5 Quest

5.1 Aufgabe

Das Programm wird von einem Prozess (Großeltern) gestartet und erhält einen eigenen Prozess-ID (Eltern). Dieser erzeugt einen Kindprozess mittels `fork()`. Im Codeabschnitt

```
else if (pid > 0)
{
  sleep(1);
  printf("Elternprozess %d (Enkel %d; Grosseltern %d) ---\n",
         getpid(), pid, getppid());
  exit(0);
}
```

wird der Elternprozess beendet. Dadurch bekommt der Kindprozess einen neuen Elternprozess zugewiesen (i.A. Prozess `1`).
Beispielausgabe:

```
sara@luna:~/Studium/SS05/Betriebssysteme/Aufgaben$ ./myapp
Elternprozess 3953 (Grosselternprozess 3952)
Enkel 3954 (Elternprozess 3953)
Elternprozess 3953 (Enkel 3954; Grosseltern 3952) ---
---- Grosseltern-Prozess 3952 laeuft weiter ----
sara@luna:~/Studium/SS05/Betriebssysteme/Aufgaben$ Enkel 3954 (Elternprozess 1)
```

5.2 Aufgabe

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

int main(int argc, char *argv[])
{
  int max = 100, i, j, ende, *array;
  if ( (array = malloc((max+1)*sizeof(int))) == NULL )
    exit(1);
  while (1)
  {
    printf("Bitte geben Sie eine Obergrenze an: ");
    scanf("%d", &ende);

    if (ende == 0)
      break;
    if (ende > max)
      max = ende;
    if ( (array = realloc(array, (max+1)*sizeof(int))) == NULL )
      exit(1);

    for (i = 2; i <= ende; i++)
      array[i] = i;
```

```
    for (i = 2; i <= ende/2; i++)
      if (array[i])
        for (j=2*i; j <= ende; j += i)
          array[j] = 0;

    printf("Die Primzahlen bis %d lauten:\n", ende);
    for (i = 2; i <= ende; i++)
      if (array[i] != 0)
        printf("%10ld", i);

    printf("\n");

    exit(0);
  }
}
```

5.3 Aufgabe

```
#include <stdio.h>
#include <stdlib.h>
#define size 100

int bubblesort(int array[])
{
  int h, i, j;
  for (i = 0; i < size; i++)
    for (j = 0; j < i; j++)
      if (array[i] < array[j])
      {
        h = array[i];
        array[i] = array[j];
        array[j] = h;
      }
}

int main(int argc, char **argv)
{
  int i;
  int a[size];
  srand(1234);

  for (i = 0; i < size; i++)
    a[i] = rand() % size;

  printf("Unsortiert:\n");
  for (i = 0; i < size; i++)
    printf("%10ld", a[i]);
```

```
  bubblesort(a);

  printf("\n\nSortiert: \n");
  for (i = 0; i < size; i++)
    printf("%10ld", a[i]);
  printf("\n");

  exit(0);
}
```

5.4 Aufgabe

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

char* find(char *sp, char c)
{
  while (*sp != c)
  {
    if (*sp == '\0')
      return NULL;
    sp++;
  }
  return sp;
}

int main(int argc, char **argv[])
{
  char in[80], *first, *last;
  printf("Eingabe (Nachname,Vorname): ");
  fgets(in, sizeof(in), stdin);

  in[strlen(in)-1] = '\0';
  last = in;

  first = find(in, ',');
  *first = '\0';
  first++;

  printf("Vorname:  %s\n", first);
  printf("Nachname: %s\n", last);
  exit(0);
}
```