Cyclone is a symmetric cryptosystem inspired loosely by the famous Enigma machine. The $n \times n$ square matrix key is best thought of as a cylinder of stacked wheels.

Each row or a "wheel" of the key is a permutation on $\{1, 2, 3, ..., n\}$, which is compactly expressed in the abbreviated form of $[f(1) \quad f(2) \quad ...f(n-1) \quad f(n)]$.

Let f_i be the permutation of row i. So the key gives us $f_1, ..., f_n$.

Then $f = f_1 \circ f_2 \circ ...f_{n-1} \circ f_n$ is *itself* an [invertible] permutation, and it is this f which is used to transform a symbol of plaintext into a symbol of ciphertext. Next, this f is circularshifted by the value of the plaintext symbol which was just its argument. Finally, each row of the key is itself shifted using this shifted f. In other words, row i (the description of f_i) is circularly rightshifted by the already-plaintext-shifted $f(i)$.

Here is the function for encoding:

```
function encode(p,q)
    k= copy(q)
    n = size(k)[begin]
    c = zeros(Int64,length(p))
    for i in 1:length(p)
        f = getf(k)
        c[i] = f[p[i]]
        f = circshift(f,p[i])
        for j in 1:n
            k[j,:] = circshift(k[j,:],f[j] )
        end
    end
    c
end
```

Note that this is not necessarily the most efficient code. It was written for ease of understanding.

In the decoding function, we need to invert f to recover the plaintext from the ciphertext, but it's otherwise the same. Note that decoding requires an exact repetition of all the changes of the key as a function of plaintext.

```
function decode(c,q)
    k= copy(q)
    n = size(k)[begin]
    p = zeros(Int64,length(c))
    for i in 1:length(c)
        f = getf(k)
        g = inverse(f)
        p[i] = g[c[i]]
        f = circshift(f,p[i])
        for j in 1:n
            k[j,:] = circshift(k[j,:],f[j] )
        end
    end
    p
end
```

This is the essence of the algorithm, but the encrypt function uses features like autospin and reversal to make an arbitrary number of rounds (hopefully) more effective.

When $n = 27$, the compositions f tend to be unique. So the key constantly wanders into a new state, meeting each symbol of plaintext with a fresh permutation. For $n \approx 10$, results are still pretty good, but not perfect.

DISCLAIMER

I'm not a professional cryptographer. I just took some independent studies and grad school. This is one of a few ideas that occurred to me at that time. If an actual cryptographer has an opinion about the security of this system, I'd be glad to hear it. To me it *seems* reasonably strong for $n = 27$, but I can't be sure.