Cryptography Homework 2—Modular Arithmetic with Python

KEY

1. Compute (you can do this easily at an interactive Python prompt.)
   1. 34 mod 18  
      Text

      Description automatically generated
   2. (34 + 97) mod 12 Note: the mod operator has the same priority as multiplication. In a tie, Python executes operators from left to right.  
      Text

      Description automatically generated
   3. 14 \* 71 mod 15  
      A screen with numbers and letters on it

      Description automatically generated with low confidence
   4. 152 / 71  
      
   5. 152 // 71 (in Python, // is integer division)  
      
   6. 152 – (152 // 71) \* 71 (this is the remainder after integer division)  
      
   7. 152 mod 71 (you should see that the answer is the remainder when you divide 152//71 (integer division)  
      
2. Compute gcd(36, 45) and gcd(44, 45) using the GCD function you imported above. You should get 9 and 1 as answers, as a check to make sure the code is correct. Now compute gcd(452, 973) and gcd(452,1496). Which one of the pairs of numbers relatively prime, and what is the GCD of the pair that is not relatively prime?  
   Text

   Description automatically generated  
   The pair 452, 973 is relatively prime so the GCD is 1. The pair 452, 1496 has a GCD of 4, so it is not relatively prime.
3. For the following numbers, compute the GCD of the number and the modulus. If the number and the modulus are relatively prime, compute the multiplicative inverse. You can use either brute force in Python, or the inverse() function from PyCryptodome.  
   from Crypto.Util.number import inverse )
   1. 17 mod 26  
      Text

      Description automatically generated
   2. 16 mod 26  
      Text

      Description automatically generated with medium confidence
   3. Of the two numbers above, which could be used for the key of an affine cipher and which could not? Why?  
      16 will not work in the affine cipher with mod 26 because 16 and 26 share a factor, GCD is not 1. 17 will work because GCD of 17 and 26 is 1.
   4. Compute the multiplicative inverse of 17 in modulus 27, 28, and 29. Note that the inverse is completely different when the modulus changes.  
      Text

      Description automatically generated  
      A picture containing text, device, meter

      Description automatically generated
4. Least Common Multiple. The LCM(a, b) is the smallest number that can be divided by both   
   a and b. It is easily computed as a \* b // GCD(a, b)
   1. Compute the lcm of something simple, like 6 and 9. Check your answer by hand to verify that the equation is correct.
   2. Compute LCM(252, 196)  
      Text

      Description automatically generated  
      Defining the LCM function was not required, but shown for fun.
5. Use the file caesarCipher.py for this. The symbol set is expanded from the most common set, A – Z, and includes lower case, numbers, and some punctuation.   
   SYMBOLS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 !?.'  
   The caesarCipher.py code has a problem, which you can see by encrypting a message with a key = 40. What happened? How can you fix it?  
   Graphical user interface, text

   Description automatically generated  
   The lower case letters are shifted to the same letter but in upper case.  
   Text

   Description automatically generated  
   I know of two solutions: never use a key of 40, or remove the lower case letters. You could shuffle the letters, but it is no longer a Caesar cipher if you do that.
6. Use the file affineCipher.py (from *Cracking Codes with Python*) for this. It also used an expanded symbol set.  
   SYMBOLS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890 !?.'
   1. This symbol set is not a field, because not all elements have a multiplicative inverse. What caused that to happen? (Hint: What is the length of SYMBOLS?) What happens if you use a key with A = 2 and B = 13, or key = 135?  
      Graphical user interface, text, application

      Description automatically generated  
        
      
   2. Add or remove characters from SYMBOLS so that all elements of SYMBOLS have multiplicative inverses. Again len(SYMBOLS) is important; you are adjusting the length so that it is a \_\_\_\_\_ number.
      1. What SYMBOLS set did you choose, and what is its length?  
         Text

         Description automatically generated
      2. Encrypt a message. Hand in the encrypted message and the key you used.