LF09:09:Layer II/II: IPv4-Adressen und -Netze

1) IPv4-Adresse [→ ZP:Sheet:2]

- bestehen aus 4 Bytes,
- die jeweils dezimal notiert und mit Punkt vom Folgebyte getrennt
- und mit abnehmender Wertigkeit von links nach rechts ausgegeben werden

IPv4-Adresse	B4	B3	B2	B1
als String:	”192.	168.	0.	0”
byteweise dezimal	192	168	0	0
byteweise hexadezimal	0xC2	0xA8	0x00	0x00
byteweise binär	0b11000000	0b10101000	0b00000000	0b00000000

ergibt IPv4-Adressstring: ”192.168.0.0”

[→ ZP:Sheet:3]

- Zahlenmäßig stünden mit 4 Bytes ~ 4.3 Milliarden IPv4-Adressen zur Verfügung

```
2^8 * 2^8 * 2^8 * 2^8
= 2^(8+8+8+8)
= 2^32
= 4.294.967.296
```

Kontextfrage: [→ ZP:Sheet:4]

Warum wären folgende IPv4-Adressangaben falsch:

- 19216800 (weil kein Punkt)
- C2.A8.0.0 (weil Hexangaben unzulässig)
- 192.168.320.0 (weil 320 den Maximalwert 255 eines Byts übersteigt)

ÜBUNG **LF09:09:IPv4-Adresses:01**

- ☐ Ermitteln Sie die IP-Adresse Ihres Rechners.
- ☐ Vergleichen Sie die mit Ihren Mitschülern.
- ☐ Welche Zusammenhänge erkennen Sie?

Lösung: Ohne Infos zur Netzsegmentierung (Subnetzmaske) sagt die IPv4-Adresse allein nichts.

2) IPv4-Subnetz [→ ZP:Sheet:5]

Nur IP-Adresse + Subnetzmaske zusammen liefern Informationen über das Netz.

- Technisch gesehen ist die Subnetzmaske die Zahl, die an allen Bitpositionen, die zum Netzteil gehören, ein Bit gesetzt hat.
- Aus der bitweisen Verknüpfung (Verunderung) der IPv4-Adresse und der Netzmaske ergibt sich die Netzadresse.
- Aus der Newtzadresse lassen sich Broadcast- und Gatewayadresse ableiten:

Typ	B4	B3	B2	B1
IPv4-Adresse/d	192	168	0	42
Subnetzmaske/d	255	255	255	0
`--------`	`--------`	`--------`	`--------`	`--------`
IPv4-Adresse/b	11000000	10101000	00000000	00101010
&Subnetzmaske/b	11111111	11111111	11111111	00000000
`-------`	`--------`	`--------`	`--------`	`--------`
= Netzadresse/b	11000000	10101000	00000000	00000000
Netzadresse/d	192	168	0	0
Broadcastadresse/d	192	168	0	255
Gatewayadresse/d	192	168	0	1*

*) oder 254

3) Exkurs: Umwandlung von Dezimalzahlen in Binärzahlen und umgekehrt [→ ZP:Sheet:6]

dec → bin:

- Teilen Sie die Dezimalzahl fortlaufend durch 2 und notieren Sie Ergebnis und Rest.
- Wiederholen Sie dies, bis das Divisions**ergebnis** 0 ist.

- Schreiben Sie der Restzahlen - von unten nach oben gelesen - von links nach rechts.
- Stellen Sie soviele Nullen voran, bis Sie 8 (bzw. 16,24,32,…) Stellen in der Binärzahl haben.

bin → dec:

- Schreiben Sie die Bitzahl von links nach rechts auf.
- Numerieren Sie darüber die einzelnen Bitpositionen von rechts nach links hochzählend durch.
- Gehen Sie die Reihe von links nach rechts durch:
  - Ist an einer Bitposition kein Bit gesetzt, notieren Sie "`0*2^Bitposition`".
  - Ist an einer Bitposition ein Bit gesetzt, notieren Sie "`1*2^Bitposition`".
  - Rechnen Sie alle "`1 * 2^Bitposition`"-Formeln aus.
  - Addieren Sie alle "`1 * 2^Bitposition`"-Ergebnisse.

