Mathematics 3159A Assignment 2 Fall 2020

Instructions

- This assignment is due on Tuesday, October 6, 2020 at 2:00 PM EDT. Late submissions will **not** be accepted.
- This assignment consists of two problems. You should choose one for submission.
- Your solution needs to be formatted using the LaTeXtemplate available on OWL.
- All solutions must be written in full sentences.
- You are not allowed to work with others or use any online resources.
- This assignment is worth 5 points.

Problem 1.

In class, we showed that if Eve has an ElGamal oracle, then she can break the Diffie–Hellman cryptosystem. In this exercise, we will show the converse of that, namely that if Eve has a Diffie–Hellman oracle, she can break the ElGamal encryption.

To this end, we fix a large prime p and a primitive root g mod p. By a Diffie–Hellman oracle, we will understand an algorithm that given any $A, B \in \mathbb{F}_p$ such that

$$A = g^a \mod p \qquad \text{and} \qquad B = g^b \mod p$$

for some $a, b \in \mathbb{N}$, returns the value g^{ab}, and does so in polynomial time.

1. State carefully the ElGamal Problem, i.e., the underlying computational problem of the ElGamal cryptosystem, just as we stated the Diffie–Hellman Problem.
2. Show carefully that if Eve has a Diffie–Hellman oracle, then she can solve the ElGamal Problem.

Problem 2.

In this exercise, we will be implementing Shanks' Babystep-Giantstep algorithm for the Discrete Logarithm Problem in $\mathbb{F}_p^*$.

Statement

The assignment has two parts.

Mathematics 3159A Assignment 2 Fall 2020

1. Write a function in Python3 called `solve` that, given a prime p and any two integers g and h, returns the smallest positive integer x such that

$$g^x \equiv h \bmod p$$

   if such a number exists, or a warning saying there is no solution if it does not.

   Your program must use Shanks' algorithm. The trivial brute-force algorithm that computes all powers of g modulo p will receive no credit.

2. Download the file `generate_input.py` from OWL, use it to obtain three pairs (p, g, h) by running

```
python generate_input.py [last three digits of your student number]
```

   and run your program on these three inputs.

Your submission must consist of a single PDF file containing:

1. the *Python code* implementing your solution;
2. and the three *inputs you generated*, and the *output of your program* run on these three inputs.

Examples

Here are some examples of what your function `solve` should do:

```
>>> solve(13,6,5)
x = 9 is a solution
>>> solve(7,2,5)
there is no solution
>>> solve(17389,9704,13896)
x = 1159 is a solution
```

Notes

- The numbers generated by `generate_input.py` are quite big, so a brute-force solution will not work.
- The file `generate_input.py` is written in Python3, and so should be your solution. Make sure you are using a 64bit version of Python3
- Your submission must use the LaTeX template available on OWL.

- Your code should not make use of any external libraries such as `numpy` or `math`. All the auxiliary functions should be implemented by you, and should be included in your submission. You should only use the most basic arithmetic operations such as `+`, `-`, `*`, `//`, `%`.
- Comments in the code are not mandatory. However in the case of an incorrect solution, the comments can provide grounds for partial credit.