Eric Jiang

PHP Matrix Encryption and Decryption Program

Provides 2×2 Matrix Encryption of Strings

The program encrypts text strings by taking each pair's ASCII code, and then multiplying it by a matrix. "A" is 65.

```
H   E   L   L   O   _

72  69  76  76  79  32
```

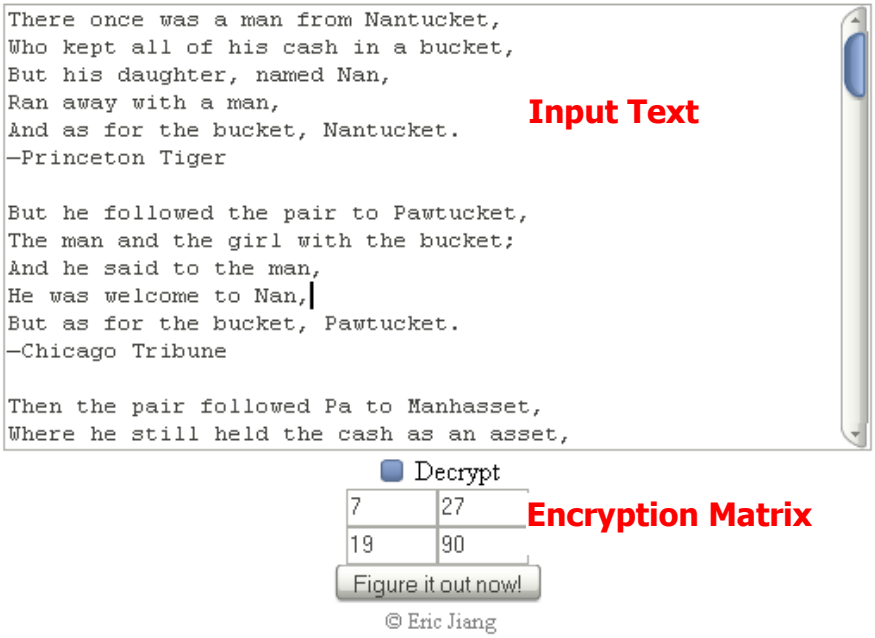
Each pair is converted into a matrix and multiplied by the key matrix.

$$\begin{bmatrix}72 & 69\end{bmatrix}\cdot\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

Which gives $\begin{bmatrix}279 & 420\end{bmatrix}$. Decoding is done by multiplying the encrypted pairs by the inverse of the key matrix.

Encryption Process

The user is presented with a prompt like this:



Text is typed into the text area, and four numbers that make up a 2×2 matrix are provided.

The matrix numbers are known as $mat1, $mat2, and so on.

```
$matrix = Array(Array($mat1,$mat2),Array($mat3,$mat4));
```

If there is no text inputted, then the program just prints “Put text here”.

```
if (!$text) echo "Put text here";
```

If there is data put in, then we’ll continue encrypting. The text should have an even number of characters, so if it’s odd (checked by seeing if mod 2 is more than 1), we add a blank space on the end.

```
if ((strlen($text) % 2) > 0) $text .= " ";
```

We check to make sure the “Decrypt” check box is cleared, and then continue. We’ll get pairs of characters, so we’ll jump by twos in the string (`$i+=2`).

```
for ($i = 0; $i < strlen($text); $i+=2) {
   codeit(hexdec(bin2hex(substr($text,$i,1))),hexdec(bin2hex(sub
str($text,$i+1,1))),$matrix);
```

We do `hexdec(bin2hex(substr($text,$i,1))`, because we need to convert each character into its corresponding ASCII number in decimal (A starts at 65). It passes the value of each letter in the pair to the function `codeit`, which will output the encrypted numbers.

The function `codeit` accepts three arguments: `$val1`, `$val2`, and `$mtx` (and optionally a fourth for decryption). They’re the first letter value, second letter value, and the key matrix, respectively.

It manually goes through multiplying the two matrices, because there are no built-in matrix functions.

Decryption Process

The decryption process is essentially finding the inverse of the key matrix, and then going through the same multiplication process as the encryption process.

It finds all the numbers, separated by spaces, with a regular expression.

```
preg_match_all('/[\d\-\.E\+]+/',$text,$matches);
```

It includes digits, dashes (for negative numbers), decimal points, and “E” and “+” for large numbers that require scientific notation.

Then, it goes through the process of finding the inverse of the matrix. It first finds the fraction that the matrix will be multiplied by.

if $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ then $\$\text{frac} = \frac{1}{(ad-bc)}$

We then make a new matrix, called `$inmtx`, that is $\$\text{frac} \cdot \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$.

The function `codeit` is also used for decoding, since the two processes are very much alike. A fourth argument is passed to `codeit`, which is simply `true`. The function multiplies the matrix, and converts the resulting numbers back into ASCII format:

```
hex2asc(dechex($temper2));
```

Matrix Encryption Advatages

Matrix encryption has an advantage over per-character encryption. Characters will come out differently depending on what other character it's paired with. For example, “EA” may come out as [264 398], but “EZ” might be [339 498]. This way, an encrypted message cannot be cracked by simple statistical analysis.

Credits

This program was written in PHP with ASCII conversion functions from http://www.php.net. The current version can be found at http://www.aztekera.com/tools/matrix.php/.

Last revised: 26 August 2007