

System-Programmierung (syspr)

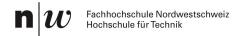
8. April 2019

thomas.amberg@fhnw.ch

Assessment I 4ibb2

Vorname:	Punkte:	/ 90,	Note:
Name:	Frei lassen für Korrektur.		
Klasse: □ Klasse 4ibb1 ⊠ Klasse 4ibb2			
Hilfsmittel:			
- Die vom Dozent ausgeteilte C Referenzkarte.			
- Lösen Sie die Aufgaben direkt auf den Prüfungsl	olättern.		
- Zusatzblätter, falls nötig, mit Ihrem Namen und	Fragen-Nr. auf jed	dem Blatt.	
Nicht erlaubt:			
- Unterlagen (Slides, Bücher,).			
- Computer (Laptop, Smartphone,).			
- Kommunikation mit anderen Personen.			
Bewertung:			
- Multiple Response: \square Ja oder \square $Nein$ ankreuze	n, +1/-1 Punkt pro	richtige/fa	alsche Antwort,
beide nicht ankreuzen ergibt +0 Punkte; Total p	ro Frage gibt es ni	e weniger a	ıls 0 Punkte.
- Multiple Choice: Eine \boxtimes <i>Antwort</i> pro Frage ank	reuzen, 4 Punkte p	oro richtige	Antwort.
- Offene Fragen: Bewertet wird Korrektheit, Vollst	tändigkeit und Kü	rze der Ant	wort.
- Programme: Bewertet wird die Skizze/Idee und	Umsetzung des Pr	ogramms.	
Fragen zur Prüfung:			
- Während der Prüfung werden vom Dozent keine	e Fragen zur Prüfu	ng beantwo	ortet.

- Ist etwas unklar, machen Sie eine Annahme und notieren Sie diese auf der Prüfung.



Erste Schritte in C

1) Für welche diese	er Typen ist sizeof(type) immer kleiner als sizeof(int)?	Punkte:	_/4
Zutreffendes ankre	euzen:		
□ Ja □ Nein	float		
□ Ja □ Nein	char		
□ Ja □ Nein	long		
□ Ja □ Nein	unsigned int		
2) Welche dieser A	usdrücke kompilieren fehlerfrei in C?	Punkte:	_/4
Zutreffendes ankre	euzen; $C = C99$:		
□ Ja □ Nein	int i = 3.141;		
□ Ja □ Nein	<pre>bool b = true;</pre>		
□ Ja □ Nein	string s = "hello";		
□ Ja □ Nein	int j = !1;		
3) Was sind zwei G	ründe System-Datentypen wie size_t zu nutzen, statt int?	Punkte:	_/4



Funktionen in C

4) Schreiben Sie ein Programm, das seine Argumente in Uppercase ausgibt: Punkte: / 12			
\$./my_program just a test JUST A TEST			
Hier ein Auszug aus der Doku, #includes und Fehlerbehandlung können Sie weglassen:			
<pre>int printf(const char *format,); // format string %s, int %d int toupper(int ch); // convert a character to uppercase, e.g. 'a' => 'A'</pre>			
Idee (kurz) und Source Code hier, oder auf Zusatzblatt mit Ihrem Namen und Fragen-Nr.:			

5) Gegeben den folgenden Code, welchen Wert hat k nach Aufruf von f()? Punkte: ____ / 4

```
int f(int *m, int n) {
   return *(m + 1) * n;
}

int main() {
   int i[] = {3, 5};
   int j = 2;
   int k = f(i, j);
}
```

Resultat und Begründung hier eintragen:

6) Gegeben den folgenden Code, welche Aufrufe von map() sind erlaubt? Punkte: ____ / 4

```
int sum(int *a, int n) { int s = 0; while (n > 0) { s += a[--n]; } return s; } int avg(int a[], int n) { return sum(a, n) / n; } void map(int a[], int n, int (*op)(int[], int)) { printf("%d", op(a, n)); }
```

Zutreffendes ankreuzen:

File In-/Output

7) Schreiben Sie ein Programm, das eine per Command-line angegebene, existierenden Datei x in mehrere Dateien (x.0, x.1, ...) einer gegebenen Maximalgrösse n splittet. Punkte: ____ / 16 Nutzen Sie dazu diese System-Calls, #includes und Fehlerbehandlung können Sie weglassen:

```
int open(const char *pathname, int flags);
int open(const char *pathname, int flags, mode_t mode); // Opens the file
specified by pathname. Or creates it if O_CREAT is used. Returns the file
descriptor. Flags include O_APPEND, O_CREAT, O_RDONLY, O_WRONLY, O_RDWR.
Modes, which are used together with O_CREAT include S_IRUSR and S_IWUSR.

ssize_t read(int fd, void *buf, size_t n); // Attempts to read up to n
bytes from file descriptor fd into buf. Returns number of bytes read ≤ n.

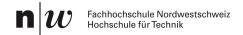
ssize_t write(int fd, const void *buf, size_t n); // Writes up to n bytes
from buf to the file referred to by fd. Returns nr. of bytes written ≤ n.

int close(int fd); // Closes the file descriptor fd.

void sprintf(char *str, const char *format, ...); // Prints to a string,
with the same format as printf(), e.g. %d for int, %s for string.

int atoi(const char *str); // Converts a string to an int.
```

