College of Engineering & Applied Sciences

CSPB 2824

Discrete Sturctures

Cryptography Mastery Workbook

Taylor Larrechea

2023

Sections

Crpytography Mastery Workbook 2
Problem 1. 3
Problem 2. 5
Problem 3. 7
Problem 4. 10
Problem 5. 13
Problem 6. 15
Problem 7. 17
Problem 8. 20
Problem 9. 22
Problem 10 25

Mastery Workbook 7

Crpytography Mastery Workbook

I have neither given nor received unauthorized assistance.

Taylor James Larrechea

Problem 1

Problem Statement

Finding Modular inverses.

(a) Find 5 numbers x such that: $x \cdot 3 \textbf{ mod } 10 = 1$.

Notice that just as we have inverses in multiplication, for example, $3 \cdot \frac{1}{3} = 1$, we can have inverses of $3 \textbf{ mod } n$.

(b) Are modular inverses unique the same way multiplication inverses are unique? Answer why or why not ?

(c) What is the same about each of the x's you found in Question 1?

Solution - Part (a)

(a) Find 5 numbers x such that: $x \cdot 3 \textbf{ mod } 10 = 1$:

- $7 \cdot 3 \textbf{ mod } 10 = 21 \textbf{ mod } 10 = 1 \rightarrow x = 7$
- $17 \cdot 3 \textbf{ mod } 10 = 51 \textbf{ mod } 10 = 1 \rightarrow x = 17$
- $27 \cdot 3 \textbf{ mod } 10 = 81 \textbf{ mod } 10 = 1 \rightarrow x = 27$
- $37 \cdot 3 \textbf{ mod } 10 = 111 \textbf{ mod } 10 = 1 \rightarrow x = 37$
- $47 \cdot 3 \textbf{ mod } 10 = 141 \textbf{ mod } 10 = 1 \rightarrow x = 47$

Solution - Part (b)

(b) Are modular inverses unique the same way multiplication inverses are unique? Answer why or why not ?

For a given integer n, there is only one multiplicative inverse of n. As for modular inverses, there can be many different modular inverses as long the numbers that are being modded are relatively prime. So no, modular inverses are not unique the same way multiplication inverses are unique.

Solution - Part (c)

(c) What is the same about each of the x's you found in Question 1?

The x value that is found in part (a) when multiplied with 3, is relatively prime with 10. A pattern arrives in that each x that I found in part (a) is just 10 more than the previous x value. So, x has to generate a number that makes it relatively prime with 10.

Problem 1 Summary

Procedure

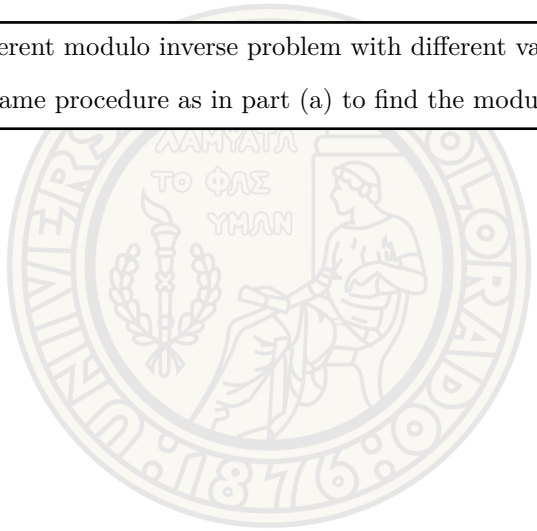
- For part (a), we need to find values for x such that $x \cdot 3 \textbf{ mod } 10 = 1$. This is done by trial and error.
- Answer the prompt.
- Answer the prompt.

Key Concepts

- A modular inverse is an integer that is multiplied by another integer where the modulo of that product with another integer is 1.
- Modular inverses are not unique in the way that multiplicative inverses are unique.
- For this given problem, each value of x is 10 greater than the previous values.

Variations

- We could be given a different modulo inverse problem with different values.
  - We would use the same procedure as in part (a) to find the modulo inverses.



Problem 2

Problem Statement

In ordinary multiplication, inverses always exist. Can you always find a modular inverse?

(a) Try $x \cdot 5 \textbf{ mod } 10 = 1$ and $x \cdot 2 \textbf{ mod } 10 = 1$

For which of the following do modular inverses exist? (Find an inverse x if you can. Use a Guess and Check method)

(b) $x \cdot 7 \textbf{ mod } 10 = 1$

(c) $x \cdot 26 \textbf{ mod } 13 = 1$

(d) $x \cdot 7 \textbf{ mod } 5 = 1$

(e) $x \cdot 5 \textbf{ mod } 21 = 1$

(f) $x \cdot 6 \textbf{ mod } 35 = 1$

(g) $x \cdot 7 \textbf{ mod } 13 = 1$

What is the rule for when Modular inverses exist?

