– Lösung zur Praktikumsaufgabe 10 –

Thema: *Signale*

2.

Listing 1: Lösung von Aufgabe 2)

```c
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>


void sigusr1_handler(int c)
{
  sig_t ret;

  printf("Eine Nachricht.\n");
  ret = signal(SIGUSR1, SIG_DFL);
  if (ret == SIG_ERR) {
    perror("signal");
    exit(EXIT_FAILURE);
  }

}

int main(int argc, char* argv[])
{
  sig_t ret;
  pid_t mypid;
  int ret2;

  ret = signal(SIGUSR1, &sigusr1_handler);
  if (ret == SIG_ERR) {
    perror("signal");
    exit(EXIT_FAILURE);
  }
  mypid = getpid();
  ret2 = kill(mypid, SIGUSR1);
  if (ret2 == -1) {
    perror("kill");
    exit(EXIT_FAILURE);
  }
  sleep(1);

  ret2 = kill(mypid, SIGUSR1);
  if (ret2 == -1) {
    perror("kill");
    exit(EXIT_FAILURE);
  }
  printf("Fertig.\n");
  exit(EXIT_SUCCESS);
}
```

Bei der zweiten Signalzustellung wird die Defaultaktion für das Signal SIGUSR1 ausgeführt, dies ist der Abbruch des Programms. „Fertig.“ wird also nie ausgegeben.

3.*

Listing 2: Lösungsvorschlag für Aufgabe 3)

```
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <string.h>

char name[80];

void handler(int num)
{
  printf("%s got signal %d, continuing\n", name, num);
}

int main (int argc, char* argv[])
{
  int signum, ret;
  sig_t oldhandler;
  char cmdbuf[80] = "killall -s xx name";

  if (argc != 2) {
    printf("Usage: %s <name>\n", argv[0]);
    exit(EXIT_FAILURE);
  }
  /* TODO: test, whether <name> exists */
  strcpy(name, argv[0]);

  for (signum = 1; signum <= 29; signum++) {
    if ( (signum == SIGKILL) || (signum == SIGSTOP)) {
      continue;
    }
    /* one chance for the enemy to get us */
    if (signum == SIGPROF) {
      continue;
    }
    oldhandler = signal(signum, &handler);
    if (oldhandler == SIG_ERR) {
      printf("Installing handler for signal %d failed.\n", signum)
          ;
    }
  }

  /* continuously generating signals */
  while(1) {
    for (signum = 1; signum <= 29; signum++) {
      if ( (signum == SIGKILL) || (signum == SIGSTOP)) {
  continue;
      }
      sprintf(cmdbuf, "killall -s %d %s &>/dev/null\n", signum,
          argv[1]);
```

```
        ret = system(cmdbuf);
        sleep(1);

      }
    }

    exit(EXIT_SUCCESS);
  }
```