accept in a Scanner reference, a high value, a high value, a boolean if the low value is inclusive, a boolean if the high value is inclusive and an Clip to be played if the value is out of range.

In addition, create a method called **readAndVerify()** that prompts the user to enter in a value and checks to make sure its within value.
If the user enters in a number out of range or an alpha-numeric character, reject the input and play the Clip that was passed in. Give them an error message and force them to type it in until an acceptable value is entered.
Example,

```
public class DoubleVerifier {

  //This is a constructor
  public DoubleVerifier(Scanner sc, double lo, boolean loInc,
     double hi, boolean hiInc, Clip eSnd) {

  }
  public double readAndVerify() {
    //Reads in a value using the Global Scanner variable provided to the
   //Constructor
   //If the value is out of range, plays the Clip error sound and
      prompts for input again until in range
```

```
 }

 //Global Variables
 private Scanner keyboard;
 private double low, high;
 private boolean highInc, lowInc;
 private Clip errorSnd;
}
```

------EXAMPLE OF TIDBIT COMPUTER STORE USING THE VERIFIERS------

//Let's suppose these are the valid ranges for tidbit

Computer Cost 0 < cost <= 12000
Interest Rate(suppose it is a real number) 0 < rate <= .20
Down Payment(suppose it is a real number 0= < rate <= .50

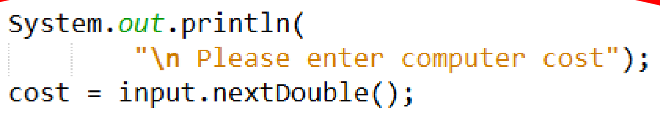
Here's how to use this class in your TidBitComputerStore as follows;

```
        Scanner input = new Scanner(System.in);
        //Sound Clips
        Clip bombSnd;      //Clips to be played
        //Load up sound file
        bombSnd = null;
        File bombSndF = new File("sounds/Explosion.wav");//folder in project
        try {
           bombSnd = AudioSystem.getClip();
           bombSnd.open(AudioSystem.getAudioInputStream(bombSndF));
        } catch (Exception e) {
           System.out.println(e);
        }
        //For the cost 0 is not ok but 12000 is
        IntVerifier costIntVer = new IntVerifier(input, 0, false, 12000, true, bombSnd);

        //For the rate we use a double verifier for fun, 0 not OK .2 is ok
        DoubleVerifier annualRateDlbVer = new DoubleVerifier(input, 0, false, .2, true,
bombSnd);
```

//For the downpayment, let's use a Double Verifier for %, 0 ok and .5 OK
DoubleVerifier downPayDlbVer = new DoubleVerifier(input, 0, true, .5, true, bombSnd);

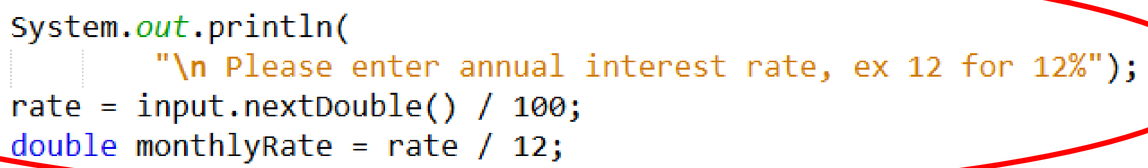
Replace this with:



```
System.out.println(
        "\n Please enter computer cost");
cost = input.nextDouble();
```

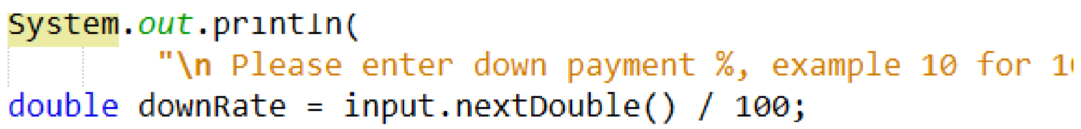
System.out.println("Please enter cost 0 < cost <= 12000");
cost = costIntVer.readAndVerify();

Replace this with:



```
System.out.println(
        "\n Please enter annual interest rate, ex 12 for 12%");
rate = input.nextDouble() / 100;
double monthlyRate = rate / 12;
```

System.out.println("Please enter rate 0 < rate <= .2");
rate annualRateDlbVer.readAndVerify();
rate = rate /12;



