Klausurvorbereitung

Formales

Dauer: 90 Minuten
Modus: schriftlich
Erlaubte Hilfsmittel: Referenzkarte aus dem Buch C von A - Z (wird bei Klausur ausgeteilt), Kugelschreiber und Nachdenken

Beurteilungsmodus: $p_{\max} = 100$ Punkte

Note	Punkte p
1	$87 < p \leq 100$
2	$75 < p \leq 87$
3	$62 < p \leq 75$
4	$50 < p \leq 62$
5	$0 \leq p \leq 50$

Allgemeines

Die Klausur umfasst die gesamten unten angegebenen Themengebiete. Beim ersten Teil der Klausur muss auf Papier ein kompilier- und lauffähiges Programm in C geschrieben werden (ca. 25-35 Punkte). Der zweite Teil umfasst kleinere Programmieraufgaben, Berechnungen und Theoriefragen.

Besonders wichtige Grundlagen-Themen, die auf jeden Fall beherrscht werden müssen, sind mit ★ gekennzeichnet.

Themengebiete

Formale Sprachen

Grundlegendes Verständnis für formale Sprachen, formale Grammatiken und der Begriffe Alphabet, kleenesche Hülle, Terminal- und Nicht-Terminalsymbol und kontextfreier Grammatik.

- Prüfen ob ein Wort einer Grammatik entspricht
- Definition einfacher Grammatiken
- ★ Verstehen und Anwenden der Erweiterten Backus-Naur-Form (EBNF)

Automaten

Verstehen der Anwendung abstrakter Maschinen. Modellierung einfacher Automaten und Umwandlung von Automaten (Turing-Maschine, Keller-Automat, endliche Zustandsautomaten) anhand praktischer Beispiele und/oder mit gegebener formaler Definition.

Grundlagen C

★ Allgemein: Lauffähige Programme in C schreiben.

- Eigenschaften der Programmiersprache C
- Aufbau eines C-Programms
- Präprozessor, Linker und Compiler
- ★ Makros und mögliche Probleme mit Makros
- Darstellung im Zweierkomplement, Berechnen des Zweierkomplements einer Zahl, Addition und Subtraktion mit Zweierkomplement
- Gleitpunkttypen, Umrechnen von Dezimal in Gleitpunktdarstellung nach IEEE 754-1985 und umgekehrt, besondere Interpretationen (0, ∞, etc.)
- ★ Zeiger und Zeigerarithmetik, Erklären von Codezeilen die Zeiger deklarieren oder definieren
- ★ Operatoren (insbes. Ternäroperator und Bitoperatoren)
- ★ Arrays und mehrdimensionale Arrays
- ★ Strings, Stringfunktionen in string.h (insbes. strcat, strcpy, strcmp, strlen)
- ★ Strukturen, Unions (Anwendung, Unterschiede)
- ★ Sichtbarkeit und Lebensdauer von Variablen (local, local static, global, global static)
- ★ Unterschiede zwischen Stack und Heap, dynamischer Speicher, virtueller Speicher
- ★ Belegungsstrategien (First-Fit, Best-Fit, etc.)
- ★ Kontrollstrukturen (Auswertung von Ausdrücken, break und continue, Gefahr bei Verwendung von goto)
- ★ Rekursion, Endrekursion, Umschreiben eines iterativen Algorithmus in eine rekursive oder endrekursive Variante und umgekehrt
- ★ Kommandozeilenargumente
- ★ Verwendung von qsort und bsearch aus der stdlib

- ★ Erzeugen von Pseudozufallszahlen mit bestimmten Eigenschaften und mit beliebiger Verteilung (gegeben die Verteilungsfunktion)
- Optimierendes Programmieren
- Sicheres Programmieren (Buffer Overflow und Memory Leaks)

Algorithmen und Datenstrukturen
Verstehen und Implementieren der folgenden Algorithmen und Datenstrukturen in Pseudocode und ANSI-C. Bewertung der Komplexität dieser Algorithmen und Datenstrukturen für verschiedene Operationen, Vor- und Nachteile in verschiedenen Anwendungsszenarien.

- ★ Stack
- ★ Liste, Skip-Liste
- ★ Queue, Ringbuffer
- ★ Baum und Traversion
- Hash-Table mit Belegungsstrategien
- ★ Suchen und Sortieren

Betriebssysteme
Verstehen der Funktionsweise von Betriebssystemen, deren Aufbau, der wichtigsten Komponenten. Verstehen von Schedulingstrategien und Darstellung an einem Beispiel. Verstehen von Prozessen und Threads und Programmierung in ANSI-C mit POSIX-Systemaufrufen.

- ★ Komponenten (Prozessor, Speicher, Ein-/Ausgabegeräte)
- ★ Bootloader, Kernel (Kernel im Detail: Monolithischer Kernel, Mikrokernel, Hybridkernel)
- ★ Privilegierung und Ringe
- ★ System Call und Kontextwechsel (5 Schritte)
- Prozessspeicher, virtueller Speicher, Primär- und Sekundärspeicher, Cache und Leistung eines Cache-Speichers
- ★ Heap und Stack und statische/dynamische Daten, Heapmanagement
- Swapping
- Virtueller Speicher und Adressumsetzung

- ★ Prozesse und Threads: Unterschied, Parallelverarbeitung
- ★ Tabelle auf Folie “Prozesse I”
- ★ Ablauf beim Prozesswechsel (Folie “Prozesse V”)
- Prozesserzeugung und Prozessbeendigung (Formen)
- ★ Programmieren mit fork, wait, etc. und pthread_create, pthread_join
- ★ Scheduling: Aufgabe und Scheduling-Strategien
- Interprozesskommunikation, Sempahore und Deadlocks