

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <string.h>
#include <sys/signal.h>

void sigusr1_handler(int sig_nr){
    int sleepVal;
    printf("\n got signal SIGUSR1 %d ...", sig_nr);
    fflush(stdout);
    sleepVal = sleep(1);
    printf(" ... SIGUSR1 done");
    fflush(stdout);
    return;
}

void sigusr2_handler(int sig_nr){
    int sleepVal;
    printf("\n got SIGUSR2 %d ...", sig_nr);
    fflush(stdout);
    sleepVal = sleep(1);
    printf(" ... SIGUSR2 done");
    fflush(stdout);
    return;
}

void installSignalHandler(){
    struct sigaction sa_usr1;
    struct sigaction sa_usr2;
    sigset_t child_sigset;

    sigemptyset(&child_sigset);

    sa_usr1.sa_handler = sigusr1_handler;
    sa_usr1.sa_mask = child_sigset;
    sa_usr1.sa_flags = 0;

    if (sigaction(SIGUSR1, &sa_usr1, NULL) != 0){
        printf("\n could not install handler for SIGUSR1 ");
    }else{
        printf("\n handler for SIGUSR1 installed");
        fflush(stdout);
    }

    sa_usr2.sa_handler = sigusr2_handler;
    sa_usr2.sa_mask = child_sigset;
    sa_usr2.sa_flags = 0;

    if (sigaction(SIGUSR2, &sa_usr2, NULL) != 0){
        printf("\n could not install handler for SIGUSR2 ");
    }else{
        printf("\n handler for SIGUSR2 installed");
        fflush(stdout);
    }
    return;
}
```

```
void childFunction(){
    int i = 0;
    int sleepVal;

    installSignalHandler();

    while (i < 12){
        sleepVal = sleep(1);
        i++;
    }
    printf( "\n exit child process \n" );
    exit(0);
}

void parentFunction(pid_t pid){
    int i = 0;
    int sleepVal;
    pid_t waitVal = 0;

    sleepVal = sleep(1);
    printf( "\n parent process: start signalling " );
    fflush(stdout);

    kill(pid, SIGUSR2);
    kill(pid, SIGUSR2);
    kill(pid, SIGUSR1);
    kill(pid, SIGUSR2);
    kill(pid, SIGUSR1);
    printf( "\n parent process: 5 signals sent " );
    fflush(stdout);
    sleepVal = sleep(4);

    printf( "\n parent process: entering loop " );
    fflush(stdout);

    while (waitVal == 0){
        kill(pid, SIGUSR1);
        sleepVal = sleep(2);
        kill(pid, SIGUSR2);
        waitVal = waitpid(pid, NULL, WNOHANG);
        sleepVal = sleep(2);
    }
    return;
}

int main(void){
    pid_t pid;

    printf( "\n SIGUSR1: %d, SIGUSR2: %d ", SIGUSR1, SIGUSR2 );
    pid = fork();
    if (pid == 0){
        childFunction();
    }
    if (pid > 0){
        parentFunction(pid);
    }
    if (pid < 0){
        printf( "\n P: could not create process " );
    }
}
```

```
    }

    return;
}

/*
handler for SIGUSR1 installed
handler for SIGUSR2 installed
got SIGUSR2 ... .. SIGUSR2 done
got signal SIGUSR1 ...
got SIGUSR2 ... .. SIGUSR2 done ... SIGUSR1 done
got signal SIGUSR1 ... .. SIGUSR1 done
got SIGUSR2 ... .. SIGUSR2 done
got signal SIGUSR1 ... .. SIGUSR1 done
got SIGUSR2 ... .. SIGUSR2 done
got signal SIGUSR1 ... .. SIGUSR1 done
got SIGUSR2 ... .. SIGUSR2 done
got signal SIGUSR1 ... .. SIGUSR1 done
got SIGUSR2 ... .. SIGUSR2 done
got signal SIGUSR1 ... .. SIGUSR1 done
exit child process
*/
```