Herzlich Willkommen bei Inf-Einf-B Block 4.5.

Heute wiederholen und vertiefen wir Speicher.

Diese Konzepte fallen vielen anfangs schwer – Sie können auch die Tutorien nutzen, um Fragen zu klären.

Kurs-Website mit Wochenplan, Videos, Notizen, Aufgaben, FAQ, uvm.: **inf.zone**

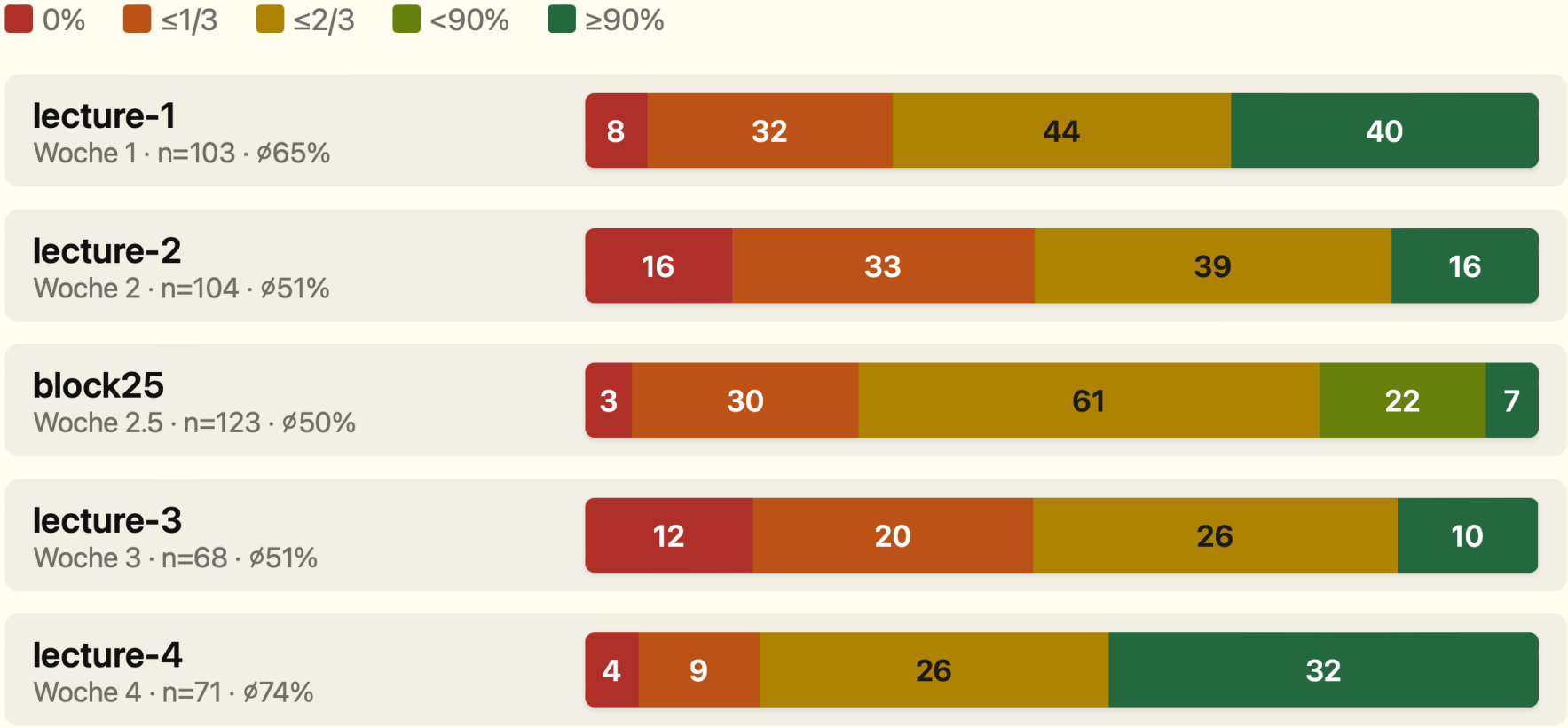
Die Vorlesungen werden aufgezeichnet und im Internet veröffentlicht. Die Videos werden allerdings meist erst 3 Wochen später online sein; Folien und Notizen am Tag nach der Vorlesung. Machen Sie sich Notizen!

Ihre Fragen sind in der Aufzeichnung in der Regel nicht zu hören (ich wiederhole sie).

Es gibt eine Pause.

