CH II IntegerUtils
COSC 2329 Component-Based Programming
Assigned: Wednesday, February 7, 2018
Due: Tuesday, February 13, 2018 11:59 PM
Late Deadline: Thursday, February 15, 2018 6:00 PM

Integer Utilities

Your task is to implement the methods specified in the following IntegerUtils interface. As of Java 1.8, interfaces can now have static methods with code. For this assignment, we aren't going to worry about whether using interfaces, rather than classes, for a group of utility methods is desirable.

IntegerUtils interface (next four pages)

```
package utils;

public interface IntegerUtils_Skeleton
{
	//EX: isEven(3) returns false
	//EX: isEven(0) returns true
	//EX: isEven(-2) returns true
	public static boolean isEven(int k)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}

	//EX: isOdd(3) returns true
	//EX: isOdd(0) returns false
	//EX: isOdd(-2) returns false
	public static boolean isOdd(int k)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}

	//EX: isPrime(-3) returns false
	//EX: isPrime(0) returns false
	//EX: isPrime(1) returns false
	//EX: isPrime(23) returns true
	public static boolean isPrime(int k)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}
//CONTINUED ON NEXT PAGE
```

```
        //EX: greatestCommonFactor(-2, 5) returns 1
        //EX: greatestCommonFactor(24, 32) returns 8
        //EX: greatestCommonFactor(289, -17) returns 17
        //EX: greatestCommonFactor(30, 42) returns 6
        //EX: greatestCommonFactor(-21, -21) returns 21
        //EX: greatestCommonFactor(0, 0) is undefined
        public static int greatestCommonFactor(int m, int n)
        {
            throw new RuntimeException("NOT IMPLEMENTED!");
        }

        //part of pre: k != 0
        //part of pre: maximum >= 0
        //part of pre: k <= maximum
        //EX: getGreatestConstrainedMultiple(7, 50) returns 49
        //EX: getGreatestConstrainedMultiple(2, 100) returns 100
        //EX: getGreatestConstrainedMultiple(3, 19) returns 18
        //EX: getGreatestConstrainedMultiple(10, 789) returns 780
        //EX: getGreatestConstrainedMultiple(-2, 46) returns 46
        //EX: getGreatestConstrainedMultiple(-52, 5) returns 0
        public static int getGreatestConstrainedMultiple(int k, int maximum)
        {
            throw new RuntimeException("NOT IMPLEMENTED!");
        }
//CONTINUED ON NEXT PAGE
```

```
	//EX: getIntegerH(7, 3) returns 23, since 23/3 = 7 and 23 % 5 = 3
	//EX: getIntegerH(15, 0) returns 45 since 45/3 = 15 and 45 % 5 = 0
	public static int getIntegerH(int h_q_3, int h_r_5)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}

	//EX: getMaximum({2, 4, 6, 8, 10}) returns 10
	//EX: getMaximum({1, 17, -22, 48, 19}) returns 48
	//EX: getMaximum({15}) returns 15
	public static int getMaximum(Set<Integer> integerSet)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}

	//EX: getMinimum([2, 4, 6, 8, 10]) returns 2
	//EX: getMinimum([1, 17, -22, 48, 19]) returns -22
	//EX: getMinimum([15]) returns 15
	public static int getMinimum(int[] intArray)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}

	//EX: isSorted([2, 4, 6, 8, 10]) returns true
	//EX: isSorted([1, 17, -22, 48, 19]) returns false
	//EX: isSorted([15]) returns true
	public static boolean isSorted(int[] intArray)
	{
		throw new RuntimeException("NOT IMPLEMENTED!");
	}
//CONTINUED ON NEXT PAGE
```

getMinimum(null) → "!!"
getMinimum([]) → "!!"

```
    public static int NO_MATCH = -1;
    //part of post: (rv == NO_MATCH) || (intArray[rv] == target)
    public static int getSmallestIndexOfMatch(int[] intArray, int target)
    {
        throw new RuntimeException("NOT IMPLEMENTED!");
    }

    //EX: reverse(92) returns 29
    //EX: reverse(48) returns 84
    //EX: reverse(0) returns 0
    //EX: reverse(-2) is undefined
    public static int reverse(int k)
    {
        throw new RuntimeException("NOT IMPLEMENTED!");
    }

    //EX: sumthing(7) = 7
    //EX: sumthing(14) = 5
    //EX: sumthing(29) = 2
    public static int sumthing(int k)
    {
        throw new RuntimeException("NOT IMPLEMENTED!");
    }
}
```

Assignment

Complete the IntegerUtils interface above. You must first change the name of the interface (and file) from IntegerUtils_Skeleton to IntegerUtils_*LastName*, where *LastName* is your last name. For example, my submission would be named IntegerUtils_Kart and would be in the file IntegerUtils_Kart.java. You must remove the "throw new RuntimeException("NOT IMPLEMENTED!");" line from each method body and replace it with a correct method body. Do not change one byte of any method signature in the interface — be sure to ask if you don't know exactly what this means. Also note that Java 1.8 allows static methods (with bodies) to be included in interfaces, so, if the compiler starts yelling at you then make sure you are using Java 1.8.

Deliverables

- One .zip file uploaded to Canvas (under CH II, IntegerUtils, or similar), including:
  - IntegerUtils_*LastName*.java (with *LastName* replaced by your actual last name)
  - Any supporting files that you make (e.g., ArrayUtils_*LastName*) [I'm not expecting any for this assignment]

Rules

- Your interface must be in the package named 'utils' (all lowercase, without the quotes).
- You are not allowed to change any of the method signatures in the interface. Yes, even if:
  - The compiler is yelling at you
  - Eclipse told you to change something via an automatic fix
  - Your interface is better than the one listed above
- Make sure that every class, interface, and enum is properly documented (even if you are copying it from this sheet or lecture notes).
- Don't hand in the first draft that works. Refine it.
- Make sure you test your code. Then test it a lot more.
- Test both "middle-of-the-road" cases as well as corner cases and "degenerate" cases.
- Ensure that your files follow the naming convention under **Deliverables**.

Notes

- I am most concerned with clarity (i.e., can I understand your algorithm using only appropriate cognitive effort on my part). I am not worried about efficiency here!
- Ensure that you are using Java 1.8.

WARNING!!!

- **This specification may be misleading, inconsistent, or incomplete! Part of the assignment is to read the assignment early, think about it, and ask any clarifying questions!**