```c
/*
   Esempio minimale di fork. Un certo numero di fork, e sia il figlio
   che il padre non fanno altro che scrivere i due ID. Il padre non
   aspetta i figli e quindi, se uno dei figli "ritarda" diventa uno
   'zombie' e viene 'adottato' dal processo 'init', per cui il parent
   ID che stampa non è quello del padre originale. Le due opzioni -c e
   -p permettono di scegliere di quanto "ritardare" l'esecuzione della
   printf nei figli e nel padre; il default è zero per tutti.
*/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>             /* fork */
#include "fun.h"

int main(int argc, char *argv[])
{
  pid_t pid;
  int sleep_child = 0;
  int sleep_parent = 0;
  int opt;
  int count = 3;
  int i;

  get_opt(argc, argv,
          &sleep_child, &sleep_parent,
          NULL);

  for (i=0; i<count; i++) {
    pid = fork();
    if (pid == 0) {

      sleep(sleep_child);
      printf("child:  %d with parent: %d\n",
             getpid(), getppid());
      /* try removing this return */
      return;
    } else {

      sleep(sleep_parent);
      printf("parent: %d with child:  %d\n",
             getpid(), pid);
    }
  }
  return 0;
}
```