Wichtig: Lesen Sie auch den Teil "**Hinweise zur Aufgabe**" auf diesem Blatt; Spezifikationen in diesem Teil sind ebenfalls einzuhalten!

Aufgabe 1: snail (14.0 Punkte)

a) Makefile

Schreiben Sie ein Makefile, das aus der Datei `snail.c` ein ausführbares Programm `snail` erstellt. Das Makefile soll Regeln für die beiden Standard-Pseudotargets `all` und `clean` enthalten und ohne eingebaute Regeln funktionieren (`make -rR`). Greifen Sie dabei stets auf Zwischenprodukte (z. B. `snail.o`) zurück. Nutzen Sie die auf der Webseite genannten Compilerflags.

b) SMTP-Client

Schreiben Sie ein SMTP-Client-Programm `snail` (**s**imple **n**atty m**ail**), das zum Versenden von E-Mails über einen fest definierten Server verwendet werden kann. Das Programm wird wie folgt aufgerufen:

```
snail [-s <subject>] <address>
```

Das Programm versendet dann eine E-Mail mit dem optionalen Betreff `subject` an die Empfängeradresse `address` über den SMTP-Server *faui03.cs.fau.de* (Port 25, **getaddrinfo(3)**, **socket(2)**, **connect(2)**). Die Textnachricht (Body-Sektion) der E-Mail wird von der Standardeingabe gelesen (bis EOF).

Die Mail soll die Header-Felder *From*, *To* und optional *Subject* enthalten. Gibt der Benutzer keinen Betreff an, so soll das *Subject*-Feld entfallen. Die Header-Sektion ist durch eine Leerzeile (`\r\n`) vom Body der E-Mail getrennt. Die Kommunikation mit dem Server ist exemplarisch in folgendem Beispieldialog festgehalten (Client fettgedruckt, Server nicht fettgedruckt):

```
220 faui03.informatik.uni-erlangen.de ESMTP Postfix
HELO faui06j.informatik.uni-erlangen.de
250 faui03.informatik.uni-erlangen.de
MAIL FROM:<be15piel@cip.cs.fau.de>
250 2.1.0 Ok
RCPT TO:<hansdampf@example.org>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: Max Mustermann <be15piel@cip.cs.fau.de>
To: <hansdampf@example.org>
Subject: Hallo Hans

Dies ist der Body der Mail. Auch hier werden Zeilen mit \r\n getrennt.
..Diese Zeile beginnt in der Eingabe mit nur einem Punkt.
.
250 2.0.0 Ok: queued as 65AC33E9A9
QUIT
221 2.0.0 Bye
```

Jede Antwort des Servers beginnt mit einem dreiziffrigen Statuscode. Prüfen Sie in jedem Schritt, ob die Antwort des Servers den erwarteten Statuscode enthält (entsprechend obigem Beispiel). Weicht der Statuscode von dem erwarteten Wert ab, soll das Programm die letzte Antwort des Servers ausgeben und sich mit einem Fehler beenden. Treffen Sie keine Annahmen über die maximale Länge der Antwortzeile, die Ihr Programm vom Server empfängt.

Alle an den Server übertragenen Zeilenumbrüche müssen im DOS-Format (`\r\n`) sein. Wenn Zeilenumbrüche auf dem Client-System nur aus einem `\n` bestehen, sind diese in der Übertragung um das fehlende `\r` zu ergänzen. Das Ende des Bodys wird dem Server durch einen Punkt in einer eigenen Zeile signalisiert, abgetrennt mit `\r\n`-Zeilenumbrüchen. Zeilen im Body, die mit einem Punkt beginnen, ist ein weiterer Punkt voranzustellen.

Das Programm soll sowohl IPv4 als auch IPv6 unterstützen. Es muss davon ausgegangen werden, dass der Server DNS-Einträge für beide Protokolle enthält, aber unter Umständen nur über ein Protokoll auf Port 25 erreichbar ist.

Hinweise zur Aufgabe:

- Der Betreff wird immer als **ein** Argument übergeben, selbst wenn er aus mehreren Wörtern besteht – für das entsprechende *Shell-Quoting* (= Betreff in Anführungszeichen setzen) ist der Aufrufer verantwortlich.
- Bevorzugen Sie für die Kommunikation die Bibliotheksfunktionen **fprintf(3)**, **fgets(3)** bzw. **getc(3)**, die mit `FILE *` arbeiten.
- Überlegen Sie, welche Funktionalität sinnvoll in Hilfsfunktionen ausgelagert werden kann, um doppelten Code zu vermeiden!
- Zur Bestimmung des vollständigen Rechnernamens (FQDN) des Client-Rechners bestimmen Sie den Kurznamen mit **gethostname(2)**, gefolgt von einem DNS-Lookup (**getaddrinfo(3)**) mit dem `hints`-Flag `AI_CANONNAME`. Das Feld `ai_canonname` der `addrinfo`-Struktur enthält dann den FQDN, der im `HELO` verwendet werden soll. (max. 255 Zeichen, siehe RFC 5321)
- Die Absenderadresse setzt sich zusammen aus dem lokalen Benutzernamen (**getuid(2)**, **getpwuid(3)**) und `cip.cs.fau.de`. Im *From*-Header soll außerdem der volle Name des Benutzers enthalten sein. Dieser findet sich im Feld `pw_gecos` der `passwd`-Struktur als Teilstring vor dem ersten Komma.

Erforderliche Dateien: `snail.c` (12 Punkte), `Makefile` (2 Punkte)
Bearbeitung: Einzeln
Bearbeitungszeit: 10 Werktage (ohne Wochenenden und Feiertage)
Abgabezeit: 17:30 Uhr