Dies ist Inf-Einf-B.

Kurs-Statistiken



Hochgeladene Übungsaufgaben

Woche 0 Scratch	Woche 1 C	Woche 2 Arrays	Woche 3 Algor.
88	130	115	78

Woche 4: 12+

PROBLEM 1

Was wird ausgegeben?

```
int x = 3;
int y = 7;
int *p = &x;
int *q = &y;
*p = *q;
p = q;
*p = 10;

printf("x=%i, y=%i\n", x, y);
```

PROBLEM 1

Was wird ausgegeben?

```
int x = 3;
int y = 7;
int *p = &x;
int *q = &y;
*p = *q;
p = q;
*p = 10;

printf("x=%i, y=%i\n", x, y);
```

x=7, y=10

PROBLEM 2

Hier fehlt etwas!
Ergänzen Sie den
Code, sodass er
kompilierbar ist.

```
char *s = "CODE";

// Soll ausgeben: CODE
printf("_____", _____);

// Soll ausgeben: D (mit Array-Syntax)
printf("_____", _____);

// Soll ausgeben: E (mit Pointer-Arithmetik)
printf("_____", _____);
```

PROBLEM 2

Hier fehlt etwas!
Ergänzen Sie den
Code, sodass er
kompilierbar ist.

```
char *s = "CODE";

// Soll ausgeben: CODE
printf("%s\n", s);

// Soll ausgeben: D (mit Array-Syntax)
printf("%c\n", s[2]);

// Soll ausgeben: E (mit Pointer-Arithmetik)
printf("%c\n", *(s+3));
```

PROBLEM 3

Verbessern Sie den fehlerhaften Code! p und x sollen verwendet werden.

```
// Ziel: x soll am Ende den Wert 42 haben

int x = 41;

int *p;

*p = 42;

printf("x = %i\n", x);
```

PROBLEM 3

Verbessern Sie den fehlerhaften Code! p und x sollen verwendet werden.

```
// Ziel: x soll am Ende den Wert 42 haben

int x = 41;

int *p = &x;

*p = 42;

printf("x = %i\n", x);
```

```
int n = 50;

int *p = &n;
```

```
string s = "HI!";
```

```
typedef struct
{
    string name;
    string number;
} person;
```

