University College London
Department of Computer Science

Cryptanalysis Exercises Lab 1

J. P. Bootle

January 20, 2017

jonathan.bootle.14@ucl.ac.uk

Version 2.0

Basic Algebra

Click on the "Ans" button to get a hint.

Shift-click on "Ans" buttons that have a green boundary to get a full solution. Click on the green square to go back to the questions.

Quiz Answer each of the following.

1. $3 + 4 \mod 5 =$ □ Ans □

2. $7 \times 8 \mod 11 =$ □ Ans □

3. GF(q) denotes a finite field, where $q = p^n$. What is the size of GF(11)?

If you are unsure about finite fields, go to slides 11-12 here.

□ Ans □

4. What is the characteristic of GF(11)?

□ Ans □

5. What is the size of GF(16)?

□ Ans □

6. What is the characteristic of GF(16)?

□ Ans □

7. Is $\mathbb{Z}_{10}$ a field?

□ True □ False

□

8. Compute $x \cdot (x^3 + 1)$ modulo $x^4 + x - 1$. Answer $=$ □

Ans □

Clear □

Getting Acquainted with SAGE

Using SAGE Math Cloud

SAGEMath is a free open-source maths software package based on Python. We will be using SAGE for most of the lab sessions in this course. SAGE can be accessed online via the SAGE Math Cloud.

1. First, click the link above and enter your details to register for a free account.
2. Once you have logged in successfully, click ‘Projects’ in the top left-hand corner.
3. Click ‘New Project’. Add a title (e.g. ‘Cryptanalysis’) and a description (e.g. ‘Code for COMPGA18‘) and click ‘Create Project’.
4. Click on the project you have just created.
5. Add a new file by clicking ‘New’. Choose a name for your file (e.g. ‘Lab Session 1’), and select ‘SageMath Worksheet’ as the type.
6. Now you are ready to write code! Type `3+4` and click run.
7. You can write many separate programs in the same window. You can also leave complex programs to run while you log out and do something else.

SAGE is built from Python. Most Python commands will work fine in SAGE, and details of more commands including number theory and algebra can be found here.

You can also install SAGE on your own computer by downloading here and using the installation guide here.

Basic Maths in SAGE

Find Sage commands to answer the following questions, and copy the answers into the boxes.

Click on the “Ans” button to get a hint.

Shift-click on “Ans” buttons that have a green boundary to get a full solution. Click on the green square to go back to the questions.

Quiz Answer each of the following.

1. What are the factors of 12345678? □
Ans □

2. What is the gcd of 3579609 and 890387967?
□ Ans □

3. What is $478^{-1} \mod 1329$?
□ Ans □

4. Is 253647728826477399266772652772816653569721 a prime number?

□ Ans □

5. What are the prime divisors of 2266719?

□ Ans □

6. What is the next prime number after 1 million?

□ Ans □

7. Create two (2 x 2) vectors, and two matrices with sizes (3 x 1) and (1 x 3). Multiply the first pair together, and the second pair together (Leave answer box blank).

□ Ans □

Clear □

Computing a modular inverse with Euclid

Click on the green letter in front of each sub-question (e.g. (a)) to see a solution. Click on the green square at the end of the solution to go back to the questions.

Click here for a reminder of the Extended Euclidean Algorithm.

Exercise 1. Let p and q be two distinct primes.

(a) Show how to use the extended Euclidean algorithm to simultaneously compute $p^{-1} \mod q$ and $q^{-1} \mod p$.
(b) What is the complexity of this approach in terms of bit operations?
(c) Compute 11^{-1} mod 17 using this method.
(d) Implement the Extended Euclidean Algorithm in SAGE, and use it to compute $7^{-1} \mod 159$.

Primality Testing

Click on the green letter in front of each sub-question (e.g. (a)) to see a solution. Click on the green square at the end of the solution to go back to the questions.

Click here for a reminder square-and-multiply algorithms.

Exercise 2.

(a) Create a function 'MyPower' which takes inputs a, k and n, and computes $a^k \mod n$ using a square-and-multiply algorithm.

(b) In the Fermat primality test, we test whether a number n is prime by computing $a^{n-1} \mod n$ and then checking whether the result is equal to 1. If the result is not 1, then the number is not prime! Using your function, and the `is_prime` function, find all of the *composite* numbers between 2 and 2000 that pass the Fermat test with $a = 2$. Repeat for $a = 5$.

(c) Using your answer to the previous question, or otherwise, find all of the Carmichael numbers between 2 and 2000. Hint: remember that if $\gcd(a, n) > 1$, then n does not need to pass the Fermat test to base a to be a Carmichael number.

(d) Test any Carmichael numbers that you have found using the Miller-Rabin test, again with $a = 2$ and $a = 5$. Do any of them pass the test?

(e) (Bonus Question) Find a number larger than 5000 which passes the Fermat test with base a, but fails the Miller-Rabin test to base a. Using the sequence of values from the Miller-Rabin test, can you factor the number without using trial division?

Solutions to Exercises

Exercise 1(a) If necessary, swap p and q so that $p > q$. Since p and q are distinct primes, $gcd(p, q) = 1$, and there exist integers A and B such that $Ap + Bq = 1$. Then $A = p^{-1} \mod q$ and $B = q^{-1} \mod p$. We compute these using the Extended Euclidean Algorithm.

One way to implement the extended Euclidean Algorithm is to use the back-tracking approach, to be demonstrated in lectures. Otherwise, the following method allows the answer to be calculated without working backwards.

