Tartalomjegyzék

- 1 A gyakorlat anyaga röviden
  - 1.1 Billenty?zet beállítása
  - 1.2 Fájl szerkesztése
  - 1.3 Jogok
  - 1.4 Honlap létrehozása
- 2 Gimp használata
- 3 Szabó Péter által összeállított oldal:

A gyakorlat anyaga röviden

Felzárkóztunk a szerdai csoporttal, így a következ?kben ismétlek az els? gyakorlaton vett dolgokból:

Billenty?zet beállítása

Ezt majdnem mindenkivel megcsináltuk, így vázlatosan: Gnome ablakkezel?t indítottunk, és a fenti panelre jobb klikk, majd hozzáadtunk (Add) egy Keyboard indicatior nev? ikont. Ez az ikon bal egérgombbal a billenty?zetkiosztások közötti váltást valósítja meg, jobb egérgombra el?jöv? menüb?l a Preference alatt a második fül (Layout) lehet módosítani/hozzáadni a kívánt kiosztást. A magyar kiosztású qwerty és qwertz az y és z helyét jelöli.

Fájl szerkesztése

Terminálból indítottunk gedit-et, majd elmentettük a létrejött fájlt. A fájl mentésekor a Browse for other folders alatt tudtuk megválasztani a mappát (a mappába be is kellett lépni), fent pedig megadtuk a fájl nevét. A kódolást UTF-8 -on hagytuk, err?l volt szó el?adáson.

Jogok

Az "http://wiki.math.bme.huls -la"http://wiki.math.bme.hu parancs részletes listát készít, amelynek els? oszlopa egy 10 karakterb?l álló sztring.

```
-rw-r--r--  1 morap student     2388 2007-07-30 03:51 .bashrc
-rw-r--r--  1 morap student   117660 2007-08-19 17:36 .B.blend
drwxr-xr-x  5 morap student     4096 2007-08-19 17:35 .blender
drwx------  5 morap student     4096 2007-08-21 20:50 .cache
drwx------  2 morap student     4096 2007-08-28 00:23 .chewing
```

Az els? bet? utal a típusra, - esetén fájlról, d esetén könyvtárról van szó. Az azt követk? hármas a user (felhasználó, továbbiakban u), a következ? hármás a group (félhasználó csoportja, továbbiakban g), majd az others (mindenki más, továbbiakban a) jogairól szól. Szokás még "http://wiki.math.bme.hua"http://wiki.math.bme.hu-val jelölni az all-t, vagyis mindenkit. Az r olvasás, a w írás, az x pedig futtathatóságot jelent. Változtatás chmod paranccsal.

Jogok állítására példa:

```
chmod "http://wiki.math.bme.huog-w"http://wiki.math.bme.hu file   #leveszi az írásjogot a csoport
chmod "http://wiki.math.bme.hua+x"http://wiki.math.bme.hu file    #Az a bet? az ALL-ra utal, minde
```

Aki szeretne olvasni b?vebben a témáról: ELTE-s tárgy honlapja, Jogkezelés linux alatt

Honlap létrehozása

Létrehoztunk egy public_html könyvtárat a saját mappánkban:

```
cd
mkdir public_html
```

gedit segítségével létrehoztunk egy index.html-t a saját mappánkban a következ? tartalommal:

```
<html>
<head>
<title>Ez a honlap címe</title>
</head>
<meta http-equiv="http://wiki.math.bme.huContent-Type"http://wiki.math.bme.hu content="http://wiki
árvízt?r? tükörfúrógép

Ez a br parancs miatt új sorban van.
<img src="http://wiki.math.bme.hukep.jpg"http://wiki.math.bme.hu>
</html>
```

és adunk futtathatósági jogot a public_html-re:

```
chmod "http://wiki.math.bme.hua+x"http://wiki.math.bme.hu ~/public_html
```

Akkor a www.math.bme.hu/~loginnev alatt megjelenik a honlapunk. Szükség van még arra, hogy az index.html-re adjunk olvasás jogot, de alapbeállítás szerint a login-otok alatt minden létrehozott fájlon van olvasás jog. További információ a honlapszerkesztésr?l a megfelel? wiki oldal(unk)on.

Gimp használata

Létrehoztunk egy kep.jpg nev? fájlt, amit a public_html-be mentettünk el. Ez is megjelenik a honlapon, a html forrás egyik sora erre hivatkozik.