Solution - Part (a)

(a) Try $x \cdot 5 \textbf{ mod } 10 = 1$ and $x \cdot 2 \textbf{ mod } 10 = 1$

- $x \cdot 5 \textbf{ mod } 10 = 1$ The gcd(10,5) = 5 and therefore a modular inverse does not exist.
- $x \cdot 2 \textbf{ mod } 10 = 1$ The gcd(10,2) = 2 and therefore a modular inverse does not exist.

Solution - Parts (b)-(g)

(b) $x \cdot 7 \textbf{ mod } 10 = 1$ $x = 3 \rightarrow 3 \cdot 7 \textbf{ mod } 10 = 21 \textbf{ mod } 10 = 1$. A modular inverse exists.

(c) $x \cdot 26 \textbf{ mod } 13 = 1$ The gcd(26,13) = 13 and therefore a modular inverse does not exist.

(d) $x \cdot 7 \textbf{ mod } 5 = 1$ $x = 3 \rightarrow 3 \cdot 7 \textbf{ mod } 5 = 21 \textbf{ mod } 5 = 1$. A modular inverse exists.

(e) $x \cdot 5 \textbf{ mod } 21 = 1$ $x = 17 \rightarrow 17 \cdot 5 \textbf{ mod } 21 = 85 \textbf{ mod } 21 = 1$. A modular inverse exists.

(f) $x \cdot 6 \textbf{ mod } 35 = 1$ $x = 6 \rightarrow 6 \cdot 6 \textbf{ mod } 35 = 36 \textbf{ mod } 35 = 1$. A modular inverse exists.

(g) $x \cdot 7 \textbf{ mod } 13 = 1$ $x = 2 \rightarrow 2 \cdot 7 \textbf{ mod } 13 = 14 \textbf{ mod } 13 = 1$. A modular inverse exists.

Synopsis

The rule for when modular inverses exist is in the equation $x \cdot a \textbf{ mod } b = 1$, gcd(b,a) must equal 1. a and b must be relatively prime. If a and b are relatively prime, then their Bezout coefficients are the modular inverses of the original equation.

Problem 2 Summary

Procedure

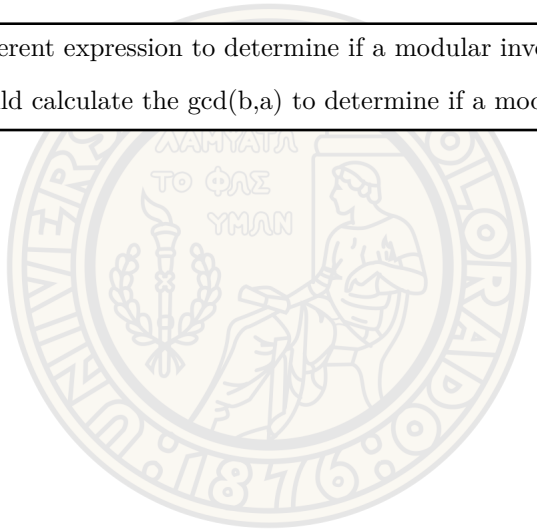
- In the formula
$$x \cdot a \textbf{ mod } b = 1$$
calculate the greatest common divisor (gcd) of gcd(b,a). If the gcd(b,a) = 1 then a modular inverse does exist.
- If the gcd(b,a) = 1, then find a modular inverse with trial by error.

Key Concepts

- In the formula
$$x \cdot a \textbf{ mod } b = 1$$
a modular inverse only exists if gcd(b,a) = 1.

Variations

- We could be given a different expression to determine if a modular inverse exists.
  - In this case we would calculate the gcd(b,a) to determine if a modular inverse does indeed exist.



Problem 3

Problem Statement

Work through #1 - 4 page 284. Summarize important ideas here that show you have studied this topic.

Solution - # 1

1. Show that 15 is an inverse of 7 modulo 26.
   - The greatest common divisor of 26 and 7 is

$$\begin{aligned} 26 &= 7 \cdot 3 + 5 \\ 7 &= 5 \cdot 2 + 1 \\ 5 &= 1 \cdot 5 + 0 \end{aligned}$$

   1. Since gcd(26,7) = 1 we can show that

$$15 \cdot 7 \textbf{ mod } 26 = 105 \textbf{ mod } 26 = 1$$

   15 is a modular inverse of 7 modulo 26.

