

Kryptografische Verfahren: Diffie–Hellman–Schlüssel–Vereinbarung Haftendorn Okt 2011

```
erg:=diffi(127,45,10,77) ▶ [ "Sie verabreden p" 127 "Sie verabreden g" 45
                                "Anton sendet" 84     "Berta sendet" 75
                                "Antons Schüssel" 52   "Bertas Schlüssel" 52 ]
```

```
isPrime(erg[1,2]) ▶ true
```

Diese Datei zeigt das **Diffie–Hellman–Verfahren**.

Die Umwandlung von Zahlen in Ziffernlisten und zurück

```
zahl2li(192837465) ▶ {1,9,2,8,3,7,4,6,5} li2zahl({1,9,2,8,3,7,4,6,5}) ▶ 192837465
```

und das **OneTimePad–Verfahren** für Ziffernlisten mit verschlüsseln und entschlüsseln.

```
onetimepad({1,2,3,4,5,6,7},{1,9,2,8,3,7,4,6,5}) ▶ {2,1,5,2,8,3,1}
```

```
onetimeinv({2,1,5,2,8,3,1},{1,9,2,8,3,7,4,6,5}) ▶ {1,2,3,4,5,6,7}
```

Man kann auch alles kombinieren:

```
diffiehell:=diffi(kry\nextprime(randInt(1000000000,1000000000000000),randInt(10000000,
100000000000),randInt(10000000,10000000000),randInt(10000000,10000000000))
```

```
▶ [ "Sie verabreden p" 15516873338011 "Sie verabreden g" 4092678947
      "Anton sendet" 4537567508925     "Berta sendet" 1985139218870
      "Antons Schüssel" 8949574962958   "Bertas Schlüssel" 8949574962958 ]
```

```
s:=diffiehell[3,2] ▶ 8949574962958 sli:=zahl2li(s) ▶ {8,9,4,9,5,7,4,9,6,2,9,5,8}
```

```
otp:=onetimepad(zahl2li(112233445566),sli) ▶ {9,0,6,1,8,0,8,3,1,7,5,1}
```

```
oti:=onetimeinv(otp,sli) ▶ {1,1,2,2,3,3,4,4,5,5,6,6} Ende: li2zahl(oti) ▶ 112233445566
```

Die Message 112233445566 kommt am Ende wieder heraus.

```
#####
#####
```

```
diffi(13,2,5,4) ▶ [ "Sie verabreden p" 13 "Sie verabreden g" 2
                      "Anton sendet" 6     "Berta sendet" 3
                      "Antons Schüssel" 9   "Bertas Schlüssel" 9 ]
```

```
diffi(123457,5678,1023,5001) ▶ [ "Sie verabreden p" 123457 "Sie verabreden g" 5678
                                      "Anton sendet" 42332   "Berta sendet" 80229
                                      "Antons Schüssel" 68591  "Bertas Schlüssel" 68591 ]
```



diffi(kry\nextprime(75247388436868349139),55665526616813868,10248276867348633,57657657001)

"Sie verabreden p"	75247388436868349167	"Sie verabreden g"	55665526616813868
"Anton sendet"	48122236671797693468	"Berta sendet"	8210090613061314686
"Antons Schüssel"	69377184492155382014	"Bertas Schlüssel"	69377184492155382014

10^{14} ist maximale Größe für randint, 10^{14} funktioniert nicht als Eintrag (Fehler gemeldet Okt. 2011)

```
diffi(kry\nextprime(randInt(10000000000,1000000000000000)),randInt(10000000,10000000000),  
randInt(10000000,10000000000),randInt(10000000,10000000000))
```

"Sie verabreden p "	26106014449211	"Sie verabreden g "	5568859236
"Anton sendet "	317823024896	"Berta sendet"	18511942563980
"Antons Schüssel "	11267699134900	"Bertas Schlüssel"	11267699134900

Spielwiese

```
erg ▶ [{"Sie verabreden p " 127 "Sie verabreden g " 45}, {"Anton sendet " 84 "Berta sendet" 75}, {"Antons Schüssel " 52 "Bertas Schlüssel" 52}] isPrime(erg[1,2]) ▶ true
```

4/10

Define LibPub **diffi**(p, g, a, b)=

Func

Local α,β,mat

$\alpha := \text{kry}\backslash pmod(g, a, p)$; $\beta := \text{kry}\backslash pmod(g, b, p)$

```
mat:=newMat(3,4)
```

$\text{mat}[1,1]:=\text{"Sie verabreden p "};\text{mat}[1,3]:=\text{"Sie verabreden g "}$

```
mat[1,2]:=p;mat[1,4]:=g;
```

$\text{mat}[2,1]:=\text{"Anton sendet "}: \text{mat}[2,3]:=\text{"Berta sendet "}$

$\mathtt{mat[2,2]} := \alpha; \mathtt{mat[2,4]} := \beta;$

$\text{mat}[3,1] := \text{"Antons Schüssel "}$; $\text{mat}[3,3] := \text{"Bertas Schlüssel "}$