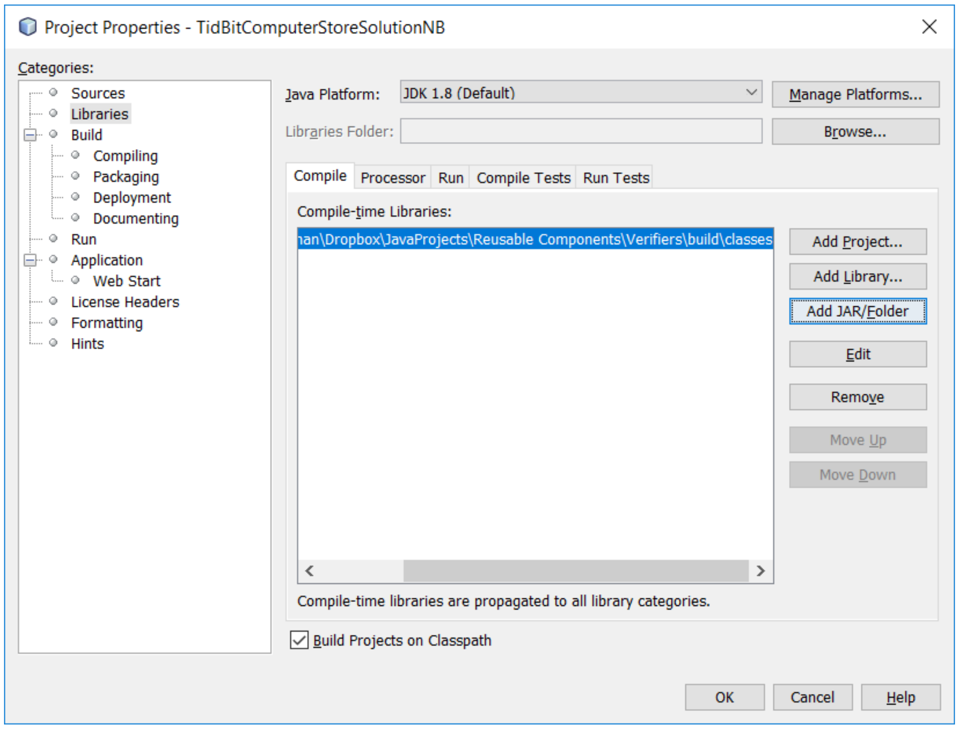
```
System.out.println(
        "\n Please enter down payment %, example 10 for 1
double downRate = input.nextDouble() / 100;
```

System.out.println("Please enter down payment 0 <= down payment <= .5");
downPayment = annualRateDlbVer.readAndVerify();

Tweak the rest of your tidbit to work with decimals as rates instead of integers.

How do I make sure that Tidbit can see assignment 5 classes?

Choose File...Project Properties
Choose Libraries...Add Jar/Folder and find the Assignment 5 build/classes folder


Project Properties - TidBitComputerStoreSolutionNB

Categories:
- Sources
- Libraries
- Build
  - Compiling
  - Packaging
  - Deployment
  - Documenting
- Run
- Application
  - Web Start
- License Headers
- Formatting
- Hints

Java Platform: JDK 1.8 (Default)

Libraries Folder:

Compile | Processor | Run | Compile Tests | Run Tests

Compile-time Libraries:

```
ıan\Dropbox\JavaProjects\Reusable Components\Verifiers\build\classes
```

Compile-time libraries are propagated to all library categories.

☑ Build Projects on Classpath

```
public static void main(String[] args) {
     Scanner input = new Scanner(System.in);
     Clip bombSnd;//, crashSnd;      //Clips to be played

       //Prepare an Audio File for the Verifiers, let's use the bomb sound
       //Load up all sound files
       File bombSndF = new File("sounds/Explosion.wav");

       //Load up sound files
       bombSnd = null;

       try {
          bombSnd = AudioSystem.getClip();
          bombSnd.open(AudioSystem.getAudioInputStream(bombSndF));
       } catch (Exception e) {
          System.out.println(e);
       }

      DoubleVerifier dv1 = new DoubleVerifier(input, 1, true, 9.5, true, bombSnd);
      double x = dv1.readAndVerify();  //forces value to be between 1 and 9.5 inclusive
}
```

In order to properly handle someone typing in letters or other symbols that are not allowed, use the following code in your read and verify method

```
double inVal=0;
try {
            inVal = keyboard.nextDouble();
                 //more logic here to decide range
                 //logic
}catch (InputMismatchException e) {
            System.out.println("@@@@@@@@@@@@@@@@@@@@");
            System.out.println(" Bad Character");
            System.out.println("@@@@@@@@@@@@@@@@@@@@");
            errorSnd.setFramePosition(0);
            errorSnd.start();
}
finally {
        keyboard.nextLine();
}
```

Project Name	Assign 5 – Good Stuff
Class 1 Name	CWHUtilities
Class 2 Name	DoubleVerifier
Class 3 Name	IntVerifier
Class 4 Name	Assign5Tester

Rubric	
Print out square roots	10
outputBoxedString	25
DoubleVerifier constructor	15
readAndVerify	40
IntVerifier constructor	15
readAndVerify for IntVerifier	20
TOTAL	125