MEA

Maltes Encryption Algorithm

# V1

Status: Unsicher + Gelöscht

## Funktionsweise

Buchstaben des Einganswertes werden mit den Buchstaben des Schlüssels addiert.  
Dabei werden die Keycodes genutzt.

## Probleme

1. Jeder Buchstabe wurde um den gleichen Wert erhöht, was dazu führt, dass man raten kann.

# V2

Status: Entwickelt; In Prüfung

## Funktionsweise

Verschlüsseln:

1. Input und Secret In Binärcodes umwandeln
2. Input tauschen:  
   B2 B4 B6 B8 B1 B3 B5 B7  
   (B steht für Bit)
3. Jeden Bit XOR Abfragen mit dem n‘ten Zeichen aus dem Secret
4. Schritt 2 und 3 für jedes Zeichen n des Secret wiederholen
5. Binär in String umwandeln

Entschlüsseln:

1. Input und Secret in Binärcodes umwandeln
2. XOR Rückgängig machen für das n’te Zeichen des Passworts (beginnend am Ende)
3. Input - Bits drehen:  
   B5 B1 B6 B2 B7 B3 B8 B4
4. 2 und 3 für jedes Zeichen n des Secret wiederholen
5. Binär in String umwandeln

(Code auf nächster Seite)

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

A – Input | B – Secret (Links XOR; Rechts umgekehrtes XOR (sind gleich idk))

## Sonstiger Code:

String toBinary(String in) {

String out = "";

for(int i = 0; i < in.length(); i++) {

out += binary(in.charAt(i), charBits);

}

return out;

}

String fromBinary(String in) {

String out = "";

for(int i = 0; i < in.length(); i+=charBits) {

out += (char)unbinary(in.substring(i, i+charBits));

}

return out;

}

## Dokumentation Sonstiger Code:

String toBinary(String in)

Wandelt einen herkömmlichen String in einen Binary String um

String fromBinary(String in)

Wandelt einen Binary-String in einen herkömmlichen String um

## Code Verschlüsselung:

String encrypt(String in, String secret) {

//Step 1

in = toBinary(in);

secret = toBinary(secret);

//Step 2 to 4

String out = "";

for (int i = 0; i < in.length(); i+=8) {

String b = in.substring(i, i+8);

for (int j = 0; j < secret.length(); j+=8) {

//Step 2

b = (b.charAt(1) + "") + (b.charAt(3) + "") + (b.charAt(5) + "") + (b.charAt(7) + "") +

(b.charAt(0) + "") + (b.charAt(2) + "") + (b.charAt(4) + "") + (b.charAt(6) + "");

//Step 3

String s = secret.substring(j, j+8);

String \_b = "";

for (int o = 0; o < 8; o++) {

if (b.charAt(o) == s.charAt(o)) \_b += "0";

else \_b += "1";

}

b = \_b;

}

out += b;

}

//Step 5

out = fromBinary(out);

return out;

}

## Dokumentation Verschlüsselung

String encrypt(String in, String secret)

Verschlüsselt den Text bei in mit dem Schlüssel secret

Aus Sonstigem Code:

toBinary(String in)

fromBinary(String in)

## Code Entschlüsselung:

String decrypt(String in, String secret) {

//Step 1

in = toBinary(in);

secret = toBinary(secret);

//Step 2 to 4

String out = "";

for (int i = 0; i < in.length(); i+=8) {

String b = in.substring(i, i+8);

for (int j = secret.length()-8; j >= 0; j-=8) {

//Step 2

String s = secret.substring(j, j+8);

String \_b = "";

for (int o = 0; o < 8; o++) {

if (b.charAt(o) == s.charAt(o)) \_b += "0";

else \_b += "1";

}

b = \_b;

//Step 3

b = (b.charAt(4) + "") + (b.charAt(0) + "") + (b.charAt(5) + "") + (b.charAt(1) + "") +

(b.charAt(6) + "") + (b.charAt(2) + "") + (b.charAt(7) + "") + (b.charAt(3) + "");

}

out += b;

}

//Step 5

out = fromBinary(out);

return out;

}

## Dokumentation Entschlüsselung:

String decrypt(String in, String secret)

Entschlüsselt den Text in mit dem Schlüssel secret

Aus Sonstigem Code:

toBinary(String in)

fromBinary(String in)

## Potenzielle Probleme:

1. Jeder Entschlüsselte Buchstabe ist der gleiche.