

Sample Hour Exam

Background to Problem 1: The greatest common divisor (gcd) of two non-zero integers a and b is the largest integer d that divides both a and b . If d is the gcd of a and b we use the notation $(a, b) = d$.

For example, the gcd of 15 and 25 is 5; $(25,15) = 5$. The gcd of 49 and 63 is 7; $(63,49) = 7$.

There is an algorithm for determining the gcd of a and b that is based on repeated applications of the idea that if $a \geq b > 0$ then

$$a = q b + r \text{ with } 0 \leq r < b.$$

In words, q is the quotient and r is the remainder you get when you divide a by b .

This algorithm for the three cases $(63,49)$, $(96,57)$ and $(55,27)$ works as follows:

$63 = 1 \cdot 49 + 14$	$96 = 1 \cdot 57 + 39$	$55 = 2 \cdot 27 + 1$
$49 = 3 \cdot 14 + 7$	$57 = 1 \cdot 39 + 18$	$27 = 27 \cdot 1$
$14 = 2 \cdot 7$	$39 = 2 \cdot 18 + 3$	\Rightarrow
\Rightarrow	$18 = 6 \cdot 3$	$(55,27) = 1$
$(63,49) = 7$	\Rightarrow	
	$(96,57) = 3$	

1. Complete the following code for a method for finding the gcd of two non-zero integers

```
public static int gcd(int a, int b)
{
    int A = Math.abs(a);
    int B = Math.abs(b);

    int c = Math.max(A,B);
    int d = Math.min(A,B);
    int r = c%d;

    while( r != 0)
    {
        // <- Enter the appropriate code here
    }
    return ;
    // <- And here
}
```

2. Two integers whose gcd is 1 are said to be relatively prime. For example 1 and 2 are relatively prime 1 and 3 are relatively prime. The integers that are between 1 and 15 and relatively prime to 15 are

1 2 4 7 8 11 13 14

Suppose you have the gcd() method of problem 1 written and working. Complete the following code for a class for determining all the integers that are ≥ 1 and $< n$ and relatively prime to N:

```
public class RelativelyPrime
{
    int total;
    int numer[] = new int[total];

    public RelativelyPrime(int N)
    {
        int count = 0;
        // count the numbers relatively prime to N below:

        // now save them in array
        total = count;
        numer = new int[total];
        count = 0;

    }

}

} // end RelativelyPrime(int N)

public static int gcd(int a, int b)
{
    //From problem 1
    // You don't have to fill this in
    // But the gcd() method will be used above
}

} // end class RelativelyPrime
```

3. Suppose that you completed the code for the class in problem 2, including the code for the gcd() method, compiled it, and found that it works.

Suppose next that you wrote the following application:

```
public class TestRelPrime
{
    public static void main(String args[])
    {
        System.out.println("gcd(35, 49) = "+RelativelyPrime.gcd(35,49));
    }
}
```

Some comments:

(1) The text file TestRelPrime.java would be in the same directory as the RelativelyPrime.class file.

(2) RelativelyPrime has not been instantiated.

Some questions:

Will this compile? Will it run?

What, if it does run, is the output?

If it does run, what feature of java is being used to enable it to run without it being instantiated?

4. Suppose a PrintOut(int N) routine is added to the class RelativelyPrime. Thus, in outline, the final program looks like this:

```
public class RelativelyPrime
{
    int total;
    int numer[] = new int[total];

    public RelativelyPrime(int N)
    {
        // From Problem 2
        // You don't have to fill this in

    }

    }// end RelativelyPrime(int N)

    public static int gcd(int a, int b)
    {
        //From problem 1
        // You don't have to fill this in
    }

    public void PrintOut(int N)
    {
        // This prints out the integers between 1 and N
        // that are relatively prime to N
        // You don't have to fill this in
    }

}

} // end class RelativelyPrime
```

Suppose that the details to this have been added, and the class has been debugged, so is working.

Write a new version of the application TestRelPrime, that uses the above class and will print out the integers between 1 and 99 that are relatively prime to 99:

```
public class TestRelPrime
{
    public class void main(String args[])
    {

    }

} // end //

} // end class TestRelPrime
```

5. Write a new version of the application `TestRelPrime` that, using the class of problem 4, will calculate the number of rational fractions a/b where

$0 < a/b < 1$,
a and b are relatively prime,
and $2 \leq b < 999$

```
public class TestRelPrime
{
    public class void main(String args[])
    {

        }// end main(String args[])

    }// end class TestRelPrime
```