

Vježba 1: Signali

Zadatak

Računati proste brojeve te preko signala periodički ispisivati status. Koristiti *setitimer* sučelje za periodički alarm (vidjeti Primjer periodičke obrade pri dnu stranice). Na signal SIGINT privremeno zaustaviti rad na idućim brojevima (programski ostvariti zaustavljanje), odnosno nastaviti s radom ako je program prethodno bio zaustavljen. Na signal SIGTERM ispisati zadnji broj koji se provjerava (ili će biti idući) i završiti s radom.

Skica rješenja:

```
pauza = 0;
broj = 1000000001;
zadnji = 1000000001;

periodicki_ispis () {
    ispisi ( zadnji );
}

postavi_pauzu () {
    pauza = 1 - pauza;
}

prekini () {
    ispisi ( zadnji );
    izadji_iz_programa ();
}

glavna_funkcija () {
    povezi_signale_s_funkcijama; // na signal SIGTERM pozovi funkciju prekini()
    postavi_periodicki_alarm;    // svakih 5 sekundi pozovi funkciju periodicki_ispis();

    ponavljaj {
        ako je ( prost ( broj ) == DA )
            zadnji = broj;
        broj++;
        dok je ( pauza == 1 )
            pauziraj ();
    }
}

/* s kill -SIGINT/SIGTERM/SIGSTOP/SIGCONT pid ispitati rad */
```

Provjera je li broj prost može se obaviti jednostavnim kodom, npr. prema:

```
int prost ( unsigned long n ) {
    unsigned long i, max;

    if ( ( n & 1 ) == 0 ) /* je li paran? */
        return 0;

    max = sqrt ( n );
    for ( i = 3; i <= max; i += 2 )
        if ( ( n % i ) == 0 )
            return 0;

    return 1; /* broj je prost! */
}
```

Primjer kako bi trebao izgledati ispis programa (ako se izvorna datoteka zove lab1.c):

```

$ gcc lab1.c -lm -o lab1
$ ./lab1
zadnji prosti broj = 1000139111
zadnji prosti broj = 1000279801
zadnji prosti broj = 1000420261
^Czadnji prosti broj = 1000478719      (stisnut Ctrl+C)
zadnji prosti broj = 1000478719
zadnji prosti broj = 1000478719      (stisnut [Ctrl]+[\\] tj. [Ctrl]+[Ž])
^\\Quit (core dumped)
$

```

(Tekst u zagradama ne ispisuje program, već se samo opisuju korisnikove radnje.)

Primjer periodičke obrade

Mnoge operacije na UNIX sustavima se oslanjaju na signale. Primjerice, jedan od načina periodičke obrade može se ostvariti korištenjem signala, ne izravno već korištenjem sučelja koje periodički šalje signal procesu, a na koji se može pozvati potrebna funkcija.

Idući primjer, [itimer_primjer.c](#), prikazuje takav program.

```

#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <sys/time.h>

void periodicki_posao ( int sig )
{
    printf ( "Radim periodicki posao\n" );
}

int main ()
{
    struct itimerval t;

    /* povezivanje obrade signala SIGALRM sa funkcijom "periodicki_posao" */
    sigset ( SIGALRM, periodicki_posao );

    /* definiranje periodičkog slanja signala */
    /* prvi puta nakon: */
    t.it_value.tv_sec = 0;
    t.it_value.tv_usec = 100000;
    /* nakon prvog puta, periodički s periodom: */
    t.it_interval.tv_sec = 0;
    t.it_interval.tv_usec = 100000;

    /* pokretanje sata s pridruženim slanjem signala prema "t" */
    setitimer ( ITIMER_REAL, &t, NULL );

    while (1)
        pause (); /* pauzira do primitka bilo kojeg signala */

    return 0;
}

/* prevodjenje i pokretanje:
* $ gcc itimer_primjer.c -o itimer_primjer
* $ ./itimer_primjer
* Radim periodicki posao
* Radim periodicki posao
* Radim periodicki posao
* ...
* ^C
* (Ctrl+C prekida izvodjenje)
*

```

```
* isprobati pokrenuti te s naredbom kill mu poslati signale:
* - SIGINT, SIGTERM, SIGKILL, SIGSTOP, SIGCONT
*
* npr. nakon pokretanja, u drugoj konzoli napraviti:
* $ ps -a
* PID TTY          TIME CMD
* 3232 pts/0      00:00:00 itimer_primjer
* 3233 pts/6      00:00:00 ps
* $ kill -SIGTERM 3232
* u prvoj konzoli će program biti prekinut s porukom "Terminated"
*/
```

Napomena:

Ako se koristi Windows Subsystem for Linux (WSL) za izradu laboratorijske vježbe te se ništa ne događa kada se pošalje signal SIGQUIT (ctrl + \, odnosno ctrl + ž), problem se može zaobići tako da se signali šalju iz druge ljuške.

Primjerice, na početku programa ispisati PID procesa u kojem se program izvodi koji računa proste brojeve (npr. neka je PID tog procesa 12345) te pokrenete još jednu ljušku iz koje pomoću naredbe "kill -SIGQUIT 12345" možete poslati taj signal te tako zaustaviti izvođenje programa. (PID procesa se može doznati iz druge ljuške i naredbom ps koja se poziva iz komandne linije ako je proces već pokrenut.)