DLP Exercises (Daily Due Sept 14)

Katherine E. Stange, CU Boulder

In the exercises that follow, you are asked to use Sage to do computations. It's strongly recommended to use Sage instead of other software, because Sage has so many in-built number theory functions that we will make use of[1]. In particular, if Sage does something with a one-line command, and I don't specifically ask you to implement it yourself, go for it. (For example, relevant to this assignment, Sage will compute the multiplicative order of an element in an `IntegerModRing` using the command `multiplicative_order()`.)

Exercise 1

Use Sage to find all primitive roots modulo the prime 29. One of them is 2, for example.

Solution.

If you type the following:

```
p = 29
R = IntegerModRing(p)
R(2).multiplicative_order()
```

Then sage will answer '28'.

The line `R = IntegerModRing(p)` tells Sage to create $\mathbb{Z}/p\mathbb{Z}$ as an object, and call it `R`. Then, the notation `R(2)` tells sage to create the residue 2 in the world $\mathbb{Z}/p\mathbb{Z}$. It's no longer an integer, `R(2)` is something "living mod p." Then I can ask for its multiplicative order, which only makes sense in that universe. (If you ask Sage `2.multiplicative_order()` you are asking for the multiplicative order of the integer 2 as a regular integer.)

The fact that the answer is 28 tells us that 2 is a primitive root. No power of 2 is 0. The unit group of $\mathbb{Z}/29\mathbb{Z}$ can't include 0 either. So if 2 is generating everything else, then it's definitely generating the whole unit group. Or, you notice I *told* you 2 is a primitive root, so that tells you 28 is the correct multiplicative order for primitive roots.

So, with that in mind, you could use the following to list all primitive roots:

```
for i in range(1,29):
    if R(i).multiplicative_order() == 28:
        print(i)
```

[1]But if you insist, you can use something else. I won't be able to support you with other languages/software, however.

The full list is

$$2, 3, 8, , 10, 11, 14, 15, 18, 19, 21, 2627$$

There are many other ways to do this, of course.

Exercise 2

Create a chart of the powers of 2 modulo 29 (use Sage).

Solutions. Here's some code that will do this quickly, after you've defined `R` as in the previous problem:

```
for i in range(1,29):
    print( i, R(2^i))
```

The output shows the exponent (i) and the result (2^i):

```
1 2
2 4
3 8
4 16
5 3
6 6
7 12
8 24
9 19
10 9
11 18
12 7
13 14
14 28
15 27
16 25
17 21
18 13
19 26
20 23
21 17
22 5
23 10
24 20
25 11
```

```
26 22
27 15
28 1
```

Exercise 3

Using the chart above, *by hand*, compute the following discrete logarithms:

1. $L_2(7)$
2. $L_2(11)$
3. $L_4(7)$ (hint: use item (1) above)

Solution. Looking at the table in the previous problem, you see that you get $2^i = 7$ when $i = 12$, so $L_2(7) = 12$.

Similarly, $L_2(11) = 25$.

For the last one, you're asking what k works to give $4^k = 7$. You know $2^{12} = 7$, so that means $(2^2)^6 = 4^6 = 7$. Hence the answer is 6.

Exercise 4

1. Using Sage, but without calling any discrete logarithm functionality directly, implement an algorithm to determine the discrete logarithm of an element a modulo n with respect to a generator (primitive root) g. Paste in the code to your solutions.
2. Use the code to compute that the discrete log of 17 to the base 2 in $\mathbb{Z}/197\mathbb{Z}$ is 159, i.e. $L_2(17) = 159$ in $\mathbb{Z}/197\mathbb{Z}$. (If this doesn't work, fix your code!)

Solutions. Here's some code that will work (`g` is the base, `n` is the modulus and `a` is what you want the log of).

```
g = 2
n = 197
a = 17
R = IntegerModRing(n)
for i in range(1,n):
    if R(g)^i == a:
        print(i)
```

Sage output: 159