Hinweis:

- 45 ergäbe binär `101101` – ein Palindrom: erschwerte die 'Rückwärtserklärung'
- 42 – die *Weltsuperzahl* – ergäbe binär `101010`, kein Prolem bei der 'Rückwärtserklärung'.

4) IPv4-Netzdefinition [→ ZP:Sheet:7]

- Die Subnetzmaske spaltet die Netzwerkadresse in einen **Netzanteil** und einen **Hostanteil**.
  - Netzadresse :- Netzanteil ○ kleinste Adresse im Hostanteil
  - Broadcastadresse :- Netzanteil ○ größte Adresse im Hostanteil
  - Gatewayadresse :- 'Netzadresse + 1' | 'Broadcastadresse -1' | …
- Die IPv4-Adresse und Subnetzmaske gibt es in 2 Formen:
  - IPv4-Adresse + ausformulierte Subnetzmaske: `192.168.0.0` + `255.255.255.0`
  - IPv4-Adresse + Anzahl der zum Netzanteil gehörenden Bits: `192.168.0.0/24` (=CIDR-Notation) **[→ ZP:Sheet:8]**
- **CIDR** steht für *Classless Inter-Domain Routing*
- Die **CIDR**-Zahl steht für die Anzahl der Bits (von links aus gelesen) , die zur Subnetzmaske gehören

---

ÜBUNG **LF09:09:IPv4-Adresses:02**

☐ Analysieren Sie das Programm auf [→ ZP:Sheet:09]
☐ Was tut es und wie?

Lösung:

Ich hatte Ihnen die Regel gegeben, dass die Netzadresse aus der Ipadresse mittels bitweiser und-Verknüpfung abgeleitet werden kann.

Anhand eines gesetzten Sets an richtigen Netzinformationen überprüft das Programm, ob diese Regel stimmt.

ÜBUNG **LF09:09:IPv4-Adresses:03**

- ☐ Schreiben Sie bitte eine Klasse, die
  - ☐ IPv4-Adresse und die Subnetzmaske im Konstruktur als Strings in Dotted-Notation nimmt und als Eigenschaften abspeichert
  - ☐ die gegebenen IPv4-Strings in Integers umwandelt und als Eigenschaften abspreichert
  - ☐ die aus den Integers die Netzadresse, die Brodcastadresse und eine der üblichen Gateway-adressen mittels Bitweiser Operatoren berechnet
  - ☐ die über getter die Ergebnisse ausgibt
- ☐ Fügen Sie eine Methode `isMember` zur Klasse hinzu, die eine fremde IPv4-Adresse als String nimmt und **`true`** oder **`false`** zurückgibt, je nachdem, ob die übergebene IPv4-Adresse zu dem Netz gehört oder nicht
- ☐ Erlauben Sie im Main-Teil Ihres Programms, dass die drei IPv4-Strings (IPv4-Adresse, Subnetz-maske und Test-IPv4-Adresse) über die Kommandozeile an Ihr Programm übergeben.
- ☐ Implementieren Sie im Main-Teil einen Test mit Refrenzwerten.

Hinweis: Nutzen Sie die Pythonbibliothek ‘ipaddress’ zur Umwandlung von IPv4-Strings in IPv4-Integers. Berechnen Sie die Netzzugehörigkeit und die abgeleiteten Netzwerte jedoch nur mit Bitweisen Operatoren

---

Lösung:

- siehe die letzen 3 Sheets in `cx.py2rf-zenprese.pdf` + zugehörige Erklärungen in `cx.py2rf-oraltrack.pdf`.
- Ersatzweise: Schrittweise Hinführung in ‘cx.py2rf ganz durcharbeiten.
- Ersatzweise: `cx.py2go` über mehrere Wochen 10 Minuten/Tag durcharbeiten.