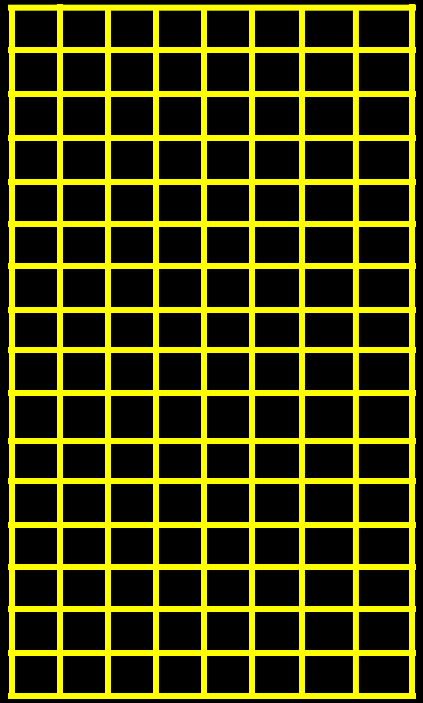
```
typedef int integer;
```

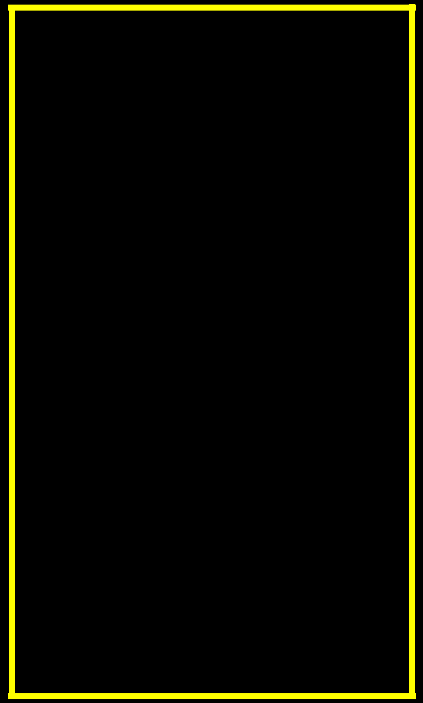
```
typedef uint8_t BYTE;
```

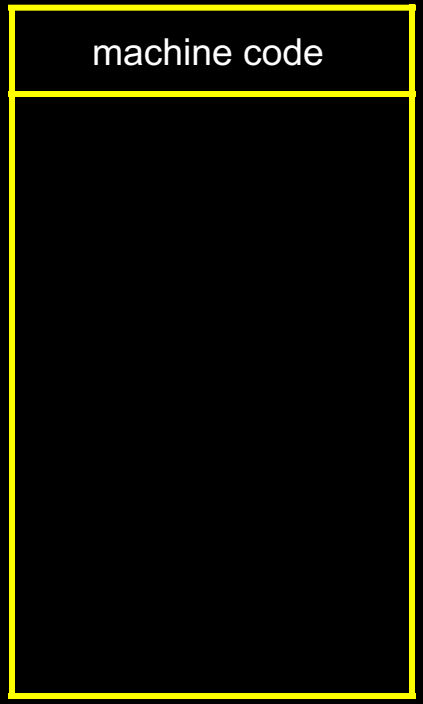
```
typedef char *string;
```

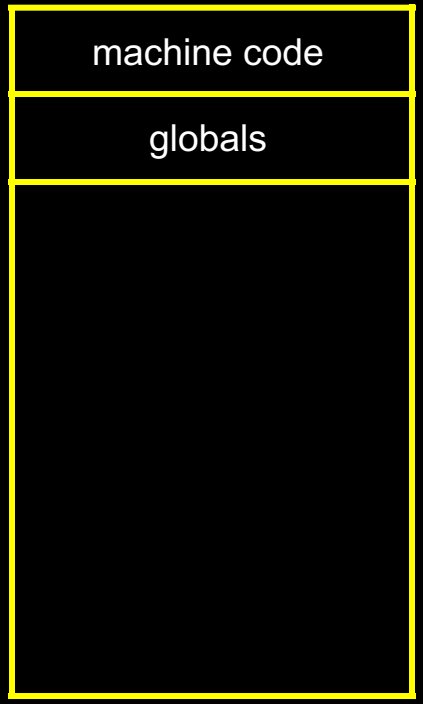
NULL

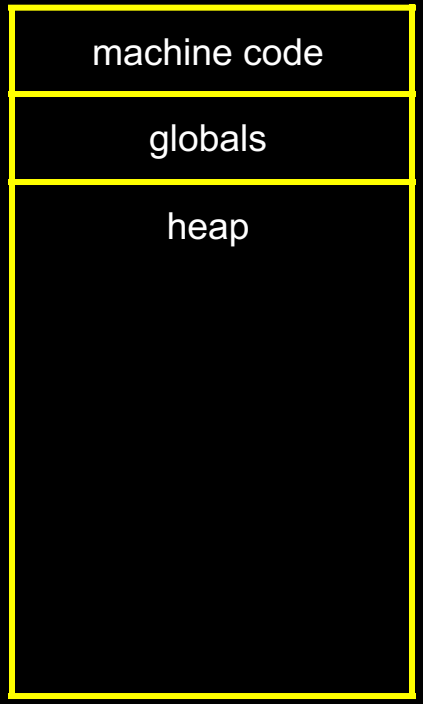
Garbage Values

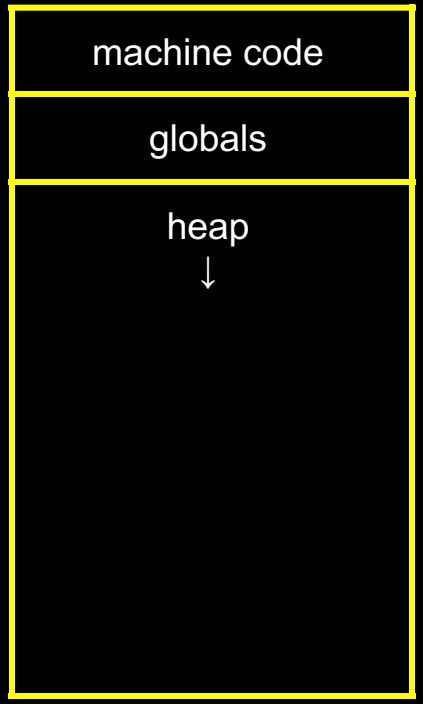


