Modulo Arithmetic Assignment

Q1. With your own words, explain the following items on your wiki.

Remainder

Modulus

Equivalence class

Congruence

Q2. Try to find the answer to the following without using a computer or calculator for the calculations.

127 mod 15

-127 mod 15

145 mod 22

395 mod 42

Q3. Are the following statements true or false? Explain your answers, and if false, write down the correct answer.

5 is congruent to 5 in modulo 5

1 is congruent to 6 in modulo 5

17 is congruent to 395 in modulo 42

1 mod 5 and -4 mod 5 are in the same equivalence class

2 mod 5 and 28 mod 5 are in the same equivalence class

A is congruent to 17 (mod 5). The value of A is...?

Q4. Answer the following questions. Show your thought process by writing on your wiki.

(918 + 335) mod 30

If A mod 9 is 7, then write the result of (A - 21) mod 9

Q5. Try to find the answer to the following without using a computer or calculator for the calculations. Describe the process that you followed on your wiki.

Find the Greatest Common Divisor (GCD) of 400 and 200.

Find the GCD of 292 and 76.

Find the GCD of 30 and 260.

Find the GCD of 12 and 56.

Q6. Try to find the modular inverse of the following numbers. Show your thought process. Feel free to use tools to achieve this (LibreOffice Calc can work well). Note B must be an integer as well.

Find the inverse of 3 modulo 8.

Find the inverse of 12 modulo 35.

Find the inverse of 19 modulo 73.

Lastly, what about 12 modulo 56?

Q7. In a short paragraph, explain the discrete logarithm problem, and how is it used in modern cryptography. Show at least two different examples.

Q8. Observe the schematic of a random number generator that is based on Elliptic Curve Cryptography (ECC). n0n0 is the seed that is repeatedly used as a multiplier for point PP, taking the x-coordinate of the result. Answer the following questions:

Does the discrete logarithm problem apply here?

What is the security of this random number generator dependent on? Is it secure?

Q9. Explain what the num_factorisation.pyx code does. Write your answer on your wiki. Experiment with the factorisation algorithm prime_factors1() using your own integers.

Q10. Implement your own factorisation algorithm in the prime_factors2() function. Show that it works. BONUS for a more efficient algorithm!

Q11. Check out the randRun(30) function. Using the given script, measure the time it takes to factorise pseudo-random numbers up to 30 digits with both functions. Run the experiment 3 times, and plot the results using your favourite plotting software (Hint: "gnuplot"). Upload the plots on your wiki.