Set $r_{-1} = p$ and $r_0 = q$. We also set $A_{-1} = 1$, $A_0 = 0$, and $B_{-1} = 0$, $B_0 = 1$. For each i, find a_{i+1}, r_{i+1} such that $r_{i-1} = a_{i+1}r_i + r_{i+1}$ as in the Euclidean Algorithm.

At each stage, compute $A_{i+1} = a_iA_i + A_{i-1}$ and $B_{i+1} = a_iB_i + B_{i-1}$. These values satisfy $A_ip - B_iq = (-1)^{i+1}r_i$. When the algorithm terminates after n steps, $r_n = gcd(p, q) = 1$. We take $A = (-1)^{n+1}A_n$ and $B = (-1)^nB_n$. □

Exercise 1(b) The Extended Euclidean Algorithm requires $O(\log(p)^2)$ bit operations. □

◀◀ ▶▶ ◀ ▶ Back

	a_i	A_i	B_i
-	-	1	0
-	-	0	1
$17 = 1 \cdot 11 + 6$	1	1	1
$11 = 1 \cdot 6 + 5$	1	1	2
$6 = 1 \cdot 5 + 1$	1	2	3

Figure 1: Gcd of 17 and 11

Exercise 1(c) Again, we can easily find the answer using the backtracking method shown in lectures. The alternative solution from an earlier part of the question is shown below.

Set $r_{-1} = 17, r_0 = 11$. Figure 1 shows working for the Extended Euclidean Algorithm. We find that $2 \cdot 17 - 3 \cdot 11 = 1$. Therefore 11^{-1} mod $17 \equiv -3 \equiv 14$.

□

◀◀ ▶▶ ◀ ▶ Back

Exercise 1(d) The SAGE code shown implements the Extended Euclidean Algorithm as presented in lectures.

```
def gcd1(a,b):
    if mod(a,b) == 0:
        return [b,0,1]
    else:
        q = (a- (a%b) )/ b
        [d, r, s]=gcd1(b,a-q*b)
        return [d,s,r-q*s]
```

When run on 159 and 7, the output is $[1, 3, -68]$, so the answer is -68.

□

Exercise 2(a) The following code implements the square-and-multiply Algorithm.

```
def MyPower(a,k,n):
    K = bin(k)[2:]
    A = a % n
    c = (A^ int(K[0]))
    for j in range(1,len(K)):
        c = (c^ 2) % n
        c = c*(A^ int(K[j])) % n
    return c
```

□

Exercise 2(b) The following code finds the answer for $a = 2$. For $a = 2$ you should get $341, 561, 645, 1105, 1387, 1729, 1905$. For $a = 5$, you should get $4, 124, 217, 561, 781, 1541, 1729, 1891$.

```
for i in range(2,2000):
    if is_prime(i)==False and MyPower(2,i-1,i)==1:
        print(i)
```

□

Exercise 2(c) The Carmichael numbers between 2 and 2000 are $561, 1105, 1729.$ □

◀◀ ▶▶ ◀ ▶ **Back**

Exercise 2(d) The following code carries out the Miller-Rabin test to base a. You should find that none pass with either $a = 2$ or $a = 5$.

```
def StrongTest(a,n):
    if (n%2)==0:
        return ’fail’
    b = n-1
    k=0
    while (b%2)==0:
        b = b/2
        k = k+1
    A = MyPower(a,b,n)
    if A == 1 or A == (n-1):
        return ’pass’
    for i in range(0,k):
        A = MyPower(A,2,n)
        if A == (n-1):
            return ’pass’
```

(code continues on the next page)

```
    if A == 1:
        return 'fail'
return 'fail'
```

□

Exercise 2(e) The number 5461 passes the Fermat test with base $a = 2$, but fails the Miller-Rabin test. From this, we can deduce that the sequence of values produced by the Miller-Rabin test ends in 1, but does not contain -1. Therefore, the sequence gives us a square-root 128 of 1 modulo 5461 which is not ± 1. We have $128^2 \equiv 1 \mod 5461$. Rearranging, $(128+1)(128-1) \equiv 0 \mod 5461$, but 128 is not congruent to ± 1. Therefore, $\gcd(129, 5461)$ and $\gcd(127, 5461)$ give non-trivial factors of 5461. We find that $5461 = 43 \times 127$. □

Solutions to Quizzes

Solution to Quiz: When working mod 5, we can add and subtract multiples of 5 freely.

$$3 + 4 = 7 = 7 - 5 \mod 5 = \boxed{2}$$

■

◀◀ ▶▶ ◀ ▶ Back

Solution to Quiz: When working mod 11, we can add and subtract multiples of 11 freely.

$$7 \times 8 = 56 = 56 - 5 \times 11 \quad \text{mod } 11 = \boxed{1}$$

■

Solution to Quiz: False, $\mathbb{Z}_{10}$ is not a field, because 2 and 5 are not invertible. ■

Solution to Quiz: We are working mod $x^4 + x + 1$, so we can add and subtract multiples of $x^4 + x + 1$ freely.

$$x \cdot (x^3 + 1) = x^4 + x \equiv -1 \mod (x^4 + x - 1) = \boxed{1}$$

■

Solution to Quiz: $A = matrix([[2,3],[3,2])$
$B = matrix([[3,4],[1,1])$
$A * B$

$A = matrix([[2,3,1]])$
$B = matrix([[3],[1],[2]])$
$A * B$ ■