```c
#include <stdio.h>

int main( void ) {

int ID = -9999;

  printf( "ANTES: %d\n", ID );
  ID = fork();
  printf( "DEPOIS: %d\n", ID );
}
```

```c
#include <stdio.h>

int main( void ) {

  execl(  "/bin/ls","ls",0 );

}
```

```c
// PSEUDO-CÓDIGO

while( 1 ){
  leia_comandos( comando, parametros );
  if( fork() != 0 ){
     // PAI
     waitpid( -1, &status, 0 );
  } else {
    // FILHO
    execve( comando, parametros, 0 );
  }
}
```

```c
#include <stdio.h>
int main(void){
  int  pfd[2];   // canais de pipe
  char l[ 20 ];  // buffer

  printf( "ANTES: %d:%d\n", pfd[0], pfd[1] );
  pipe( pfd );
  printf( "DEPOIS: %d:%d\n", pfd[0], pfd[1] );

  write( pfd[ 1 ], "ALO MUNDO", 10 );
  read( pfd[ 0 ], l, 6 );
  printf( "RESULTADO: %s\n", l );
  return 0;
}
```

```c
#include <stdio.h>
int main(void){
  int  pid, pfd[2];
  char l[ 20 ];

  pipe( pfd );
  pid = fork();
  printf( "%d:%d\n", pfd[ 0 ], pfd[ 1 ] );

  switch( pid ){
    case -1: exit(1);
               break;

    case 0:  close( pfd[ 0 ] );
             write( pfd[ 1 ], "ALO MUNDO", 10 );
              break;

    default: close( pfd[ 1 ] );
             read( pfd[ 0 ], l, 6 );
             printf( "%s", l );
            break;
  }
  return 0;
}
```

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

int main( void )
{
  char  lc[ 81 ];
  char *argv[ 20 ];
  int   pid, i, status;

  while( 1 ) {

    printf( "Prompt > " );
    gets( lc );

    if( ! strcmp( lc, "" ) )
      continue;

    argv[ 0 ] = strtok( lc, " " );
    if( ! strcmp( argv[ 0 ], "exit" ) )
      exit( 0 );

    i = 1;
    while( i < 20 && (argv[ i ] = strtok( NULL, " " )) )
      ++i;

    if( (pid = fork()) == -1 ) {
      printf( "Erro no fork\n" );
      exit( 1 );
    }

    if( pid == 0 )
      if( execvp( argv[ 0 ], argv ) ) {
        printf( "Erro no execv\n" );
        exit( 1 );
      }

    wait( &status );

  }
}
```