Solution - # 2

2. Show that 937 is an inverse of 13 modulo 2436.
   - The greatest common divisor of 2436 and 13 is

$$\begin{aligned} 2436 &= 13 \cdot 187 + 5 \\ 13 &= 5 \cdot 2 + 3 \\ 5 &= 3 \cdot 1 + 2 \\ 3 &= 2 \cdot 1 + 1 \\ 2 &= 1 \cdot 2 + 0 \end{aligned}$$

   1. Since gcd(2436,13) = 1 we can show that

$$937 \cdot 13 \textbf{ mod } 2436 = 12181 \textbf{ mod } 2436 = 1$$

   937 is a modular inverse of 13 modulo 2436.

Solution - # 3

3. By inspection (as discussed prior to Example 1), find an inverse of 4 modulo 9.
   - By inspection, we can show that 7 is a modular inverse of 4 modulo 9. Namely,

$$7 \cdot 4 \textbf{ mod } 9 = 28 \textbf{ mod } 9 = 1.$$

Solution - #4

4. By inspection (as discussed prior to Example 1), find an inverse of 2 modulo 17.
   - By inspection, we can show that 43 is a modular inverse of 2 modulo 17. Namely,

$$43 \cdot 2 \textbf{ mod } 17 = 86 \textbf{ mod } 17 = 1.$$

Synopsis

After working through problems 1 through 4 (I am only showing problems 1 and 3 for space simplicity) the first thing that I noticed for a modular inverse to occur is that the gcd of the two numbers in the modulo must be **relatively prime**. If these two numbers are relatively prime, then a modular inverse will exist for those numbers. Then, for a number to be considered relatively prime we must have the following expression be true:

$$x \cdot a \textbf{ mod } b = 1 \tag{1}$$

where x is considered to be the number that we are checking to see if it is an inverse of a **mod** b. If equation (1) holds, then we know that x is a modular inverse.

Furthermore, we can find the modular inverse by calculating the Bezout coefficients. Take number 3 for example:

$$9 = 4 \cdot 2 + 1 \tag{2}$$

$$4 = 1 \cdot 4 + 0 \tag{3}$$

$$\gcd(9,4) = 1 \quad \text{(9 and 4 are relatively prime)} \tag{4}$$

$$1 = 9(1) + 4(-2) \quad \text{(1 and -2 are the Bezout coefficients)} \tag{5}$$

$$-2 \cdot 4 \textbf{ mod } 9 = -8 \textbf{ mod } 9 = 9 \textbf{ mod } 8 = 1 \quad \text{(-2 is indeed a modular inverse of 4 modulo 9)} \tag{6}$$

The result from equation (5) shows us that the following statement is true

$$4 \cdot (-2) \equiv 1 \textbf{ mod } 9.$$

Problem 3 Summary

Procedure

- For problems 1 and 2, show that the gcd(b,a) = 1 to show that a and b are relatively prime and thus a modular inverse exists.
  - After showing that gcd(b,a) = 1, show by the method of inspection that the proposed number is a modular inverse of a and b.
- For problems 3 and 4, show by inspection that the provided expression has a modulo of 1.
- For the synopsis, summarize the problems and show a quick example of the Bezout coefficients.

Key Concepts

- If the gcd(b,a) in the expression
$$x \cdot a \textbf{ mod } b = 1$$
is equal to 1, then a modular inverse does exist.
- We can show by the method of 'inspection' that a given number is a modular inverse.
- If we can calculate the Bezout coefficients for a given set of numbers, we can show that at least one is a modular inverse.

Variations

- We could be given a different set of expressions to determine if a modular inverse exists.
  - We would then use the same procedure in each part to determine if a given integer is a modular inverse.

Problem 4

Problem Statement

You are stranded on an island. Each day 2 jugs wash up on the beach, which are marked with exact measures of a and b cups. You also have an empty (unmeasured tub), and a jug maker. Jugs last one day and then disappear. An evil Volcano Demon requires you to measure out exactly 1 cup each day or someone will be tossed into the Volcano (then, Volcano Demon can make a single cup measure, and make big cake recipes measured in cups - his dream is being on a cooking show one day).

Below are the jug measures in cups that show up each day for 9 days. For each day:

- Using the provided cups and the jug maker, find the smallest cup measure you can create each day; use the jug making videos or Euclidean Algorithm for intuition. You may put down your answers: you do not need to show your work.
- On which days will someone be tossed into a Volcano (you are unable to measure out 1 cup)?

