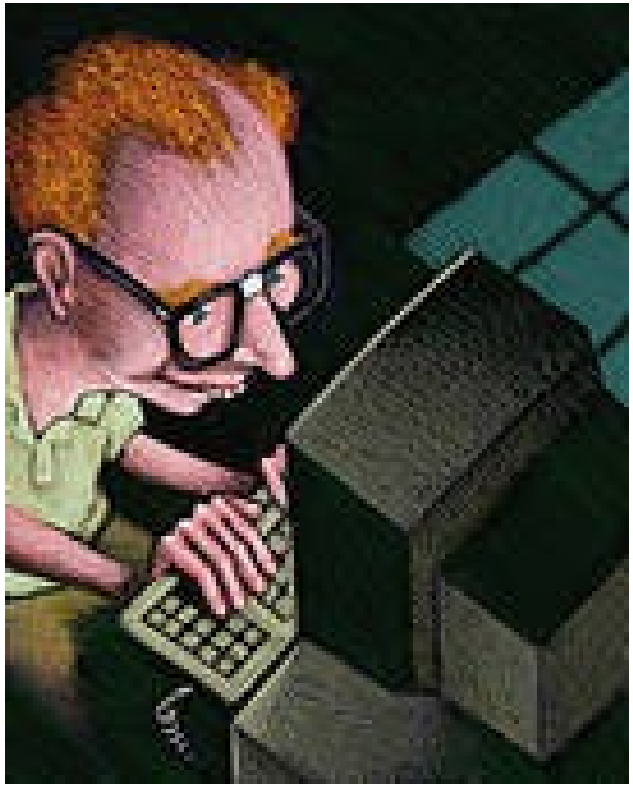
Knjižnjice
krmiljenje zaslona
Uvod v sistemsko programiranje



Standardna knjižnjica C

Funkcije z nizi

int strlen(s) Vrne število znakov v nizu s (brez nultega znaka).
char *strchr(s, c) Vrne kazalec na prvi nastop znaka c v nizu s. (sicer vrne NULL)
char *strrchr(s, c) Vrne kazalec na zadnji nastop znaka c v nizu s.
int strcpy (s2, s1) Kopira niz s1 v niz s2.
int strncpy(s2, s1, n) Kopira niz s1 v niz s2, vendar največ n znakov.
char *strcmp (s2, s1) Primerja niza in vrne:

- pozitivno vrednost če je s2>s1,
- 0..če je s2=s1
- negativno vrednost sicer

char *strncmp (s2, s1) Podobno kot strcmp, vendar primerja največ n znakov char *
strstr (s2, s1) V nizu s2 išče podniz s1 in vrne kazalec nanj

Dem

Konverzija podatkov

Pri pregledu naslednjih funkcij za konverzijo podatkov imejmo definirano:

```
#include < stdio.h >
int base;
char *s, *end;
```

int atoi (s) Pretvorba števila v nizu s v format int.
long atol (s) Pretvorba števila v nizu s v format long int
double atof (s) Pretvorba števila v nizu s v format double
double strtod (s,end) Pretvorba števila v nizu s, pisanem v znanstveni notaciji. Funkcija nastavi kazalec end na znak, ki je zaključil pretvorbo

long strtol (s, end, base)
unsigned long strtoul (s, end, base)

Konverzija niza s v long oziroma unsigned long. Pri tem je base uporabljena osnova (mora biti med 2 in 36). Vodeče ničle so ignorirane.

Dem

Funkcije z znaki

*So deklarirane v datoteki **ctype.h**. Zato imejmo:*

```
#include < ctype.h >
int c;
```

Na voljo imamo naslednje funkcije:

```
int isalpha (c)
int isupper (c)
int islower(c)
int isalnum (c)
int isdigit (c)
int isprint (c)
int iscntrl (c)
int ispunct (c)
int isspace (c)
int toupper (c)
int tolower (c)
int toascii (c)
```

Primer: konverzija črk niza v velike črke

```
i = 0;
while (s[i] != 0) {
    if ( isalpha( s[i] ))  s[i]= toupper(s[i++ ]);
}

/* ali pa z uporabo kazalca... */
while (*s != 0) if ( isalpha(*s)) {
    *s = toupper(*s); s++;
}
```

DEMO

Matematične funkcije

So deklarirane v datoteki **math.h**. Zato imejmo:

```
#include < math.h >
double x, y , *pd ;
long k;
int *pi , i;
```

Na voljo imamo:

