

1 Before Getting Started on the Exercises

The exercises in this homework are similar to the examples we have been looking at in class. They are of the form

$$\text{Input} \implies \text{Do Calculations} \implies \text{Output}$$

2 Exercises

For each of the following create a new Java project with an appropriate name and then write a program that solves the given problem.

Remember:

- Shift+Ctrl+F to format the program, or Shift+Command+F on the Mac.
- The standard comments of at the top,
 - Name
 - Date
 - Description

For each program, you will be submitting the java code file through MyClasses, as you did before. I also want either a Microsoft Word docx file, LibreOffice Writer odt, or a text file (which you can create with NotePad++) which contains,

- The answers to the three main questions:
 - WANT:** What do I want as the final result of the program?
 - HOW:** What is the calculation I will need to do?
 - NEED:** What do I need to complete this calculation?
- The answers above reformatted into an algorithm.
- Output of at least three runs of each program on different data inputs.

This docx, odt, or text file is to be uploaded to MyClasses as well. You can copy and paste output from the Eclipse console area to the word or text program.

2.1 Programming Exercises

1. Write a program that will convert miles to feet. Sample runs are below. The user input the 4 and the 10.12345.

```
Input the number of miles: 4
There are 21120.0 feet in 4.0 mile(s).
```

```
Input the number of miles: 10.12345
There are 53451.816 feet in 10.12345 mile(s).
```

2. Write a program that will convert feet to miles. Sample runs are below. The user input the 5280 and the 1000000.

```
Input the number of feet: 5280
There are 1.0 miles in 5280.0 feet.
```

```
Input the number of feet: 1000000
There are 189.3939393939394 miles in 1000000.0 feet.
```

3. Write a program that will convert feet to feet and inches. Sample runs are below. The user input the 12.34567 and the 123.321.

```
Input the number of feet: 12.34567
12.34567 feet = 12 feet 4.14804 inches.
```

```
Input the number of feet: 123.321
123.321 feet = 123 feet 3.8519999999999754 inches.
```

Note the the number of feet in the output has no decimal point. To do this calculation you need to separate the values on the left and right of the decimal point of the input. Consider the 12.34567 input, this is 12 feet (the number on the left of the decimal) and 0.34567 feet (which we will convert to inches). In Java, and a few other languages like Java (such as C and C++) you can extract the portion to the left by using what is called a *cast*. In Java if the variable *x* is holding a decimal number we can extract the portion to the left by placing `(int)` in front of it. Consider the following program and its output.

```
1 public class HW01CastExample {
2
3     public static void main(String[] args) {
4         double x = 12.12345;
5         int y = (int) x;
6         System.out.println(x);
```

```
7         System.out.println(y);
8         System.out.println(x - y);
9     }
10 }
```

```
12.12345
12
0.123450000000000006
```

Also notice that the portion to the right of the decimal is the difference between the decimal value and the extracted integer value. In addition, the last output should have been 0.12345 and not 0.123450000000000006, this is round-off error which happens frequently. We will discuss later in the course why this happens, for now we will not worry about a little round-off error.

4. In the game of Craps the roller rolls two six-sided die and the sum of the die rolls determines winning, losing, or other rolls. Write a program that simulates rolling a pair of six-sided dice and outputs each die value and the sum of the values. You can simulate rolling one die by choosing one of the integers 1, 2, 3, 4, 5, or 6 at random. The `Math.random()` command gives us a decimal number between 0 and 1 (not including 1). To convert this to a random number generator that returns 1, 2, 3, 4, 5, or 6 we can use the following expression,

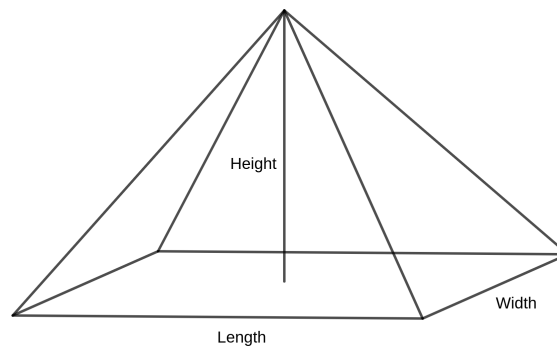
```
(int) (Math.random()*6) + 1
```

Multiplying `Math.random()` by 6 gives a decimal number between 0 and 6 (not including 6), then the `(int)` in front is a cast which converts and truncates the double to an integer between 0 and 5, that is, 0, 1, 2, 3, 4, 5. Finally, adding 1 gives us the possible output of 1, 2, 3, 4, 5, or 6.