1. 12,9
2. 27,5
3. 8,4
4. 66,11
5. 25,5
6. 12,3
7. 9,4
8. 141,19
9. 89,55

Solution

1. 12,9: Euclid's algorithm will show that the gcd(12,9) = 3 and therefore the smallest cup measure is 3. **Tossed into Volcano**.
2. 27,5: Euclid's algorithm will show that the gcd(27,5) = 1 and therefore the smallest cup measure is 1. **Not tossed into Volcano**.
3. 8,4: Euclid's algorithm will show that the gcd(8,4) = 4 and therefore the smallest cup measure is 4. **Tossed into Volcano**.
4. 66,11: Euclid's algorithm will show that the gcd(66,11) = 11 and therefore the smallest cup measure is 11. **Tossed into Volcano**.
5. 25,5: Euclid's algorithm will show that the gcd(25,5) = 5 and therefore the smallest cup measure is 5. **Tossed into Volcano**.
6. 12,3: Euclid's algorithm will show that the gcd(12,3) = 3 and therefore the smallest cup measure is 3. **Tossed into Volcano**.
7. 9,4: Euclid's algorithm will show that the gcd(9,4) = 1 and therefore the smallest cup measure is 1. **Not tossed into Volcano**.
8. 141,19: Euclid's algorithm will show that the gcd(141,19) = 1 and therefore the smallest cup measure is 1. **Not tossed into Volcano**.
9. 89,55: Euclid's algorithm will show that the gcd(89,55) = 1 and therefore the smallest cup measure is 1. **Not tossed into Volcano**.

From the above, we can see that the days that the person is thrown into the Volcano are on days 1,3,4,5 and 6. This is because the cup measurements that wash up on the beach have a gcd greater than 1.

Problem 4 Summary

Procedure

- For each pair of jugs that wash up on shore, calculate the gcd of the two jugs to determine if someone is going to be tossed into the volcano.
- If the gcd of the two jugs is 1 (or the quantities in the jugs are relatively prime) then the person will not be tossed into the volcano.
- If the gcd of the two jugs is not 1 (or the quantities in the jugs are not relatively prime) then the person will be tossed into the volcano.

Key Concepts

- To satisfy the Volcano Demon, the gcd of the two jugs must be 1.

Variations

- We could be given a different set of jugs to determine if the Volcano Demon has been satisfied.
  - We would then proceed to calculate the gcd of these new jugs to determine if the Volcano Demon has been satisfied.

Problem 5

Problem Statement

The Volcano God again. Same set up, HOWEVER, the jug maker is broken. This time you have the same need to make the smaller measure with jugs but you have no jug maker. Notice the Euclidean Algorithm requires that we use these new jugs (the remainders) again and again.

For day 8, a day no one was thrown into the Volcano, show the recipe for measuring out a single cup measure. Show your work. You may use either a formal or informal process to find the answers (the recipes numbers are the Bezout Coefficients).

Remember:

- What tells me the smallest amount I can measure? (GCD)
- What tells me the recipe for measuring? (Bezouts)

Solution

On day 8, jugs with measurements 141 and 19 wash up on the shore. First, we want to calculate the gcd of these jugs. Namely,

$$141 = 19 \cdot 7 + 8 \tag{1}$$
$$19 = 8 \cdot 2 + 3 \tag{2}$$
$$8 = 3 \cdot 2 + 2 \tag{3}$$
$$3 = 2 \cdot 1 + 1 \tag{4}$$
$$2 = 1 \cdot 2 + 0. \tag{5}$$

From equation (4) we can see that the gcd(141,19) = 1. Now we wish to find the Bezout coefficients. Namely,

$$1 = 3 - 2(1) \quad \text{(Rewriting equation (4))} \tag{6}$$
$$2 = 8 - 3(2) \quad \text{(Rewriting equation (3))} \tag{7}$$
$$3 = 19 - 8(2) \quad \text{(Rewriting equation (2))} \tag{8}$$
$$8 = 141 - 19(7) \quad \text{(Rewriting equation (1))} \tag{9}$$
$$3 = 19 - (141 - 19(7))(2) \quad \text{(Substituting (9) into (8))} \tag{10}$$
$$= 19 + 141(-2) + 19(14) \quad \text{(Simplifying)} \tag{11}$$
$$= 141(-2) + 19(15) \quad \text{(Simplifying)} \tag{12}$$
$$2 = 141(1) + 19(-7) - (141(-2) + 19(15))(2) \quad \text{(Substituting (9) and (12) into (7))} \tag{13}$$
$$= 141(1) + 19(-7) + 141(4) + 19(-30) \quad \text{(Simplifying)} \tag{14}$$
$$= 141(5) + 19(-37) \quad \text{(Simplifying)} \tag{15}$$
$$1 = 141(-2) + 19(15) - (141(5) + 19(-37))(1) \quad \text{(Substituting (12) and (15) into (6))} \tag{16}$$
$$= 141(-2) + 19(15) + 141(-5) + 19(37) \quad \text{(Simplifying)} \tag{17}$$
$$= 141(-7) + 19(52). \quad \text{(Simplifying)} \tag{18}$$

