fork

4) Scrivere una programma C che implementi la seguente situazione:
Un processo P crea due figli ed aspetta la terminazione di entrambi.
Il primo figlio F1 stamperà sullo schermo un messaggio indicando il proprio pid ed i numeri da 1 a 5000. Dopo di ciò terminerà.
Il secondo figlio stamperà sullo schermo 5000 volte il pid del padre e poi eseguirà il comando /bin/ls.
Dopo la terminazione dei due figli P stamperà sullo schermo un messaggio indicando il pid dei due figli ed il fatto che sono terminati e poi a sua volta terminerà.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
int main(){
  pid_t F1, F2;
  F1 = fork();
  // effettuare i controlli sulla riuscita della fork

  // FIGLIO 1
  if(F1 == 0){
    printf("[FIGLIO 1]: il mio pid e': %d\n", getpid());
    for(int i = 1; i <= 5000; i++){
      printf("%d\n", i);
    }
    exit(EXIT_SUCCESS);
  }

  // PADRE
  else{
    F2 = fork();
    // effettuare i controlli sulla riuscita della fork

    //FIGLIO 2
    if(F2 == 0){
      for(int i = 1; i <= 5000; i++){
        printf("[FIGLIO2]: il pid di mio padre e' %d\n", getppid());
      }
      execlp("/bin/ls", "ls", NULL);
    }

    //PADRE
    waitpid(F1, NULL, NULL);
    printf("[PADRE]: F1 e' terminato (pid: %d)\n", F1);
    waitpid(F2, NULL, NULL);
    printf("[PADRE]: F2 e' terminato (pid: %d)\n", F2);
  }
  return 0;
}
```