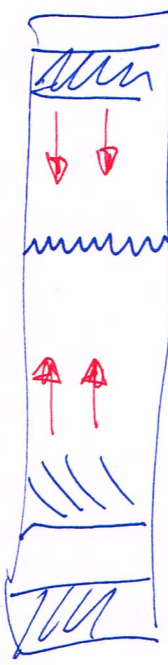
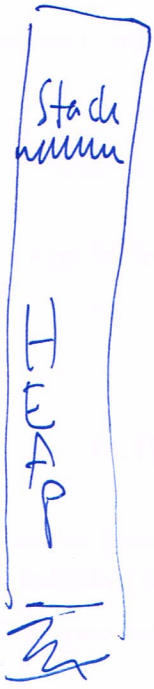
GdBS
20.03.2021
1/3

6.

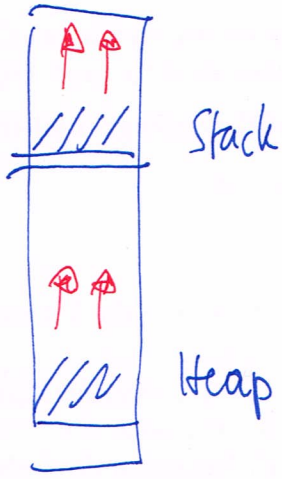
noch



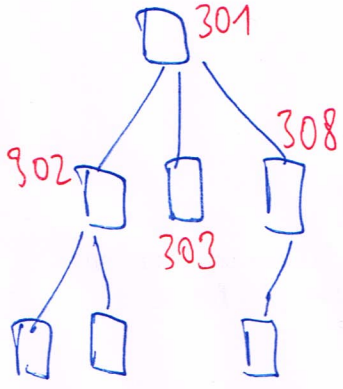


0

Alternative:



7.



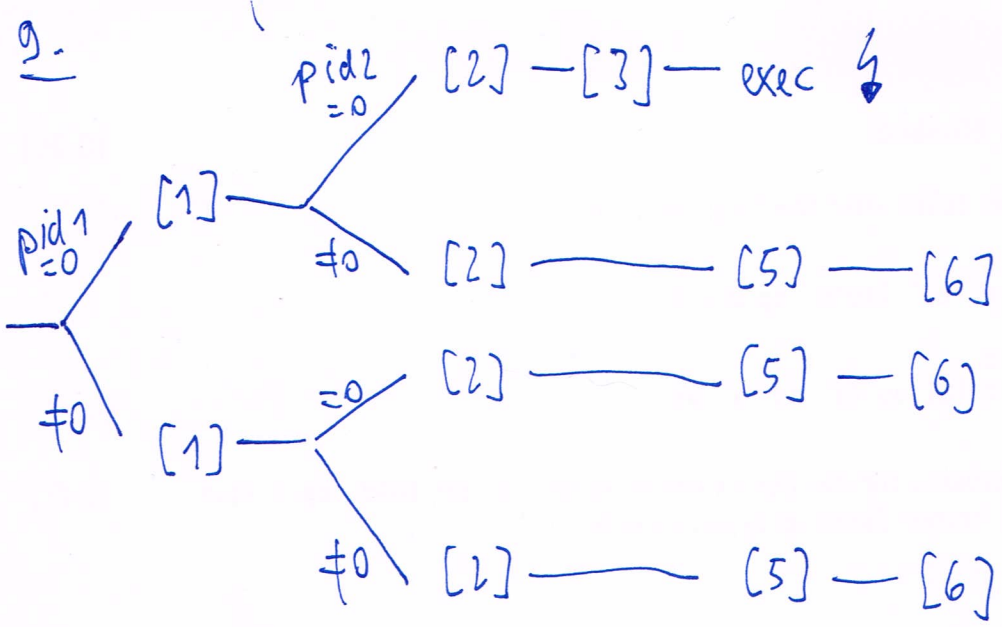
getchpid()? nein!

getppid()

```
pid1= fork(); //302
pid2= fork(); //303
pid3= fork(); //308
```

```
// wer ist der andere Prozess?
int pid= fork();
int andere_pid;
if (pid==0) { andere_pid= getppid(); } //im Sohn
else        { andere_pid= pid; } //im Vater
```

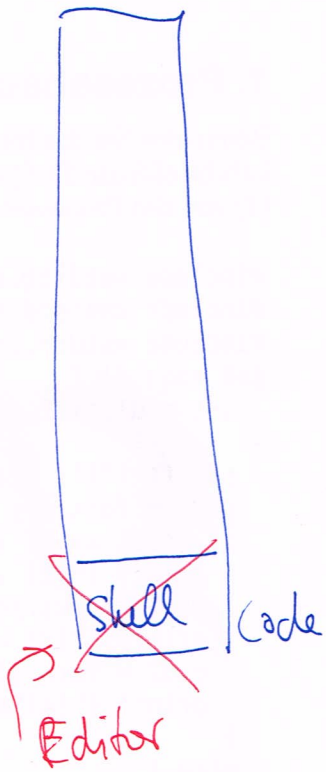
9.