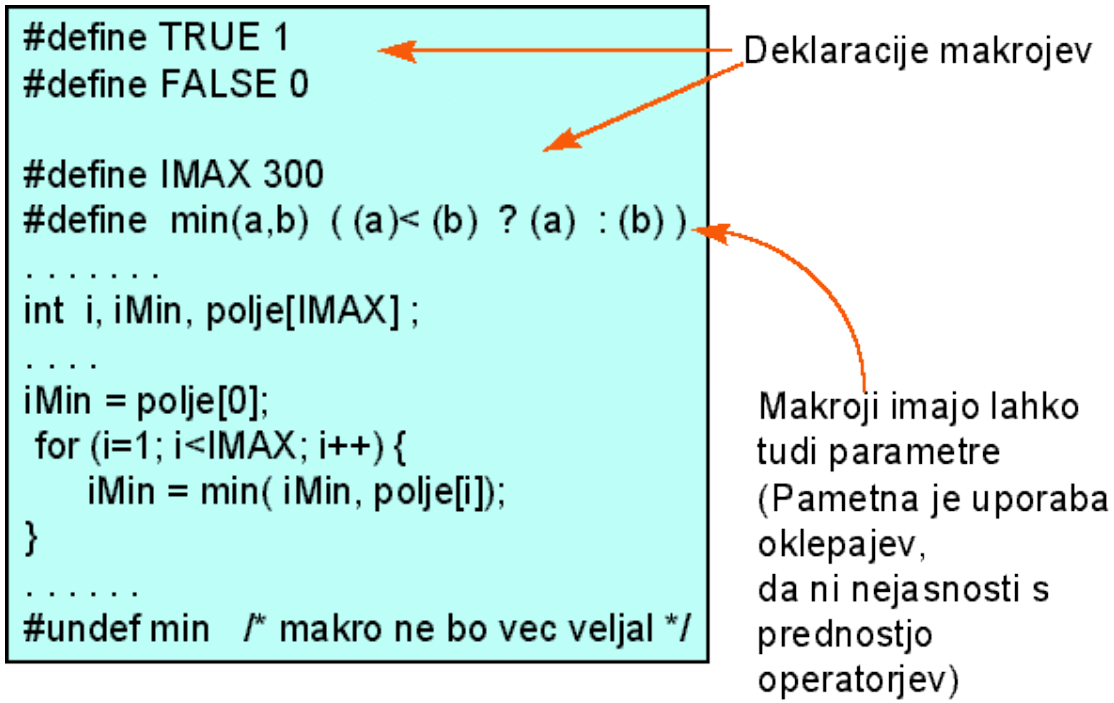
int abs (i)
ilong labs (k)
double fabs (x)
double fmod (x, y) *Vrne ostanek deljenja x / y*
double modf (x, pd)
double ldexp (x, i) *Vrne (x* (2 na i))*
double frexp (x, pi)
double floor (x) *Vrne največji integer, ki se gre v x*
double ceil (x) *Vrne najmanjši integer,ki ni manjši od x*
double sqrt (x)
double pow (x, y)

double sin (x) , double cos(x) , double tan(x), double asin (x), double acos (x)
double atan (x) .
double atan2 (x) *Vrne arctangens y/x. Uporabi predznaka argumentov za določitev kvadranta*
double exp (x) double log (x) double log10 (x)

C-jev predprocesor

Pred samim prevajanjem pregleda program v jeziku C predprocesor. Ta spozna navodila (direktive), za katera je značilno. da se začnejo z znakom #. Predprocesorju običajno povemo, katere datoteke naj vljuči v nas program (#include...), Deklariramo makroje (#define..), ki jih nato v našem programu razširi. Lahko tudi pogojimo, katere dele našega programa naj vključi v prevedeno kodo in katere ne.

Definicija in uporaba makrojev – primer:



Pogojno prevajanje

Primer

```
. . . . .
#ifdef min
   minimum = min(a,b);
#else
   if(a <b) minimum = a;
   else minimum = b;
#endif
. . . . . .
```

Direktivi sledi skupina stavkov (v novi vrsti)

Pregled direktiv za pogojno prevajanje

#if (kostantni izraz)

#if . . . defined (simbol)