From the result in equation (18) we can then say the Bezout coefficients are

$$s = -7, t = 52. \tag{19}$$

These values (s and t) will allow us to make a cup measure of 1 on day 8 where the jug measurements are 141 and 9. Essentially, we take out 7 measurements of 141 and add in 52 measurements of 19.

Problem 5 Summary

Procedure

- Begin by showing that the gcd(141,19) = 1.
- Proceed to solve for the remainders that were found in the EEA of the problem.
- Proceed to substitute the remainder equations from the EEA into the other remainder equations until they are simplified in terms of 141 and 19.
- Repeat the above procedure until we have simplified the remainder equation for 1.
- The Bezout coefficients are then the multiples that are next to 141 and 19.

Key Concepts

- In the formula

$$x \cdot a \textbf{ mod } b = 1$$

the Bezout coefficients tell us α and β in

$$a(\alpha) + b(\beta) = \gcd(a, b)$$

are modular inverses (or at least one of them is).

Variations

- We could be asked to find the Bezout coefficients of other expressions.
  - We would then use the same procedure as in this problem to find the Bezout coefficients.

Problem 6

Problem Statement

Notice there are 3 fundamental situations with the pairs of jugs.

- Some pairs of jugs you can use to measure any amount asked for.
- Some pairs of jugs you can only measure multiples of the given jugs.
- Some pairs of jugs which can measure some additional amounts smaller than each of the given jugs, but not all possible amounts.

Explain mathematically what each case means in terms of Primes, GCD, etc.. Give an example of each.

Solution

- Some pairs of jugs you can use to measure any amount asked for.
  - This is a scenario in which the gcd of the jugs is 1. Because of this, we can use the extended Euclidean algorithm and calculate the Bezout coefficients to measure any desired amount. Because the gcd of the jug measurements is 1 this implies that the measurements are relatively prime. For example, this scenario occurs on day 2 when the jugs that wash up on shore measure 25 and 9.
- Some pairs of jugs you can only measure multiples of the given jugs.
  - This is a scenario in which the gcd of the jugs is not 1. Specifically, the gcd of the jugs is the smaller measurement of the two jugs that wash up on shore. Take for example day 4 when jugs with measurements 66 and 11 wash up on shore. In this scenario we can only measurement jugs that are multiples of 11. In this specific example the gcd of the two jugs measurements is 11. This is why we can only measure multiples of the given jugs.
- Some pairs of jugs which can measure some additional amounts smaller than each of the given jugs, but not all possible amounts.
  - This is a scenario in which the gcd of the jugs is again not 1. Specifically, the gcd of the jugs is a measurement that will equally divide both jug measurements. Take for example day 1 when jugs with measurements 12 and 9 wash up on shore. In this example the gcd of the two measurements is 3. So we can make measurements that are smaller than each given jug but not all possible measurements.

Problem 6 Summary

Procedure

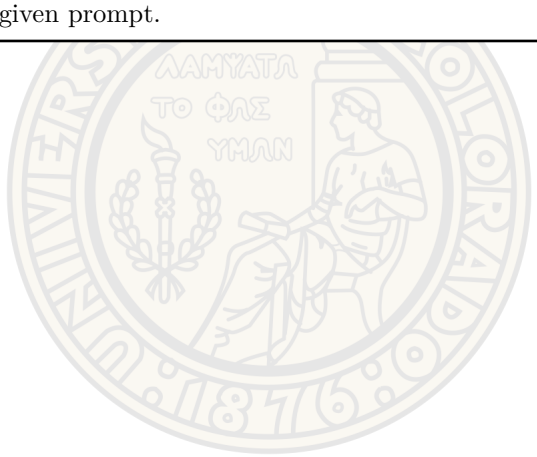
- Answer the prompts using mathematical principles that are found in the problem.

Key Concepts

- If the gcd of the two jug measurements is 1, then you can make any jug measurement that is asked of you.
- If the gcd of the two jug measurements is equal to the smaller of the two jugs, then you can only make measurements that are multiples of the given jugs.
- If the gcd of the two jug measurements is smaller than the smaller of the two jugs but not equal to one, then you can make smaller measurements that are factors of each jug measurement.

