Hochschule Augsburg   Wintersemester 2014/2015
Fakultät für Informatik

Prüfung

Prüfungsfach:	Systemnahe Programmierung
Datum/Uhrzeit:	29. Januar 2015 / 8:30 Uhr
Raum:	wie angeküundigt
Prüfer:	Dr. Hubert Högl
Dauer:	**60** Minuten
Hilfsmittel:	keine

Hinweise:

1. Dieses Angabenblatt hat auch eine **Rückseite**! Bitte sofort überprüfen.
2. **Schreiben** Sie bitte **nicht** auf das **Angabenblatt**. Verwenden Sie für Ihre Antworten die separat ausgeteilten Bögen. **Die Angaben dürfen Sie behalten.**
3. Schreiben Sie **nicht mit Bleistift**.

Viel Glück!

Aufgabe 1 (4 Punkte)

Geben Sie für jede der Sprachen **Assembler, C und Python** Antworten auf die folgenden Fragen:

a) Ist die Sprache **portabel**?

b) Falls portabel, mit **welcher Technik** wird die Portabilität erreicht?

c) Gibt es den Begriff der **Adresse** in der Sprache?

d) Eine Zeile der Sprache entspricht in etwa **wie vielen Maschinenbefehlen**?

Aufgabe 2 (6 Punkte)

Welche **drei Möglichkeiten** gibt es, um in Assembler den **Array Datentyp** zu realisieren? Ein Array sei eine Folge einer bestimmten Anzahl von Elementen mit gleichem Typ. Verdeutlichen Sie jede Variante mit einer kleinen Skizze. Schreiben Sie für jede Variante den **Assembler-Code für die Iteration** über alle Elemente.

```
        # Code noch unvollstaendig!
array:  .long   3, 9, 4, 8
```

Aufgabe 3 (4 Punkte)

Beantworten Sie bitte folgende Fragen zum Stack:

1. Wie heissen die Assemblerbefehle um auf auf den Stack zuzugreifen?

2. Wie implementieren Sie diese Befehle aus elementaren `mov`, `sub` und `add` Befehlen?

Aufgabe 4 (3 Punkte)

Wozu dienen die drei Bereiche `.data`, `.bss` und `.text` in einem Programm:

```
.section .data
...    # was ist hier?
.section .bss
...    # und hier?
.section .text
...    # und hier?
```

Aufgabe 5 (2 Punkte)

Statten Sie die folgende Funktion `add(a, b)` mit einem Prolog und einem Epilog aus. Die Funktion gibt die Summe der beiden Argumente zurück.

```
add:
    movl 8(%esp), %eax
    addl 4(%esp), %eax
    ret
```

Aufgabe 6 (4 Punkte)

Sehen Sie sich das folgende Assemblerprogramm an und beantworten Sie die unten folgenden Fragen dazu:

```
.section .data

array:
   .byte  1, 2, 3, 4, 5, 6, 7, 8, 0

.section .bss
   .lcomm speicher, 8 * 8

.section .text

.globl _start

_start:
    movl $array, %eax
    movl $speicher, %ebx
loop:
```

```
    cmpb $0, (%eax)
    je  exit
    movb (%eax), %cl
    movb %cl, speicher(%ebx)
    addl $1, %eax
    addl $1, %ebx
    jmp  loop
exit:
    movl $1, %eax
    movl $0, %ebx
    int $0x80
```

1. (4 Punkte) Stellen Sie sich die Variable `speicher` als 2-dimensionale Speichertabelle vor mit 64 Einträgen (siehe die folgende Tabelle). Welche Werte stehen in der Tabelle wenn das Programm am Label `exit` angelangt ist?

2. (4 Punkte) Wie müssen Sie das Programm verändern, so dass die Werte aus `array` in die **Diagonale** der Speichertabelle geschrieben werden? Sie haben dazu mit `edx` noch ein Register frei. Schreiben Sie die dazu nötige Änderung in Ihr Lösungsblatt.

Speichertabelle:

```
<speicher>:    0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+8>:  0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+16>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+24>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+32>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+40>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+48>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
<speicher+56>: 0x00   0x00   0x00   0x00   0x00   0x00   0x00   0x00
```

Aufgabe 7 (4 Punkte)

Hier sind einige Fragen zur C Aufrufkonvention:

1. In welcher Reihenfolge werden die Argumente der Funktion `cfun(int a, int b, int c)` auf dem Stack abgelegt?

2. Wie wird der Rückgabewert einer Funktion an den Aufrufer übergeben? Unterscheiden Sie: (a) der Wert ist 32-Bit gross, (b) der Wert ist grösser als 32-Bit.

3. Wer kümmert sich um die Sicherung der Register – der Aufrufer oder der Aufgerufene?

4. Wer korrigiert den Stack, der Aufrufer oder der Aufgerufene?

Aufgabe 8 (4 Punkte)

Obwohl Sie den Systemaufruf `poll()` wahrscheinlich nicht kennen, können Sie seinen Aufruf in Assembler sicher skizzieren. Dieser Aufruf hat die Nummer 168.

```
int poll(struct pollfd *fds, nfds_t nfds, int timeout);
```

In Ihrer Lösung nennen Sie die Parameter einfach `fds`, `nfds` und `timeout`. Wohin müssen diese Parameter übergeben werden?

Aufgabe 9 (4 Punkte)

Punkte: a) 2, b) 2, c) 2, d) 2

Im Buch von Bartlett wird im Kapitel 9 die Funktionsweise des Heap erklärt. Beantworten Sie dazu folgende Fragen:

a) Wie heissen die wesentlichen Funktionsaufrufe zur Verwendung des Heap?

b) Was verstehen Sie unter *Unmapped Memory*?

c) Wie kann der Heap-Speicherbereich vergrössert werden?

d) In welchen Datenstrukturen werden die vom Heap angeforderten Speicherblöcke verwaltet? Zeichnen Sie ein Diagramm zur Erläuterung.

Ende der Prüfung