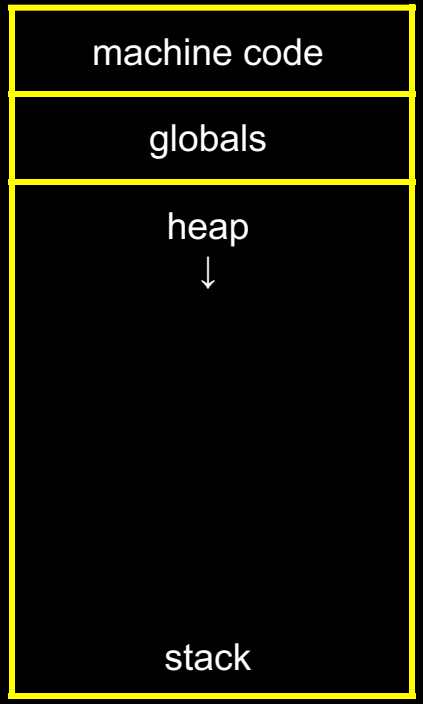


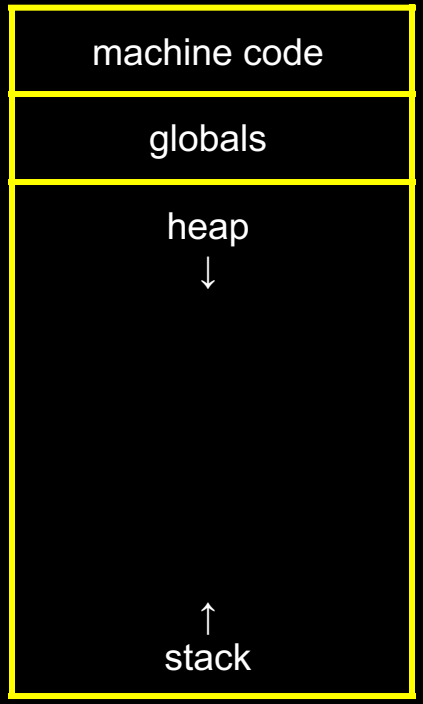

machine code

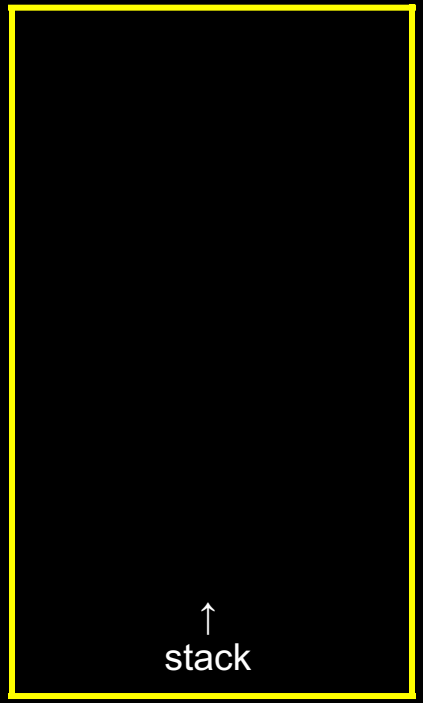

machine code
globals

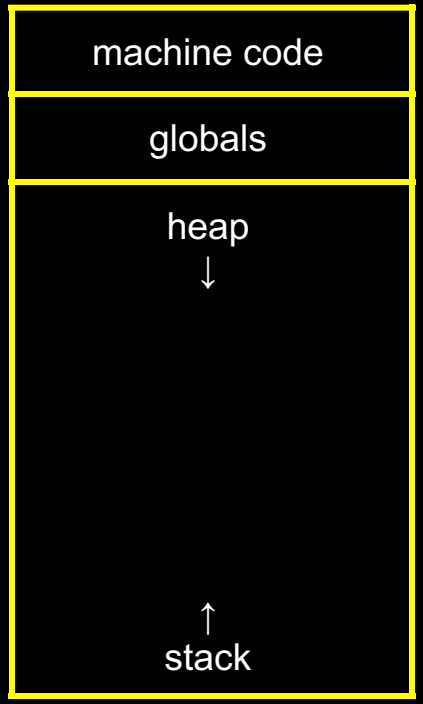

machine code
globals
heap

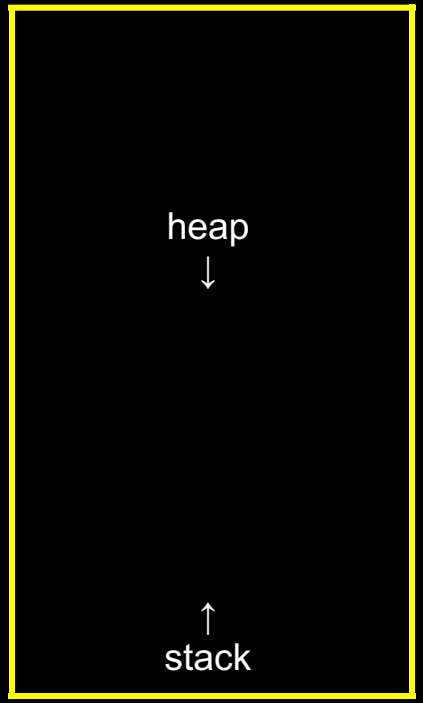

machine code
globals
heap
↓


machine code
globals
heap
↓
stack


machine code