Variations

- We could be asked the same or similar questions for a different set of jugs.
  - In this case we would resort to discussing what the gcd of the jug measurements is and how that corresponds to the given prompt.



Problem 7

Problem Statement

Now let's formalize the process for this new method without a jug maker, which is the Extended Euclidean Algorithm by working through the algorithm using the example in the video.

1. Using 77 and 14, what variables represent 77 and 14 in the algorithm? (be specific)
2. Which values will update through the program, m and n, or m_0 and n_0?

s1, t1 and s2, t2 are the initial "recipes" for m and n.

- $77 = 1 \cdot 77 + 0 \cdot 14$
- $14 = 0 \cdot 77 + 1 * 14$

3. Rewrite above using s1, t1 and s2, t2 and m and n, and m_0 and n_0

Before the loop we have now initialized our settings.

4. What is the condition on the loop (what will cause it to stop?)
5. What are k and q and why are we finding them?
6. Which lines update m and n? (write them here)

The code in blue is calculating the new values (the new "recipe") of s1, t1 and s2, t2. Notice the use of s1, t1, s2, t2

7. How are these DIFFERENT variables related to s1, t1 and s2, t2?
8. How many distinct variables are being updated in the blue code?
9. How many are being updated in the pink code? (look carefully)

Solution

1. Using 77 and 14, what variables represent 77 and 14 in the algorithm? (be specific)
   - 77 represents the initial value of m and 14 represents the initial value of n of which we are trying to calculate the gcd for.
2. Which values will update through the program, m and n, or m_0 and n_0?
   - While present in the while loop, m and n are being updated. m_0 and n_0 are not being changed as they are just the initial values of m and n that are fed to the algorithm.

s1, t1 and s2, t2 are the initial "recipes" for m and n.

- $77 = 1 \cdot 77 + 0 \cdot 14$
- $14 = 0 \cdot 77 + 1 * 14$

3. Rewrite above using s1, t1 and s2, t2 and m and n, and m_0 and n_0

```
1       m_0 = m
2       n_0 = n
3       (s_1,t_1) = (1,0)
4       (s_2,t_2) = (0,1)
5
```

Before the loop we have now initialized our settings.

4. What is the condition on the loop (what will cause it to stop?)

- When n becomes less than or equal to 0.

5. What are k and q and why are we finding them?
   - k is the modulo of m and n. This is the remainder that is calculated in the extended Euclidean algorithm and we need this as it is essentially to the algorithm.
   - n is the updated value of k. This is needed to determine when the loop is going to break.
6. Which lines update m and n? (write them here)

```
1        m = n
2        n = k
3
```

The code in blue is calculating the new values (the new "recipe") of s1, t1 and s2, t2. Notice the use of s1, t1, s2, t2

7. How are these DIFFERENT variables related to s1, t1 and s2, t2?
   - They are essentially the next increment for the Bezout coefficients. These values rely on the next Bezout coefficients.
8. How many distinct variables are being updated in the blue code?
   - The new values for $\hat{s}_1, \hat{t}_1, \hat{s}_2, \hat{t}_2$ are being updated. So this means that four variables are being updated that are reliant upon previous values. These variables technically didn't exist before these lines in the algorithm so one could make the argument that they aren't being updated and instead are being assigned.
9. How many are being updated in the pink code? (look carefully)
   - s_1, t_1, s_2, t_2 are being updated. So four variables are being updated in the pink code.

Problem 7 Summary

Procedure

- Answer the prompts by referencing the algorithm that is found in Siriam's video.

Key Concepts

- In the context of this algorithm, the numbers 77 and 14 represent the input parameters for the algorithm.
- The values m and n are the values that are updated in the group of m, n, m_0, n_0.
- The while loop that is in the algorithm will run until n becomes less than or equal to 0.
- This algorithm is solving for the Bezout coefficients for a given set of numbers m and n.

Variations

- We could be asked to analyze a different algorithm and comment on specific aspects of it.
  - We would then just use common sense and analyze the algorithm.

Problem 8

Problem Statement

Write the code for the Extended Euclidean Algorithm using Sriram's algorithm. Show the code here in plain text with comments - use feedback on code style from last week. This can be a screenshot.

