Worksheet for Lab #2 Ex2: Square-free Integer

http://www.comp.nus.edu.sg/~cs1010/labs/2016s1/lab2/controlstructures.html

Task Statement

Read 4 positive integers *lower1*, *upper1*, *lower2*, *upper2*, determine which of the 2 ranges [*lower1*, *upper1*] and [*lower2*, *upper2*] contains more square-free integers, and report the number of square-free integers in the range that has more square-free integers.

A square-free integer is a positive integer not divisible by any square number except 1.

For this exercise, let's try a bottom-up design.

Step 1

We should have a function that takes in an integer and determines if it is square-free or not. Let's call this function **is_square_free()**. Complete the table about **is_square_free()** below.

Return type	***Parameter***		***Precondition***
	Type	***Name***	

Step 2

Write out your algorithm for **is_square_free()**. What control structures does it use?

Algorithm:

_____ is_square_free(_____) {

}

The algorithm uses (circle all appropriate answers): Sequence / Selection / Repetition

Step 3

You may practise **Incremental Coding** (refer to Unit 7: Testing and Debugging) here. The correctness of your final program depends on the correctness of **is_square_free()** function. So it is worthwhile checking that the function works before you proceed.

How do you think you can test the **is_square_free()** function?

Step 4

After you have ensured that **is_square_free()** is perfect, how do you use it to solve the task? Since you are going to do the same thing (count the number of square-free integers) for TWO ranges, you should write a function for it. Let's call it **count_square_free()**.

Complete the table about **count_square_free()** below.

Return type	***Parameters (types and names)***	***Precondition***

Step 5

Write out your algorithm for **count_square_free()**.

Algorithm:

```
_______ count_square_free( ____________________________ ) {




}
```

Step 6

Complete your main() function and test your program thoroughly before you submit it to CodeCrunch.