In this exercise you will write a program that provides two library functions for creating a tree of processes and for managing signals in the created tree (to a certain extent).

You are to provide the implementation for the following two functions.

- void `createProcess (void (*F) (void))`: that receives a pointer to a function F and using the `fork()` system call creates a new child process which is then viewed as a single node in the process tree. The created process executes F. The child process should not return from this function.

- `int sendSignal(…)`: this function is used in order to send a signal from the process invoking the call to all processes found the sub-tree of processes rooted at the caller process (recall that the entire sub-tree was created using `createProcess()`). The process invoking the function sends a SIGINT signal to all its offsprings where the transition of the signal within the process tree is to be done as follows:

  Each process that receives a signal sends the same signal to all its children and waits for an 'OK' signal to be returned from all of its children. Once this is achieved it sends an 'OK' signal back to its parent process. This signal handling ends when the initial process that invoked `sendSignal` receives an 'OK' signal back from all its children.

  For the 'OK' signal you are to use:

  ```
  #define SIGOK 10
  ```

You can assume that all the processes created never terminate.