Solution

Below is my code for the extended Euclidean algorithm in Python.

```
# EEA - Greatest Common Divisor or Euclidean algorithm for two numbers m and n
# Input:
#   m - Integer value that represents the larger of the two values in the EEA calculation
#   n - Integer value that represents the smaller of the two values in the EEA calculation
# Algorithm:
#   * Assign values to the initial of m and n with m_0 and n_0
#   * Assign values to the initial Bezout coefficients s_1,t_1 with 1 and 0
#   * Assign values to the updated Bezout coefficients s_2,t_2 with 0 and 1
#   * Execute the while loop until n is less than or equal to 0 (usually equal to 0)
#       * Calculate the modulo of m and n and assign it to k
#       * Calculate the integer division of m and n and assign it to q
#       * Update m to the current value of n and n to to the current value of k
#       * Calculate new values for the Bezout coefficients S_1,T_1 with s_2,t_2
#       * Calculate the new updated Bezout coefficients S_2,T_2 with s_1 - q * s_2, t_1 - q * t_2
#       * Update the Bezout coefficients (which will eventually be returned) s_1,t_1 with S_1,T_1
#       * Update the updated Bezout coefficients s_2,t_2 with S_2,T_2
#   * Return the GCD (m) and the Bezout coefficients (s_1,t_1) as an array after the loop
# Output:
#   This algorithm returns an array of values related to the Extended Euclidean Algorithm
#   m - This is the GCD of the two original numbers m and n
#   s_1 - This is the first Bezout coefficient (s) that is returned
#   t_1 - This is the second Bezout coefficient (t) that is returned
def EEA(m,n):
    m_0,n_0 = m,n
    s_1,t_1 = 1,0
    s_2,t_2 = 0,1
    while (n > 0):
        k = m % n
        q = m // n
        m = n
        n = k
        S_1,T_1 = s_2,t_2
        S_2,T_2 = s_1 - q * s_2, t_1 - q * t_2
        s_1,t_1 = S_1,T_1
        s_2,t_2 = S_2,T_2
    return [m,s_1,t_1]

```

I used parallel assignment in Python to cut down on the syntax of the actual function itself. Comments are listed in green text that explain the input, algorithm, and output for this function. This code came from Siriam's video in this weeks lectures.

Problem 8 Summary

Procedure

- Write code in Python that calculates the Bezout coefficients for a given set of numbers

Key Concepts

- This problem showcases a way to code the Extended Euclidean Algorithm (EEA) in Python such that it will return the gcd of two numbers and the Bezout coefficients.

Variations

- We could be asked to program a different algorithm.
  - In this case we would then write a new algorithm that achieves what is asked of us.

Problem 9

Problem Statement

Test your code with #41 - 44 page 273. Show the results as linear combinations as instructed (either by hand or you can include as code).

HW Helper Function

Before I go into testing my code with the problems in the book, I want share a function that I created to output cleaner results.

```
def HWHelper(m,n):
    m_0,n_0 = m,n
    print(f"We seek to the Bezout coefficients of {m_0} and {n_0}, namely, express gcd({m_0},{n_0}) as a linear combination of {m_0} and {n_0}.")
    print(f"The Extended Euclidean Algorithm by default is: gcd(m,n) = sm + tn.")
    result = EEA(m,n)
    print(f"The gcd(m,n) = {result[0]}, the Bezout coefficients are s = {result[1]}, t = {result[2]}.")
    print(f"Finally, we have: {result[0]} = ({result[1]}){m_0} + ({result[2]}){n_0} expressed as a linear combination.")

```

Solution - # 41

```
HWHelper(26,91) #Output follows after this

We seek to the Bezout coefficients of 26 and 91, namely, express gcd(26,91) as a linear combination of 26 and 91.
The Extended Euclidean Algorithm by default is: gcd(m,n) = sm + tn.
The gcd(m,n) = 13, the Bezout coefficients are s = -3, t = 1.
Finally, we have: 13 = (-3)26 + (1)91 expressed as a linear combination.

```

Solution - # 42

```
HWHelper(252,356) #Output follows after this

We seek to the Bezout coefficients of 252 and 356, namely, express gcd(252,356) as a linear combination of 252 and 356.
The Extended Euclidean Algorithm by default is: gcd(m,n) = sm + tn.
The gcd(m,n) = 4, the Bezout coefficients are s = -24, t = 17.
Finally, we have: 4 = (-24)252 + (17)356 expressed as a linear combination.

```

Solution - # 43

```
HWHelper(144,89) #Output follows after this

We seek to the Bezout coefficients of 144 and 89, namely, express gcd(144,89) as a linear combination of 144 and 89.
The Extended Euclidean Algorithm by default is: gcd(m,n) = sm + tn.
The gcd(m,n) = 1, the Bezout coefficients are s = 34, t = -55.
Finally, we have: 1 = (34)144 + (-55)89 expressed as a linear combination.

