Instructions

1. You are encouraged to work in groups of two. If you cannot find a partner, you can work alone.
2. Please submit **one** copy of the assignment; if you are working with a partner, both names should appear on the assignment.
3. For **Part A** of the assignment, you must submit an electronic copy of your Perl application using subversion. Note that you will get a grade of zero if your program does not compile.

Part A

Write a Perl application which implements the ElGamal digital signature scheme. Your command-line program ought to be invoked as follows: `sign 11 6 3 7` and then accept a single line of `ASCII` text until the new-line character appears (i.e., until you press enter). That is, once you type `sign 11 6 3 7` at the command line, and press return, you type a message: '`A message.`' and after you have pressed return again, the digital signature, which is going to be a pair of positive integers, will appear below.

We now explain how to obtain this digital signature: first convert the characters in the string '`A message.`' into the corresponding `ASCII` codes, and then obtain a *hash* of those codes by multiplying them all modulo 11; the result should be the single number 5. To see this observe the table:

A	65	10
	32	1
m	109	10
e	101	9
s	115	1
s	115	5
a	97	1
g	103	4
e	101	8
.	46	5

The first column contains the characters, the second the corresponding `ASCII` codes, and the i-th entry in the third column contains the product of the first i codes modulo 11. The last entry in the third column is the hash value 5.

We sign the hash value, i.e., if the message is $m =$ `A message.`, then we sign hash$(m) = 5$. Note that we invoke `sign` with four arguments, i.e., we invoke it with p, g, x, k (in our running example, 11,6,3,7 respectively).

Here p must be a prime, $1 < g, x, k < p-1$, and $\gcd(k, p-1) = 1$. This is a condition of the input; you don't have to test in your program whether the condition is met—we may assume that it is.

Now the algorithm signs $h(m)$ as follows: it computes

$$\begin{aligned} r &= g^k \pmod p \\ s &= k^{-1}(h(m) - xr) \pmod{(p-1)} \end{aligned}$$

If s is zero, start over again, by selecting a different k (meeting the required conditions).

The signature of m is precisely the pair of numbers (r, s).

In our running example we have the following values:

$m =$ `A message.`; $\quad h(m) = 5; \quad p = 11; \quad g = 6; \quad x = 3; \quad k = 7$

and so the signature of '`A message.`' with the given parameters will be:

$$\begin{aligned} r &= 6^7 \pmod{11} = 8 \\ s &= 7^{-1}(5 - 3 \cdot 8) \pmod{(11-1)} = 3 \cdot (-19) \pmod{10} = 3 \cdot 1 \pmod{10} = 3 \end{aligned}$$

i.e., the signature of '`A message.`' would be $(r, s) = (8, 3)$.

Part B

1. Can you identify the (possible) weaknesses of this digital signature scheme? Can you compose a different message m' such that $h(m) = h(m')$?
2. If you receive a message m, and a signature pair (r, s), and you only know p, g and $y = g^x \pmod p$, i.e., p, g, y are the *public* information, how can you "verify" the signature—and what does it mean to verify the signature?
3. Research on the web a better suggestion for a hash function h.
4. **[Bonus Question]** Show that when used without a (*good*) hash function, ElGamal's signature scheme is *existentially forgeable*; i.e., an adversary Eve can construct a message m and a valid signature (r, s) for m.
5. **[Bonus Question]** In practice k is a random number; show that it is absolutely necessary to choose a new random number for each message.
6. **[Bonus Question]** Show that in the verification of the signature it is essential to check whether $1 \le r \le p-1$, because otherwise Eve would be able to sign message of her choice, provided she knows one valid signature (r, s) for some message m, where m is such that $1 \le m \le p-1$ and $\gcd(m, p-1) = 1$.