Class 29: Trick-or-Treat Protocols

Upcoming Schedule

- **Note:** I mistakenly listed two different dates for when you should read Tyson's *Golden Age of Science*. I will talk about it in class on Wednesday. It would definitely be beneficial if you can read it before then.
- **Thursday's office hours:** since I'm out of town Thursday, Peter Chapman will cover my office hours, 9:45-11am in Rice Hall bagel shop area (instead of my office). All the other normal office hours will be held this week:
  Monday: noon-1:30pm (Kristina, Rice 1st); 1:15-2:00pm (Dave, Rice 507)
  Tuesday: 11am-noon (Dave, Rice 507); 5-8pm (Valerie/Jonathan, Rice 1st)
  Wednesday: 5-6:30pm (Jiamin, Rice 1st)
  Thursday: 9:45-11am (Peter, Rice 1st); 1-2:30pm (Joseph, Rice 1st); 4:30-6pm (Jonathan, Rice 1st); 6-7:30pm (Jiamin, Rice 1st)
  Fridays, noon-1:30pm (Peter, Rice 1st)
- **Monday, 7 November:** Problem Set 6

One-Way Function

A one-way function, $f(x) = y$, is a function that is:

Invertible: there exists a function f^{-1} such that $f^{-1}(f(x)) = x$ for all x.

One-way: it is much, much, much easier to compute $f(x)$ than to compute $f^{-1}(y)$.

Factoring

```
(define (factors n)
  (list-reverse (factors-helper (- n 1) n)))
(define (factors-helper t n)
  (if (< t 2) null
      (if (is-divisible? n t)
          (cons t (factors-helper (- t 1) n))
          (factors-helper (- t 1) n))))
```

```
def factors(n):
  res = []
  for d in range(2, n):
    if n % d == 0:
      res.append(d)
  return res
```

What is the running time of multiplication?

What is the running time of the **factors** procedure?

Does this prove that factoring is hard? (and that multiplication is a one-way function)

RSA Encryption System
Developed by Ron Rivest, Adi Shamir, and Len Adelman in 1978.

This was the first publicly* known "*public key* cryptosystem". A public key cryptosystem is a cryptosystem where the *encryption* and *decryption* keys are different.

> * Both GHCQ (United Kingdom, successor to Bletchley Park) and NSA (USA) claim to have secretly developed public-key cryptosystems before RSA.

$E(M) = M^e \bmod n$
$D(C) = C^d \bmod n$

$n = pq$ p, q are prime
d is relatively prime to $(p - 1)(q - 1)$
$ed \equiv 1 \pmod{(p - 1)(q - 1)}$

Public key: e, n
> Anyone can obtain this from the "Tricker's Bureau" (Certificate Authority)

Private key: d
> This is the "trap-door" that makes it easy for the owner of the private key to reverse the one-way function.

For more on RSA, see Kate McDowell's **rsacrypto.org** site!

Homework: next time you see https in your web browser, click on the lock to see the certificate. See how much you can figure out about what is in it.

Objects and Inheritance

An **object** packages state and procedures.

A **class** provides procedures for making and manipulating a type of object.
The procedures for manipulating objects are called **methods**. We *invoke* a method on an object.

Inheritance allows one class to refine and reuse the behavior of another.

In Python we can use inheritance by creating a new class that is a subclass of an existing class:

> *ClassDefinition* ::= **class** *SubClassName* **(** *SuperClassName* **) :**
> *FunctionDefinitions*