```

Solution - # 44

```
HWHelper(1001,100001) #Output follows after this

We seek to the Bezout coefficients of 1001 and 100001, namely, express gcd(1001,100001) as a linear combination of 1001 and 100001.
The Extended Euclidean Algorithm by default is: gcd(m,n) = sm + tn.
The gcd(m,n) = 11, the Bezout coefficients are s = -999, t = 10.
```

```
6   Finally, we have: 11 = (-999)1001 + (10)100001 expressed as a linear combination.
7
```

Problem 9 Summary

Procedure

- Write a helper function that will print to console the problems that we are asked to solve.
- Use the helper function and show the output that is produced from the algorithm and helper function.

Key Concetps

- This problem showcases the use of the Extended Euclidean Algorithm (EEA) and shows the output for a set of numbers.
- The EEA returns the gcd of two numbers and the Bezout coefficients.
- The Bezout coefficients (at least of them) are solutions to modular inverses.

Variations

- We could be asked to demonstrate the use of a function with different initial values.
  - In this case we would use the function with these new values.

Problem 10

Problem Statement

Finally, for a project check in, work through the example on 8 and 9 page 300 - 301 on your own.

Create a similar SIMPLE 4 letter word as an example following this format and using the keys from the example.

Solution

The four letter word that I am going to use in this example is **WORK**.

The key that is going to be used to encrypt this word is (2537,13). It follows that

$$\gcd(e,(p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1. \tag{1}$$

To encrypt this message, we first need to represent the original letters of the word **WORK** with their numerical equivalences. For **WORK** it would be represented with

$$\text{WORK} = \ 22\ 14\ 17\ 10. \tag{2}$$

The numerical equivalences of the letters in **WORK** are represented with an integer value where we first assign A = 0, B = 1, ... Z = 25. We now need to perform a shift on the letters in **WORK**. We are going to use the Caesar shift method where we shift the letters by 3 and then perform the modulo of that new value with 26. Precisely,

$$f(p) = (p+3) \ \textbf{mod}\ 26. \tag{3}$$

Performing the Caesar shift of **WORK** with that found in equation (3) we now have

$$f(W) = (22+3) \ \textbf{mod}\ 26 = (25) \ \textbf{mod}\ 26 = 25 \tag{4}$$
$$f(O) = (14+3) \ \textbf{mod}\ 26 = (17) \ \textbf{mod}\ 26 = 17 \tag{5}$$
$$f(R) = (17+3) \ \textbf{mod}\ 26 = (20) \ \textbf{mod}\ 26 = 20 \tag{6}$$
$$f(K) = (10+3) \ \textbf{mod}\ 26 = (13) \ \textbf{mod}\ 26 = 13. \tag{7}$$

We now have the Caesar shift of **WORK** to be represented with 25 17 20 13. In two separate blocks **WORK** would be represented as 2517 2013. We now need to encrypt the blocks with the following mapping

$$C = M^{13} \ \textbf{mod}\ 2537. \tag{8}$$

Performing the mapping on the two blocks 2517 2013 we now have

$$C = 2517^{13} \ \textbf{mod}\ 2537 = 1744 \tag{9}$$
$$C = 2013^{13} \ \textbf{mod}\ 2537 = 1951. \tag{10}$$

The encrypted message is now

$$\textbf{1744 1951}. \tag{11}$$

Problem 10 Summary

Procedure

- Show that the gcd of the keys that are presented in the problem is equal to one.
- Write out a numerical definition of the word that we are trying to encrypt.
- Perform a shift on each letter of the word and calculate the modulo of that number.
- Use FME to encrypt the blocks of numbers.
- Show the encrypted word with the results from the FME algorithm.

Key Concepts

- This problem showcases in a snapshot how a piece of data can be encrypted with modulo operations.
- We use gcd to show that the keys and the Euler Totient function of our prime numbers are relatively prime.
- We represent letters in a message with numerical values such that they can be encrypted later on.
- Encryption takes advantage of FME and other mathematical operations to safely encrypt messages.
- Encryption in the method that is provided in this example is a safe way to make sure people who intercept a message may not be able to break the encryption easily.

Variations

- We could be asked to decrypt a message with known public keys and value of n.
  - We would then use FME with the private key that is given to us to unwind the message and show what it represents. Namely,

    $$M = C^d \textbf{ mod } n$$

    where M is our decrypted block of text, C is the encrypted block of text, d is our private key, and n is the value that is found from our Euler Totient function value with our prime numbers.