

Name(s) _____

Please sit in teams of 2. At least one person on each team needs to set up an account on code.org
<http://code.org> Click "Sign in" to create your own account and then join our class using the code KLCXNY

Activity Guide - Multiplication + Modulo

Summary

In this activity you'll multiply numbers as input into the modulo operation and explore some interesting properties that relate to cryptography.

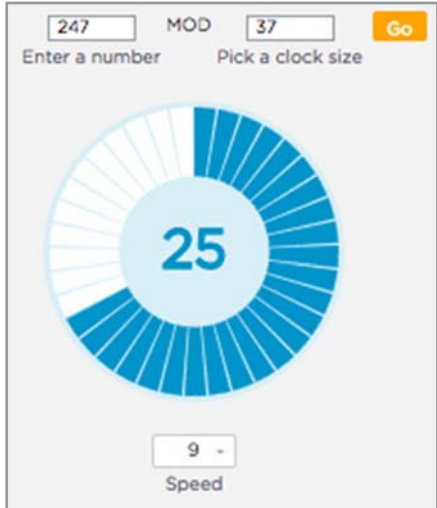
Goal: Understand how multiplication + modulo can be used to make computationally-hard-to-crack encryption.

- Tools:** - <https://studio.code.org/s/csp4-2019/stage/9/puzzle/4>
- **Calculator:** you probably want a calculator handy for multiplying big numbers
 - **The "Mod Clock" widget** in code studio (pictured at right)

Assumption: You have been introduced to the modulo operation and the "clock" analogy for it.

Step 1: Experiment with the Mod Clock

Goal: familiarize yourself with properties of the Modulo operation



Get your feet wet - play

- Try inputting different values into the mod clock for *both* the "number" and the "clock size".
- Try big numbers and small numbers for both

Questions:

1. Using a clock size of 50, write a list of 5 numbers that produce a result of 0.
2. With clock size of 50 how many total numbers are there that produce a result of 0? (If the list is short, write it out. If the list is long, describe a pattern of what the numbers are).
3. Using a clock size of 13, can you find a number to input that produces a *result* of 13? (If so, what is it? If not, why not?)
4. Using a clock size of 13, find the answers to the following:

1 MOD 13	
10 MOD 13	
100 MOD 13	

1,000 MOD 13	
10,000 MOD 13	
100,000 MOD 13	

Are these results surprising or interesting? Why or why not?

Step 2: Toward encryption - Use multiplication to produce inputs

Experiment - Small changes to inputs, big changes to outputs.

Using a clock size of 37, let's multiply two numbers (we'll call them A and B) to use as input, then make small changes to each while holding the other constant. We'll always use the formula $A * B \text{ MOD } M$. We'll start with $A=20$ and $B=50$ and $M=37$. So here is the first result...

$$20 * 50 \text{ MOD } 37 = 1$$

Now find in the rest of these values making small adjustments to A and B individually.

Use a calculator, if necessary, to compute $A * B$. Use the Mod Clock to compute the modulus of the result.

Increment A	
$21 * 50 \text{ MOD } 37$	
$22 * 50 \text{ MOD } 37$	
$23 * 50 \text{ MOD } 37$	

Increment B	
$20 * 51 \text{ MOD } 37$	
$20 * 52 \text{ MOD } 37$	
$20 * 53 \text{ MOD } 37$	

Result: What do you notice about these results? Is there a pattern? Could you predict the result of $25 * 50 \text{ MOD } 37$?

Experiment 2 - Guessing inputs is hard?: Using a clock size of 101, we'll give you the value of A and even hold the result of the modulo operation constant. **Your task:** find a value for B (the blank) that makes the math work out.

1)

$$2 * \underline{\quad} \text{ MOD } 101 = 1$$

2)

$$3 * \underline{\quad} \text{ MOD } 101 = 1$$

3)

$$4 * \underline{\quad} \text{ MOD } 101 = 1$$

Takeaways:

Solving modulus equations like $2 * \underline{\quad} \text{ MOD } 101 = 1$ is "hard" because you can't solve it like a typical equation. There are no easy patterns or shortcuts like other equations you might see in a math class. As you learned in step 1 (hopefully) there is an infinite list of single values for which $\underline{\quad} \text{ MOD } 101 = 1$. (The list is 1, 102, 203, 304, 405...etc). With multiplication, to solve $2 * \underline{\quad} \text{ MOD } 101 = 1$ you end up randomly guessing to find some number to multiply by 2 that gives you a result in that list.

Things get especially "hard" when you use a prime number as the clock size. Thanks to some special properties of prime numbers **with a prime clock size there's only one solution to each modulus equation**. You are guaranteed that there is the number less than the clock size itself, but there are still 100 different values you have to try. With a **brute force search** you could go through them all in a couple minutes. But what if the clock size were a 50-digit prime number?

Encryption!

Whenever you have a problem for which the only way to solve it is by random guessing or brute force search over a large range of values, you have a candidate for an encryption. Next you'll get to try it!

Your TA will lead you in an activity using the Public Key Crypto Widget.

<https://studio.code.org/s/csp4-2019/stage/9/puzzle/6>