globals
heap
↓
↑
stack


↑
stack


machine code
globals
heap
↓
↑
stack


heap
↓
↑
stack

heap overflow

stack overflow

buffer overflow

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

scanf

...

File-I/O

Input und Output

fopen

fclose

fprintf

fscanf

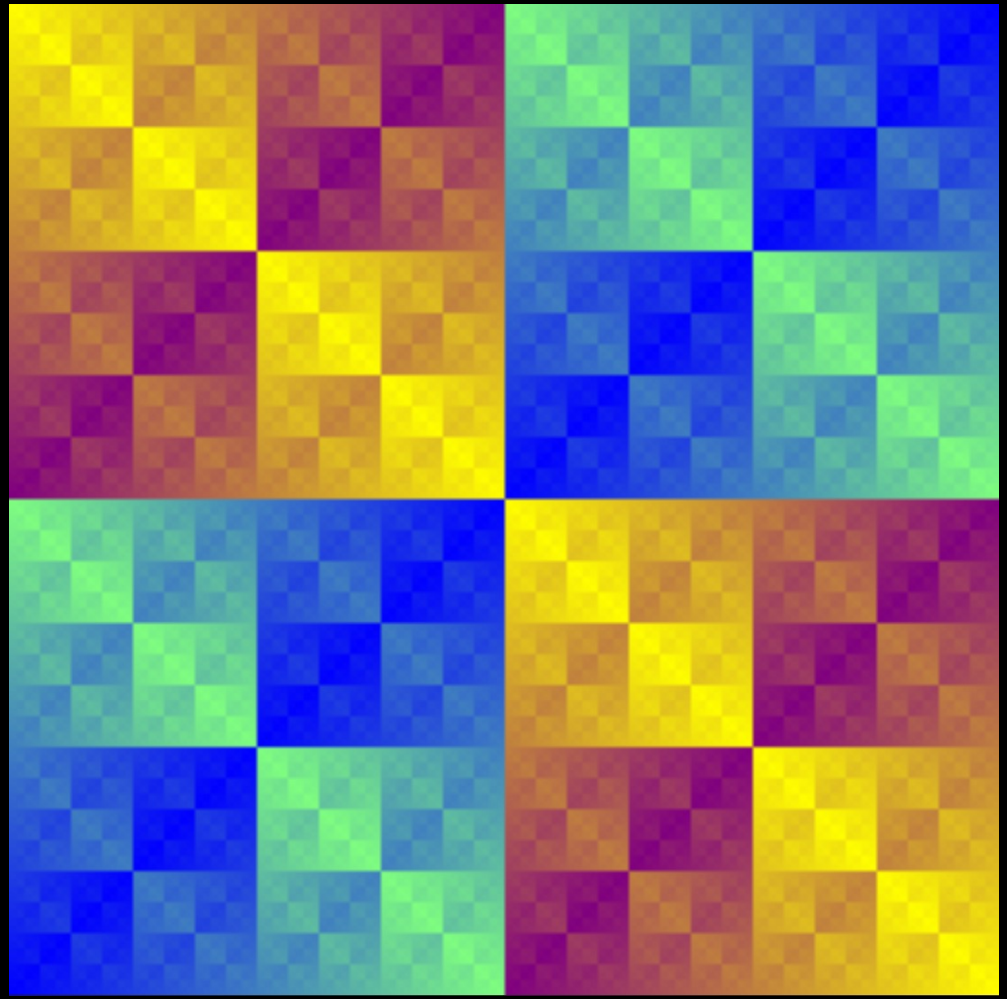
fread

fwrite

fseek

…

BMP



Dies war Inf-Einf-B.