**htw saar**

Studiengang Kommunikationsinformatik (Master)
Studiengang Praktische Informatik (Master)
Prof. Dr.–Ing. Damian Weber

# Sicherheit und Kryptographie – Praktische Übung 1

### Aufgabe 1 (GP/PARI Basics)

Starten Sie eine GP/Pari-Instanz, vorzugsweise auf einem UNIX–System.

Der Link zum User's Manual ist

`http://pari.math.u-bordeaux.fr/pub/pari/manuals/2.3.3/users.pdf`

Eine online–Hilfe zu einem Befehl oder einer Funktion erhält man mit

```
?

?isprime

?for

?if
```

a) Führen Sie mit den Operatoren `+-*/%^` einige arithmetische Operationen durch, belegen Sie Variablen, rechnen Sie Klammerausdrücke.

b) Rechnen Sie arithmetische Operationen mit Brüchen.

c) Testen Sie je ein Beispiel für die Funktionen
`min(),max(),ceil(),frac(),truncate(),exp(),log(),sqrt(),sqrtn()`

d) Benutzen Sie `divrem()`, um für die Division von 70 durch 11 gleichzeitig den Quotient und Rest zu finden.

e) Multiplizieren Sie die Polynome

$$f(X) = X^5 + X^2 + 1, \quad g(X) = 2 * X^3 - X - 1$$

f) Berechnen Sie $f(2)$ und $g(-1)$.

g) Bilden Sie die Ableitungen von $f$ und $g$. *Hinweis:* Funktion `deriv()`

h) Finden Sie die Nullstellen von $f$ und $g$. *Hinweis:* Funktion `polroots()`

i) Finden Sie die Nullstellen von $f$ und $g$ modulo 23. *Hinweis:* Funktion `polrootsmod()`

j) Wählen Sie zufällige 500-bit-Zahlen.
*Hinweis:* Funktion `random()`

k) Wählen Sie die nächste Primzahl $> 100000$.
Wählen Sie die nächste Primzahl $> 237901$.
*Hinweis:* Funktion `nextprime()`

l) Wählen Sie eine zufällige 500-bit-Primzahl (`random(), nextprime()`).

m) Prüfen Sie, ob 70709, 70711, 70713,70717,70719 Primzahlen sind.
*Hinweis:* Funktion `isprime()`

n) Berechnen Sie von denen, die keine Primzahlen sind, die Primfaktorzerlegung.
*Hinweis:* Funktion `factor()`

o) Geben Sie die 100-ste Primzahl aus.
*Hinweis:* Funktion `prime()`

p) Erzeugen Sie zwei Vektoren und berechnen Sie das Skalarprodukt.

q) Erzeugen Sie zwei $2 \times 2$ Matrizen und berechnen Sie das Matrixprodukt.

r) Berechnen Sie die Determinante Ihrer Matrizen.

s) Invertieren Sie Ihre Matrizen und prüfen Sie durch Multiplikation, dass wirklich die inverse Matrix gefunden wurde.

t) Lösen Sie ein Gleichungssystem mit `matsolve()`.

u) Setzen Sie die Präzision auf 200 Dezimalstellen mittels

```
default(realprecision, 200)
```

Geben Sie nun $\pi$ mit `Pi` und die Eulersche Zahl $e$ aus und berechnen Sie

$$\frac{1}{7}, \frac{1}{13}, \frac{5678}{9999}, \frac{13717421}{111111111}$$

auf 200 Dezimalstellen.

v) Berechnen Sie die Anzahl der Möglichkeiten beim Lotto.
*Hinweis:* Funktion `binomial()`

w) Berechnen Sie den größten gemeinsamen Teiler $d$ und das kleinste gemeinsame Vielfache $m$ von $a = 3066$ und $b = 4074$. Vergleichen Sie $d \cdot m$ und $a \cdot b$. Haben Sie eine Erklärung für die Beobachtung?

**Aufgabe 2 (GP/PARI Sichere Primzahlen)**

Nutzen Sie eine Schleife und eine Verzweigung, um sichere Primzahlen erzeugen zu können. Diese erschweren Attacken gegen das Diffie–Hellman-Protokoll (und auch gegen RSA), indem Primzahlen $p$ gesucht werden, bei denen $(p-1)/2$ ebenfalls eine Primzahl ist.

Nebenbemerkung: Man kann dies auch umgekehrt definieren, ist $q$ eine Primzahl und $2q+1$ ebenfalls eine Primzahl, dann heißt $q$ Sophie–Germain–Primzahl (und $2q+1$ ist dann eine sichere Primzahl).

Gehen Sie schrittweise vor, erst kleine Schritte ausprobieren (z.B. wie funktionieren `for, if, ...`) und später erst die Erkenntnisse zusammensetzen.

**Aufgabe 3 (GP/PARI Pollard-$\varrho$)**

Ein Problem des diskreten Logarithmus sei in der Form

$$g^x \equiv h \bmod p$$

gegeben.

Programmieren Sie die Pollard-$\varrho$–Folge, sodass Sie auf dem Bildschirm die Gleichheit zweier Folgenglieder sehen können und daraus den Logarithmus $x$ ausrechnen können.

Gehen Sie wieder schrittweise vor, erst alle kleinen Schritte testen, später erst die Erkenntnisse zusammensetzen.

Hinweise:

- Rechnen Sie mit Elementen der Form `\Mod(g,p)`.
- Wenn Sie von `Mod(g,p)` den Wert `g` erhalten wollen, benutzen Sie `lift()` (etwa um mod 3 zu bestimmen, welcher Fall der Zahlenfolge eintritt).
- Schleifen werden mit `for` gebildet, die Syntax

  ```
  for(X=a,b,seq): the sequence is evaluated, X going from a up to b.
  ```

  wobei `seq` eine Sequenz von Kommandos darstellt.
- Verzweigungen werden mit `if` gebildet.

  ```
  if(a,seq1,seq2):
  if a is nonzero, seq1 is evaluated, otherwise seq2.
  seq1 and seq2 are optional, and if seq2 is omitted,
  the preceding comma can be omitted also.
  ```

- Lösen Sie mit Ihrem Code die Probleme

  a) $5^x \equiv 7 \bmod 263$

  b) $2^x \equiv 3 \bmod 347$

  c) $7^x \equiv 8 \bmod 359$

  erst durch Reduzierung mit der Pohlig–Hellman-Methode auf die Primzahlteiler $q$ von $p-1$, dann mit der Pollard–Folge, die auf das auf $q$ reduzierte Problem angewandt wird. In den Fällen $q = 2$ kann das Ergebnis direkt abgelesen werden.