Idee (kurz) und Source Code hier, und auf Zusatzblatt mit Ihrem Namen und Fragen-Nr.:



Prozesse und Signale

8) Was passiert mit einem Prozess, dessen Pa	rent bereits terminiert hat?	Punkte: / 4
Eine Antwort (von 4) ankreuzen:		
\square Er wird zu einem Zombie-Prozess.	☐ Er wird vom <i>init</i> -Prozess	adoptiert.
\square Er wird sein eigener Parent-Prozess.	☐ Die Parent Prozess-ID wi	rd 0.
9) Schreiben Sie ein Programm, welches das S	SIGINT Signal (CTRL-C) genau	einmal ignoriert,
und dann das Default-Verhalten wieder herst	ellt, d.h. es terminiert bei CTL-C	C. Punkte: / 8
<pre>typedef void (*sighandler_t)(int); sighandler_t signal(int signum, sighandler_t) SIG_DFL, or a programmer-defined fund</pre>		
int pause (void); // Pause causes the signal terminates the process or cau		
Idee (kurz) und Source Code hier, oder auf Z	usatzblatt mit Ihrem Namen un	d Fragen-Nr.:



10) Gegeben die Stack Datenstruktur (unten), implementieren Sie *stack_push()*, um pro *item* einen Node zu allozieren und in den Stack einzufügen, und die Funktion *stack_pop()*, um das zuletzt eingefügte *item* abzuholen und Speicher des Nodes freizugeben.

Punkte: ___ / 9

```
typedef struct node {
    struct node *next;
    int item;
} Node;

static Node *stack = NULL;

void stack_push(int item); // TODO
int stack_pop(); // TODO
```

Hier ein Auszug aus der Doku, #includes und Fehlerbehandlung können Sie weglassen:

```
void *malloc(size_t size); // Allocates size bytes and returns a pointer
to the allocated memory.

void free(void *ptr); // Frees the memory space pointed to by ptr, which
must have been returned by a previous call to malloc().
```

Idee (kurz) und Source Code hier, oder auf Zusatzblatt mit Ihrem Namen und Fragen-Nr.:

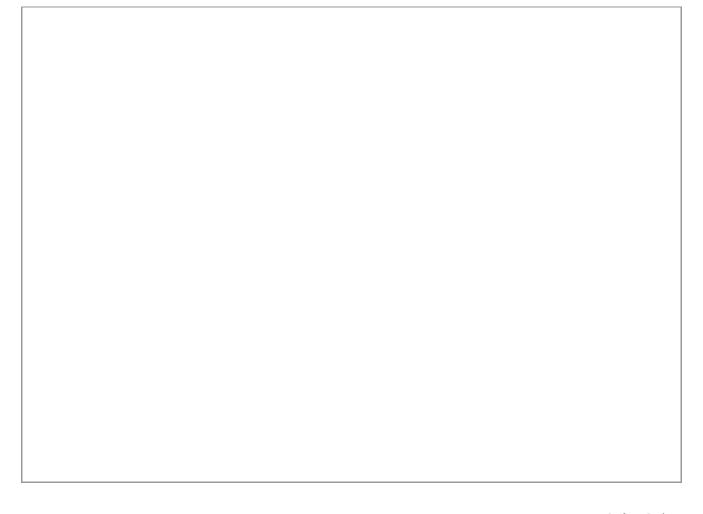
Prozess-Lebenszyklus

}

11) Was ist der Output dieses Programms, und wieso?

```
#include ...
int main(void) {
    for (int i = 0; i < 2; i++) {
        pid_t pid = fork();
        if (pid == 0) {
            printf("child pid = %d\n", getpid());
        } else {
            printf("parent pid = %d\n", getpid());
            wait(NULL);
        }
    }
    return 0;</pre>
```

Output und Begründung hier eintragen; Annahme: #includes sind da, Parent läuft zuerst:



Punkte: ____ / 9



Threads und Synchronisation

/



Zusatzblatt zu Aufgabe Nr	_ von (Name)