8.

Fork + exec

```
pid= fork();    // kindprozess
if (pid==0) {
    exec("editor");
} else {
    printf("Kate editor gestartet")
}
```



Windows / spawn

spawn ("editor")

echo Hallo > ausgabe.txt

Prof. Dr. Hans-Georg Eßer
FH Südwestfalen
Übungsblatt 1
20.03.2021

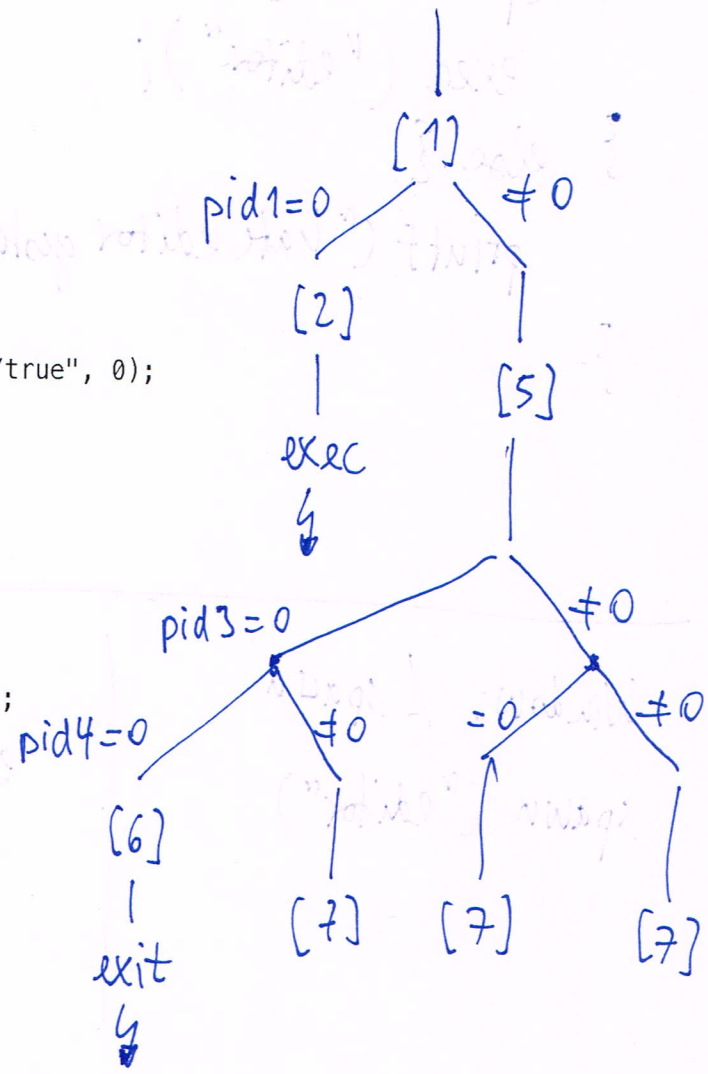
20.03.
3/3

1. Prozessbaum

Betrachten Sie das folgende Programm `forktest.c` (Listing 1) und erstellen Sie (wie in Aufgabe 9 auf Lehrbrief-Seite 53 f.) einen Prozessbaum, aus dem Sie ablesen können, wie oft die Ausgaben [1] bis [7] von den Prozessen erzeugt werden.

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
int main () {
  int pid1,pid2,pid3,pid4;

  printf ("[1] Start\n");
  pid1 = fork ();
  if (pid1 == 0) {
    printf ("[2] vor dem exec\n");
    execl ("/usr/bin/true", "/usr/bin/true", 0);
    printf ("[3] nach dem exec\n");
    pid2 = fork ();
    printf ("[4]\n");
  }
  else {
    printf ("[5] zweiter Zweig\n");
    pid3 = fork ();
    pid4 = fork ();
    if (pid3+pid4 == 0) {
      printf ("[6] pid-Summe ist 0\n");
      exit (0);
    }
    printf ("[7] Ende\n");
  }
}
```



(*) (pid3==0) && (pid4==0)