#ifdef simbol

#ifndef simbol

#else

#elif simbol

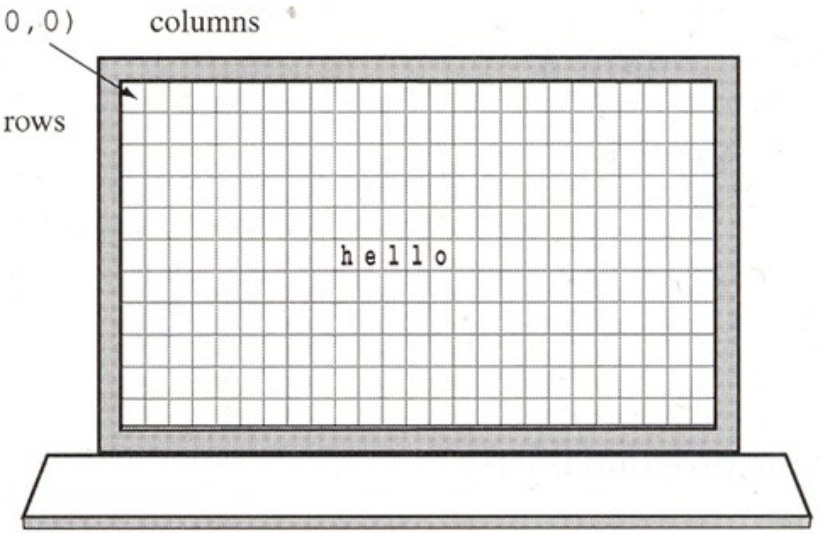
#endif

DEMO

DEMO 2

Krmiljenje zaslona

Pri nekaterih programih želimo "zaslonsko usmerjeno" zapisovanje in branje. V primeru operacijskega sistema LINUX si pomagamo s knjižnico funkcij z imenom "curses". Struktura programa, ki uporablja curses




```
#include < curses.h >
. . . . .
initscr();  /* iniciacija zaslona */
cbreak();   /* razlicne nastavitve */
nonl(); noecho(); . . . . .
while (!done) {
      /* nekaj primerov klicov za zapis na zaslon */
      move(row, col);
       addch(ch);
       printw (" Vrednost = %d \n ", vrednost);     . . . .
       refresh( );   /* azuriranje zaslona */     . . . .
}
endwin( );  /* pred izstopom pocisti za sabo */ exit (0);
```

Prevod in povezovanje programa: gcc <program file> -lncurses

Boljši primer

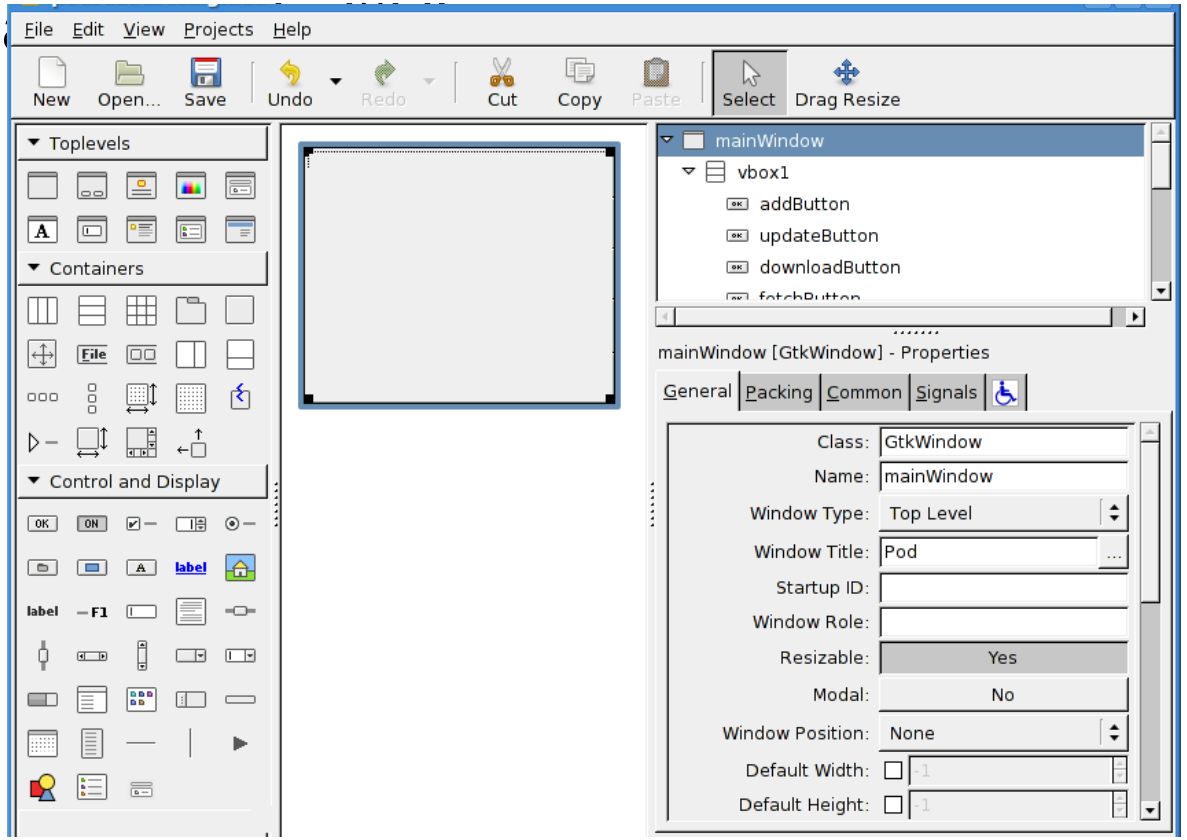
```
#include <ncurses.h>

int main() {
   int ch;

    initscr();                  /* Start curses mode                */
    raw();                      /* Line buffering disabled   */
    keypad(stdscr, TRUE);       /* We get F1, F2 etc..              */
    noecho();                   /* Don't echo() while we do getch */

     printw("Type any character to see it in bold\n");
     ch = getch();              /* If raw() hadn't been called
                                 * we have to press enter before it
                                 * gets to the program              */
     if(ch == KEY_F(1))         /* Without keypad enabled this will */
          printw("F1 Key pressed");/*  not get to us either */
                                /* Without noecho() some ugly escape
                                 * charachters might have been printed
                                 * on screen                        */
     else {
         printw("The pressed key is ");
         attron(A_BOLD);
         printw("%c", ch);
         attroff(A_BOLD);
     }
     refresh();                 /* Print it on to the real screen */
     getch();                   /* Wait for user input */
     endwin();                  /* End curses mode          */

     return 0;
}
```