$\text{mat}[3,1] := \text{kry}\backslash pmod(\beta, q, p)$: $\text{mat}[3,4] := \text{kry}\backslash pmod(\alpha, b, p)$

Return $\text{mat}[3,2]$: key $\text{pmob}(\beta,a,p).\text{mat}[3,1]$: key $\text{pmob}(\alpha,b,p)$

[Return mail](#)

diffi(19,13)

	Sie verabreden p		Sie verabreden g	
diffi(19,13,7,10)	"Anton sendet"	10	"Berta sendet"	6
	"Antons Schüssel"		"Bertas Schlüssel"	
	9		9	

"Sie verabreden p "	40370526855713	"Sie verabreden g "	7294570375
"Anton sendet "	26292241391196	"Berta sendet"	975849381668
"Antons Schüssel "	27622376416869	"Bertas Schlüssel"	27622376416869

* onetimepad 8/14

```

Func
© (message, key) -> (cryptogramm)
© alles als Ziffernlisten
Local dif,k
k:=key
Loop
dif:=dim(mess)-dim(k)
If dif=0 Then
Return mod(mess+k,10)
ElseIf dif<0 Then
k:=left(k,dim(k)+dif)
ElseIf dif>0 Then
k:=augment(k,k)
EndIf
EndLoop
EndFunc

```

onetimeinv({3,4,2,1,7,5},{1,9,2,8,3,7})
 $\rightarrow \{2,5,0,3,4,8\}$

Wichtig für das Programm ist,
 $\text{mod}(mess+k,10)$

Die Modulo-Funktion ist "listable", sie arbeitet auf den Elementen der Liste einzeln.

Ebenso kann man Listen elementweise mit + addieren.
 $\{1,2,3\} + \{10,20,30\} \rightarrow \{11,22,33\}$
 $\text{mod}(\{11,22,33\},7) \rightarrow \{4,1,5\}$

Die anderen Programmteile dienen dazu, die

onetimepad({2,5,0,3,4,8},{1,9,2,8,3,7,4,6}) $\rightarrow \{3,4,2,1,7,5\}$ Schlüssel länger
onetimeinv({3,4,2,1,7,5},{1,9,2,8,3,7,4,6}) $\rightarrow \{2,5,0,3,4,8\}$
onetimepad({2,5,0,3,4,8},{1,9,2}) $\rightarrow \{3,4,2,4,3,0\}$ Schlüssel kürzer
onetimeinv({3,4,2,4,3,0},{1,9,2}) $\rightarrow \{2,5,0,3,4,8\}$

onetimeinv 8/14

```

Func
©(cryptogramm, key) -> message
© alles als Ziffernlisten
Local dif,k
k:=key
Loop
dif:=dim(cryp)-dim(k)
If dif=0 Then
Return mod(cryp-k,10)
ElseIf dif<0 Then
k:=left(k,dim(k)+dif)
ElseIf dif>0 Then
k:=augment(k,k)
EndIf
EndLoop
EndFunc

```

onetimeinv({3,4,2,1,7,5},{1,9,2,8,3,7}) $\rightarrow \{2,5,0,3,4,8\}$

OneTimePad invers Kryptogramm und Schlüssel als Ziffernlisten.

onetimeinv({3,4,2,1,7,5},{1,9,2,8,3,7,4,6}) Schlüssel länger

onetimepad({2,5,0,3,4,8},{1,9,2}) $\rightarrow \{3,4,2,4,3,0\}$

Wenn der Schlüssel kürzer war, ist auch das Cryptogramm anders.

onetimeinv({3,4,2,4,3,0},{1,9,2}) $\rightarrow \{2,5,0,3,4,8\}$

zahl2li

6/9

```
Define LibPub zahl2li(zahl)=  
Func  
© zahl->Ziffernliste  
Local i,z,li  
li:= $\{ \}$   
z:=zahl  
While z>0  
li:=augment( $\{ \text{mod}(z,10) \}$ ,li)  
z:=floor $\left(\frac{z}{10}\right)$   
EndWhile  
Return li  
EndFunc
```

zahl2li(112233445566)► $\{ 1,1,2,2,3,3,4,4,5,5,6,6 \}$ **li2zahl($\{ 1,1,2,2,3,3,4,4,5,5,6,6 \}$)**

► 112233445566

li2zahl

4/8

```
Define LibPub li2zahl(liste)=  
Func  
© (Ziffernliste)-> Zahl  
Local z,li,i  
li:=liste: z:=0  
For i,1,dim(liste)  
z:=z*10+li[1]  
li:=mid(li,2)  
EndFor  
Return z  
EndFunc
```

li2zahl($\{ 2,5,0,3,4,8 \}$) ► 250348**zahl2li(250348)** ► $\{ 2,5,0,3,4,8 \}$