Output of this program should look like the following. Note that there is no user input needed.

```
The first die is a 3
The second die is a 5
Your total roll is 8
```

5. In a pyramid the base is a rectangle with sides of length, Length and Width, the other faces are triangles from each of the four sides of the base to a point above the base. The distance (perpendicular distance) from the top point to the base is of length, Height. The volume of a pyramid is one third the product of the length, width, and height.



Write a program to calculate the volume of a pyramid. Sample runs are below.

```
Input the Length: 3
Input the Width: 4
Input the Height: 6
The Volume is = 24.0
```

2.2 Challenge Exercise

Challenge Exercises are optional, they will be graded as extra credit.

1. Write a program that helps the cashier at a store count back change. The program should ask for the amount of money to give back in change for a purchase (for example, \$5.47) and then output the number of twenties, tens, fives, one dollars, quarters, dimes, nickels, and pennies. You do not need to go higher than a twenty dollar bill in change but the program should handle amounts like \$125.24. The output should always use the highest denomination possible, an output of 12524 pennies does not solve the problem. For example, for an input of \$5.47, the program should output 1 five, 1 quarter, 2 dimes and 2 pennies. For the input of \$125.24, the program should output 6 twenties, 1 five, 2 dimes and 4 pennies.

You will need the remainder operator for this calculation, recall that $25/4$ is integer division and 4 into 25 gives a quotient of 6 with a remainder of 1 . So the expression $25/4$ will return 6 , to get the remainder we use $25 \% 4$. Type in and run the following program a few times with different input get a feel of how $/$ and $\%$ work with integers.

```
1 import java.util.Scanner;
2
3 public class HW001_Tester {
4
5     public static void main(String[] args) {
6         Scanner keyboard = new Scanner(System.in);
7         System.out.print("Input the Numerator: ");
8         int num = keyboard.nextInt();
9         System.out.print("Input the Denominator: ");
10        int den = keyboard.nextInt();
11
12        int quo = num / den;
```

```
13         int rem = num % den;
14
15         System.out.println("The quotient of " + num + "/" + den + " is " + quo);
16         System.out.println("The remainder of " + num + "/" + den + " is " + rem);
17     }
18 }
```

Here is a hint on how to use the integer division and remainder operations to create the change program. Let's say that the amount of change is \$125.24 and we will be using a twenty dollar bill as the maximum denomination. \$125.24 is 12524 cents, which is a number without any decimal place, an integer. There are 2000 cents in a twenty dollar bill so if we calculate $12524 / 2000$ we get 6 and if we calculate $12524 \% 2000$ we get 524. The 6 is the number of twenties and 524 is the number of cents left over that we still need to change. Now if we calculate $524 / 1000$ and $524 \% 1000$ we get 0 and 524 respectively. Note that 1000 is the number of cents in a ten. The results tell us that we give back 0 tens and we still have 524 cents to change. You probably see where this is going but for completeness we will finish out the calculation. Calculate $524 / 500$ and $524 \% 500$ we get 1 and 24 respectively, so one 5 is given back as change. Calculate $24 / 100$ and $24 \% 100$ we get 0 and 24 respectively, so no ones are given back as change. Calculate $24 / 25$ and $24 \% 25$ we get 0 and 24 respectively, so no quarters are given back as change. Calculate $24 / 10$ and $24 \% 10$ we get 2 and 4 respectively, so two dimes are given back as change. Calculate $4 / 5$ and $4 \% 5$ we get 0 and 4 respectively, so no nickels are given back as change and we also see that we have 4 pennies to give back. So converting the user's decimal input to the integer number of cents to be returned is a good first step. Also, be careful with the number of cents, round-off error could give you an error. You may also want to look up the `Math.round(x)` function.

For this exercise use only the basic arithmetic operations and Math functions. Even if you have seen more fancy things like decision and repetition structures you may not use them.