Demo

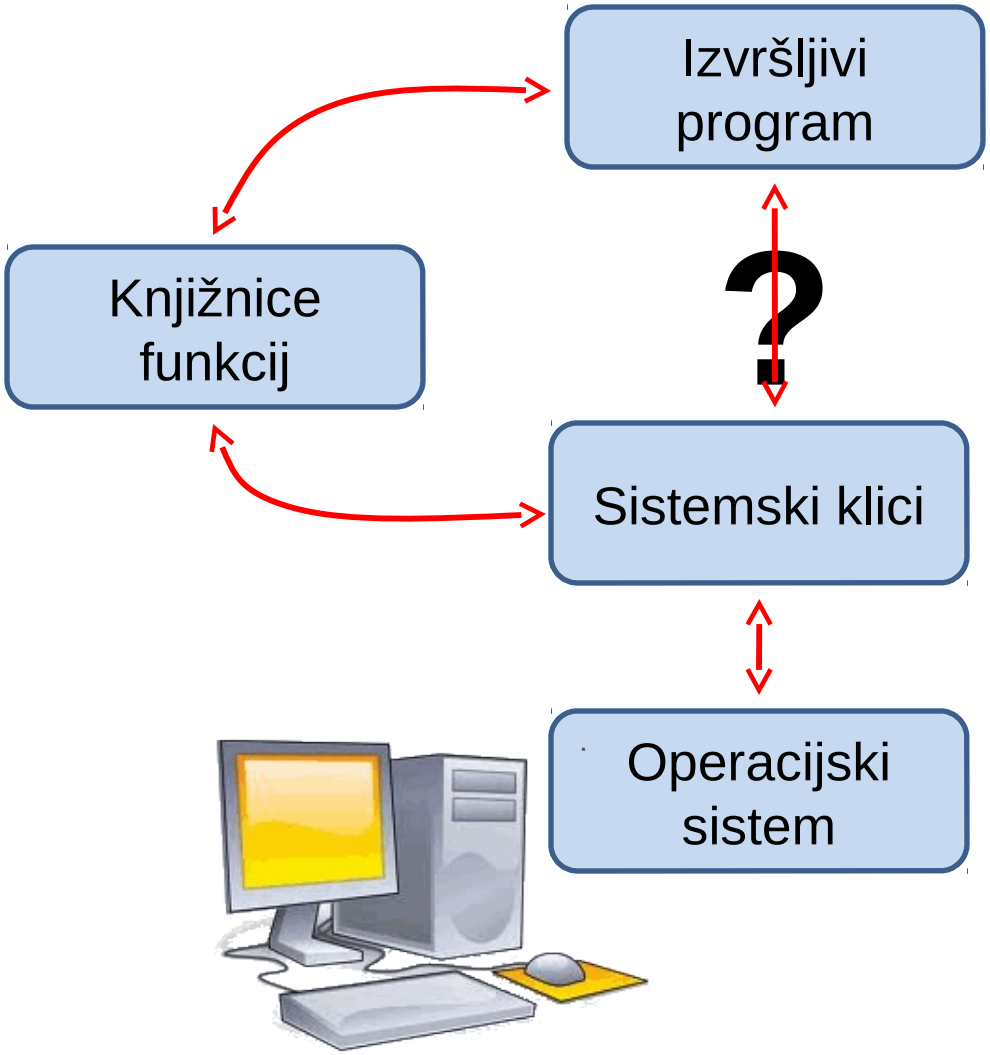
Krmiljenje zaslona

Raje uporabljajmo druga orodja (c#, java, delphi,...)

… To pa je že zgodb
vmesnikih:



Uvod v sistemsko programiranje



Sistemski klici za delo z datotekami

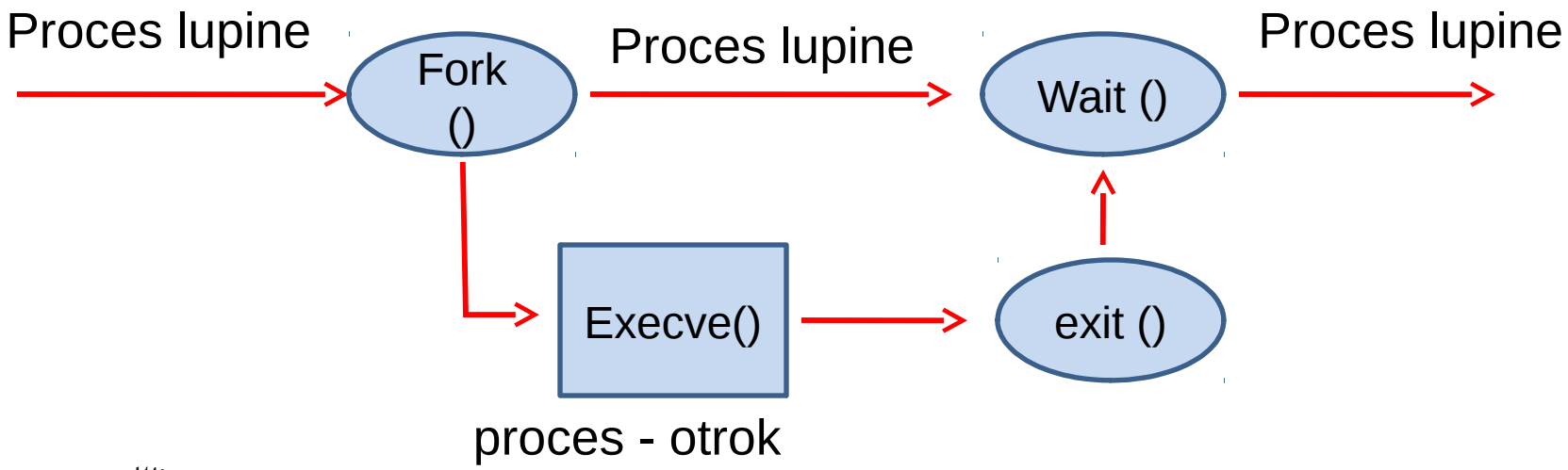
System call	Description
fd = creat(name, mode)	One way to create a new file
fd = open(file, how, ...)	Open a file for reading, writing or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information
s = fstat(fd, &buf)	Get a file's status information
s = pipe(&fd[0])	Create a pipe
s = fcntl(fd, cmd, ...)	File locking and other operations

- **s** je koda napake
- **fd** je opisnik datoteke
- **position** je položaj v datoteki

Delček kode, ki ponazarja tipično zaporedje dogodkov:

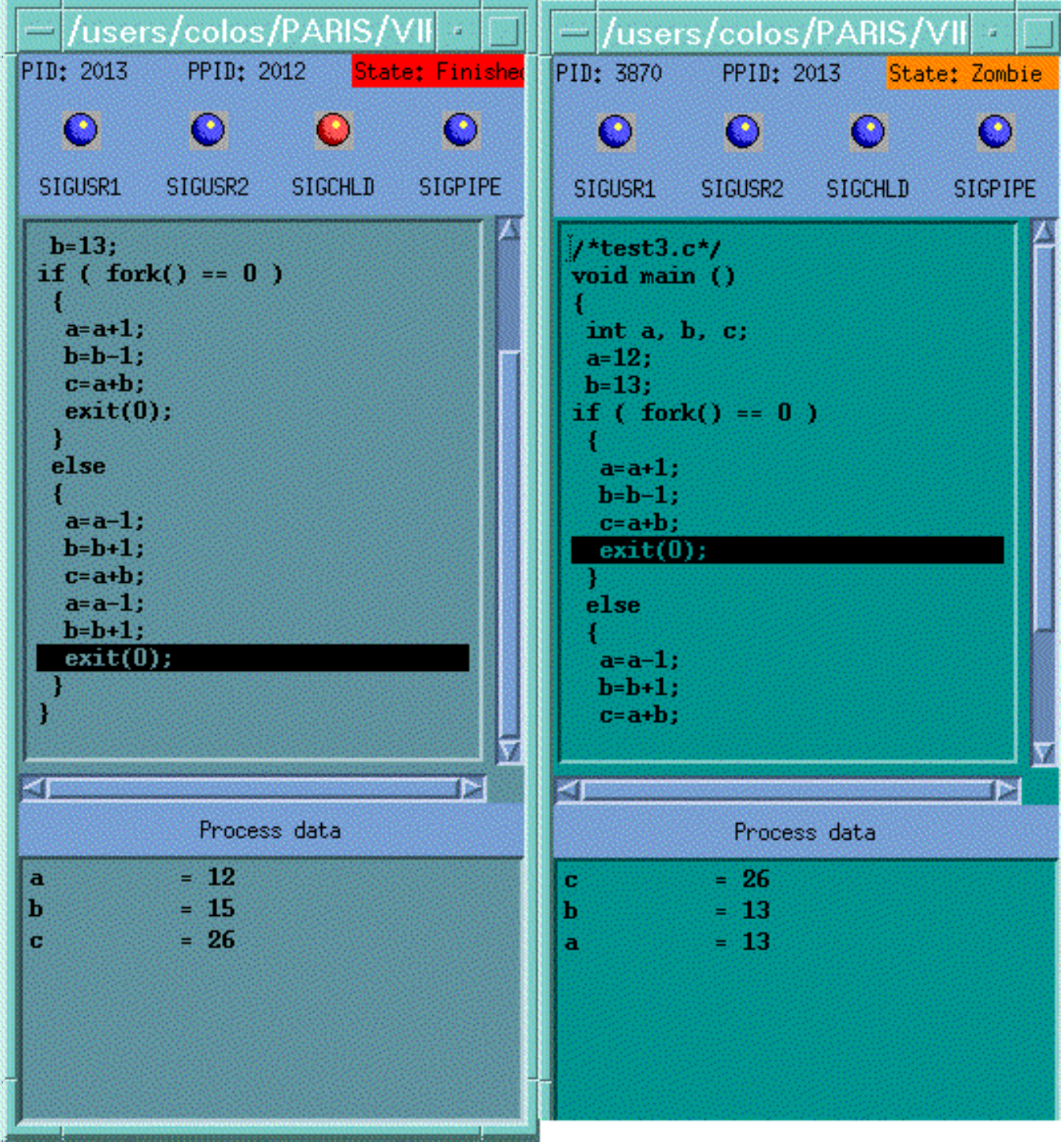
```
int fd; /*File descriptor */
...
fd = open (fileName, ...); /* Open file, return file desciptor */
    if (fd == -1) { /* Set some error condition */ }
...
read (fd, ...); /* Read from file */
...
write (fd, ...); /* Write to file */
...
lseek (fd, ...); /* Seek within file */
close (fd); /* Close the file, freeing file descriptor */
```

Kako deluje lupina LINUX



Poenostavljena koda lupine

```
while (TRUE) {                  /* repeat forever */
    type_prompt();                  /* display prompt on the screen */
    read_command(command,params); /* read input line from keyboard */

    pid = fork();                   /* fork of a child process */

    if(pid !=0) {
        waitpid (-1, &status,0);     /*parent waits for child /*
    }
   else if (pid>0) {
        execve (command, params, 0); /* child does the work*/
  }
```



Animirana demonstracija

Primerjava sist.klicev UNIX in Win32

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time