This document gives you a set of exercises in writing Java snippets using the tool *jshell*. Jshell is similar to the Python interpreter or the Matlab command window. It's a great way to experiment with basic Java syntax. However, as your projects get bigger, you’ll want a more structured and reproducible way of testing them, which is why we’ll use the JUnit framework built into Eclipse for most of this course.

The exercises will help you learn the syntax and semantics of:

method declarations  
 method calls  
 return statement  
 printing  
 basic operations, including the remainder operation %  
 if- and if-else statements  
 conditional expression  
 for-loop and while-loop.

**How to learn about jshell**

To find out about the basics of jshell, type jshell into the JavaHyperText (JHT) filter field and read the pdf file about the basics of jshell.

**Exercises**

Now that you have done that, complete the following tasks by typing code into jshell (you will remember the syntax better if you type, rather than copy-paste). We recommend saving a copy of your working code, along with notes about any mistakes you made along the way, to a document that you can reference later.

1. Copy the following function into jshell:

/\*\* = the product of x with itself. \*/

int square(int x) {

return x\*x;

}

Try calling your function by typing square(3) at the prompt (which should return 3\*3 = 9).

2. Write a function named "product" that returns the product of two integers. Confirm that it works by typing product(3, 4) at the prompt and checking its answer. Try a few examples of your own.

3. Read the JHT entry on return .

4. Write a procedure named printRemainder that prints the remainder when its first integer parameter is divided by its second. A procedure uses void instead of a return type like "int" in the method in Exercise 1. The body can be a single statement System.out.println(...); where you replace the "..." by the expression that computes the remainder using operator %. If you are unsure how to do this, look at JHT entry % .

5. We're going to ask you to write five different methods to compute the maximum of two values in order to introduce you to the if-statement, the if-else statement, the conditional expression (ternary operator), the return statement in a procedure, and the block. To start, read the pdf file on JHT entry if-statement .

5A. Write a function max1 that returns the larger of its two integer parameters. Use an if-else statement to determine whether the first parameter is larger than the second. The then-part and else-part should be return statements.

5B. Write a function max2 that returns the larger of its two integer parameters. The body should consist of (1) an if-statement (not an if-else statement) whose then part is a return statement and (2) a return statement. This manner of not using an else-statement can often simplify code.

5C. Write a function max3 that returns the larger of its two integer parameters. The body should be a single return statement whose expression is a conditional expression. If need be, look at the JHT entry ? . Yes, just type a query (question mark) into the Filter Field.

5D. Write this procedure max4, which prints the larger of its two integer parameters:

void max4(int x, int y) {

if (x >= y) {

System.out.println("max is " + x);

return;

}

System.out.println("max is " + y);

}

Put these calls into jshell to see whether the procedure works: max4(5, 6) max4(7, -2).

Here are two important points about this procedure.

(1) The then-part consists of a *block* {...} . A block is needed because the then-part has two statements in it. Whenever the then-part or else-part requires several statements, *aggregate* them using a block.

(2) This example shows how to use a return statement in a procedure. It's execution terminates execution of the method body; nothing more happens after that return statement is executed.

5E. Write this procedure max5, which prints the .. THIS WILL swap x and y if y is larger, thus intrducing the local variable.

Write a function named `max2` with the same specification as `max`, but use a to implement it with a single return statement, instead of using an if-statement.

5. Implement the following function using a while loop:

/\*\* = the first power of 2 larger than or equal to threshold. \*/

int powerCeiling(int threshold) {

int pow = 1; // = 2^0

// TODO: Multiply pow by 2 until it satisfies the requirement of the function’s return value

return pow;

}

6. Write a for-loop to compute the sum of all the integers from 1 to 100.

7. Complete the following function:

/\*\* Compute the mean of xs, then return an array containing the difference between each element in xs and its mean. \*/

double[] zScores(double[] xs) {

// TODO

}