

Week 6 - Programming Assignment [optional: extra credit]

[Help](#)

Warning: The hard deadline has passed. You can attempt it, but **you will not get credit for it**. You are welcome to try it as a learning exercise.

☐ In accordance with the Coursera Honor Code, I (Antonio Diaz Arroyo) certify that the answers here are my own work.

Question 1

Your goal in this project is to break RSA when the public modulus N is generated incorrectly. This should serve as yet another reminder not to implement crypto primitives yourself.

Normally, the primes that comprise an RSA modulus are generated independently of one another. But suppose a developer decides to generate the first prime p by choosing a random number R and scanning for a prime close by. The second prime q is generated by scanning for some other random prime also close to R . We show that the resulting RSA modulus $N = pq$ can be easily factored.

Suppose you are given a composite N and are told that N is a product of two relatively close primes p and q , namely p and q satisfy

$$|p - q| < 2N^{1/4} \quad (*)$$

Your goal is to factor N .

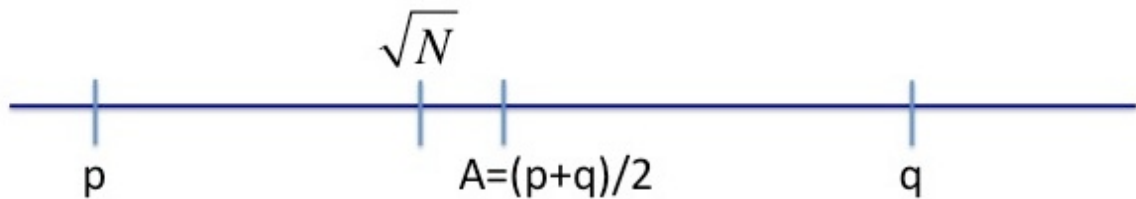
Let A be the arithmetic average of the two primes, that is $A = \frac{p+q}{2}$. Since p and q are odd, we know that $p + q$ is even and therefore A is an integer.

To factor N you first observe that under condition (*) the quantity \sqrt{N} is very close to A . In particular

$$A - \sqrt{N} < 1$$

as shown below. But since A is an integer, rounding \sqrt{N} up to the closest integer reveals the value of A . In code, $A = \text{ceil}(\text{sqrt}(N))$ where "ceil" is the ceiling function. Visually, the

numbers p, q, \sqrt{N} and A are ordered as follows:



Since A is the exact mid-point between p and q there is an integer x such that $p = A - x$ and $q = A + x$. But then

$$N = pq = (A - x)(A + x) = A^2 - x^2 \text{ and therefore } x = \sqrt{A^2 - N}$$

Now, given x and A you can find the factors p and q of N since $p = A - x$ and $q = A + x$.

In the following challenges, you will factor the given moduli using the method outlined above. To solve this assignment it is best to use an environment that supports multi-precision arithmetic and square roots. In Python you could use the [gmpy2](#) module. In C you can use [GMP](#).

Factoring challenge #1: The following modulus N is a products of two primes p and q where $|p - q| < 2N^{1/4}$. Find the smaller of the two factors and enter it as a decimal integer.

```
N = 17976931348623159077293051907890247336179769789423065727343008115 \
    77326758055056206869853794492129829595855013875371640157101398586 \
    47833778606925583497541085196591615128057575940752635007475935288 \
    71082364994994077189561705436114947486504671101510156394068052754 \
    0071584560878577663743040086340742855278549092581
```

Factoring challenge #2: The following modulus N is a products of two primes p and q where $|p - q| < 2^{11} N^{1/4}$. Find the smaller of the two factors and enter it as a decimal integer.

Hint: in this case $A - \sqrt{N} < 2^{20}$ so try scanning for A from \sqrt{N} upwards, until you succeed in factoring N .

```
N = 6484558428080716696628242653467722787263437207069762630604390703787 \
    9730861808111646271401527606141756919558732184025452065542490671989 \
    2428844841839353281972988531310511738648965962582821502504990264452 \
    1008852816733037111422964210278402893076574586452336833570778346897 \
    15838646088239640236866252211790085787877
```

Factoring challenge #3: (extra credit) The following modulus N is a products of two primes p and q where $|3p - 2q| < N^{1/4}$. Find the smaller of the two factors and enter it as a decimal integer.

Hint: use the calculation below to show that $\sqrt{6N}$ is close to $\frac{3p+2q}{2}$ and then adapt the method above to factor N .

```
N = 72006226374735042527956443552558373833808445147399984182665305798191 \
63556901883377904234086641876639384851752649940178970835240791356868 \
77441155132015188279331812309091996246361896836573643119174094961348 \
52463970788523879939683923036467667022162701835329944324119217381272 \
9276147530748597302192751375739387929
```

The only remaining mystery is why $A - \sqrt{N} < 1$. This follows from the following simple calculation. First observe that

$$A^2 - N = \left(\frac{p+q}{2}\right)^2 - N = \frac{p^2+2N+q^2}{4} - N = \frac{p^2-2N+q^2}{4} = (p-q)^2/4$$

Now, since for all x, y : $(x-y)(x+y) = x^2 - y^2$ we obtain

$$A - \sqrt{N} = (A - \sqrt{N}) \frac{A + \sqrt{N}}{A + \sqrt{N}} = \frac{A^2 - N}{A + \sqrt{N}} = \frac{(p-q)^2/4}{A + \sqrt{N}}$$

and since $\sqrt{N} \leq A$ it follows that

$$A - \sqrt{N} \leq \frac{(p-q)^2/4}{2\sqrt{N}} = \frac{(p-q)^2}{8\sqrt{N}}$$

By assumption (*) we know that $(p-q)^2 < 4\sqrt{N}$ and therefore

$$A - \sqrt{N} \leq \frac{4\sqrt{N}}{8\sqrt{N}} = 1/2$$

as required.

Further reading: the method described above is a greatly simplified version of a much more general [result](#) on factoring when the high order bits of the prime factor are known.

Enter the answer for factoring challenge #1 in the box below:

Question 2

Enter the answer for factoring challenge #2 in the box below:

Question 3

Enter the answer for factoring challenge #3 in the box below:

Question 4

The challenge ciphertext provided below is the result of encrypting a short secret ASCII plaintext using the RSA modulus given in the first factorization challenge. The encryption exponent used is $e = 65537$. The ASCII plaintext was encoded using PKCS v1.5 before the RSA function was applied, as described in [Lecture 11.4](#).

Use the factorization you obtained for this RSA modulus to decrypt this challenge ciphertext and enter the resulting English plaintext in the box below. Recall that the factorization of N enables you to compute $\varphi(N)$ from which you can obtain the RSA decryption exponent.

Challenge ciphertext (as a decimal integer):

```
2209645186741038177630656113488341801741006978789283107173183914367613560012053800
4282329650473509424343946219751512256465839967942889460764542040581564748988013734
8641204523252293201764879166664029975091887299716905260832220677716000193292608700
09579993724077458967773697817571267229951148662959627934791540
```

After you use the decryption exponent to decrypt the challenge ciphertext you will obtain a PKCS1 encoded plaintext. To undo the encoding it is best to write the decrypted value in hex. You will observe that the number starts with a '0x02' followed by many random non-zero digits. Look for the '0x00' separator and the digits following this separator are the ASCII letters of the plaintext. (note: the separator used here is '0x00', not '0xFF' as stated in the lecture)

☐ In accordance with the Coursera Honor Code, I (Antonio Diaz Arroyo) certify that the answers here are my own work.

[Submit Answers](#)[Save Answers](#)

You cannot submit your work until you agree to the Honor Code. Thanks!