Szabó Péter által összeállított oldal:

Elméleti tematika:

- felhasználói felületek
  - grafikus felület (pl. Ctrl-Alt-F7)
    - ablak
    - ablak címsora
    - start menü
    - program indítása (pl. Alt-F2)
    - tálca (taskbar)
    - virtual desktop (pl. Ctrl-Alt-Bal, Ctrl-Alt-Jobb)
  - szöveges, parancssori felület (command line) (pl. Ctrl-Alt-F1 ... Ctrl-Alt-F6)
    - karakterek téglalap alakú elrendezése
    - parancssor
    - szöveges, nem parancssori programok, pl. webböngész?k

            · Azért szeretjük a paranccsori és az egyéb szöveges programokat, mert ezeket távolról (pl. ssh-n keresztül) is lehet futtatni, tehát a parancs kiadásához és a program vezérléséhez nem kell az adott gép mellett ülni, elég, ha van internet-kapcsolat.
    - terminálablak (pl. xterm)
- hardver (hardware) (a felsorolás csak a legfontosabb hardvereszközöket említi)
    - számítógép (computer)
        - processzor (processor)
        - memória (memory)
        - háttértároló (storage)
    - kábel
    - küls? hardvereszközök
        - képerny? (monitor, screen, display)
        - billenty?zet (keyboard)
        - egér (mouse)
        - nyomtató (printer)
        - lapolvasó (scanner)
        - fényképez?gép (camera), videókamera (camcoder), telefon, PDA stb.
    - hálózati eszköz
        - modem
        - hub
        - kapcsoló (switch)
        - router, gateway, t?zfal
        - vezetéknélküli elérési pont (wireless access point)
- szoftver
    - alkalmazás (application) (a felsorolás nem teljes)
        - (sima) szövegszerkeszt? (text editor)
            - Gedit (Unix, GNOME)
            - Kate (Unix, KDE)
            - Notepad++ (Windows)
            - Textpad (Windows)
            - SCIte (Unix és Windows)
            - vim (Unix terminálablak, Windows)
            - EMACS (Unix és Windows)
            - joe (Unix terminálablak)
            - Notepad = Jegyzettömb
        - dokumentumszerkeszt? (word processor)
            - OpenOffice Writer (Unix és Windows)
            - Microsoft Word (Windows)
        - táblázatkezel? (spreadsheet)
            - OpenOffice Calc (Unix és Windows)
            - Microsoft Excel (Windows)
        - webböngész? (web browser)
            - Mozilla Firefox = IceApe (Unix és Windows)
            - Opera (Unix és Windows)
            - Microsoft Internet Explorer (Windows)
            - links, links2 (Unix, szöveges)
            - lynx (Unix, szöveges, régi)
        - médialejátszó (media player)
            - MPlayer (Unix és Windows)
            - Windows Media Player (Windows)
        - levelez?program = levelez?kliens (mail client, mail user agent, MUA)
            - Mozilla Thunderbird = IceDove (Unix és Windows)

        - Microsoft Outlook (Windows)
        - Pine (Unix és Windows0
        - Mutt (Unix)
    - computer algebra program
        - Maple (Unix és Windows): numerikus számolás, absztrakt és pontos számolás (törtek, gyökök, szögfüggvények stb.), analízis (határérték, deriválás, integrálás), függvényábrázolás stb.
        - Mathematica (Unix és Windows): a Maple-hez hasonló célokra
        - GAP (Unix és esetleg Windows): absztrakt algebrához (permutációk, csoportok, testek, hálók stb.)
        - Matlab (Windows): gyors mátrixm?veletek, de szimbolikus számolásra nem képes
        - Octave, SciLab (Unix és Windows): két ingyenes majdnem Matlab-klón
    - fájlkezel? (file manager)
        - Norton Commander-klón
            - Midnight Commander (Unix)
            - Total Commander (Windows)
            - FAR Manager (Windows)
            - Volkov Commander (DOS)
        - Windows intéz? (Windows
        - Nautilus (Unix, GNOME)
    - ablakkezel? (window manager, desktop-kezel? (desktop environment)
        - Windows ablakkezel? (Windows 95, Windows XP, Vista)
        - icewm (Unix)
        - GNOME desktop (Unix, GNOME)
        - KDE desktop (Unix, KDE)
        - xfce (Unix)
    - terminálablak (terminal window)
        - Windows Parancssor (Windows)
        - XTerm (Unix)
        - GNOME Terminal (Unix, GNOME)
        - KDE Terminal (Unix, KDE)
    - folyamatkezel? (task manager)
        - Task Manager (Windows)
        - top (Unix terminálablak)
        - ps (Unix parancssor)
    - parancsértelmez?, burok (shell)
        - command.com, cmd.exe (Windows)
        - bash (Unix)
        - zsh (Unix)
    - távoli parancssor (remote shell)
        - ssh (f?leg Unix)
        - PuTTY (ssh-kliens Windowsra)
    - szöveges program továbbfuttatása (terminal session management)
        - screen (Unix)
    - szoftverfejlesztést segít? program (pl. fordító (compiler))
    - játékprogram (game)
- operációs rendszer (operating system)
    - operációs rendszerek (a lista egyáltalán nem teljes)
        - Microsoft által gyártott
            - MS-DOS
            - Windows 3.1
            - Windows 95

- Windows NT
- Windows XP
- Windows Vista
- Windows CE (nem PC-re, hanem kéziszámítógépekre)

· Unix

- Linux (szabad szoftver, egyes fajtái fizet?sek)
  - Debian disztribúció (szerverekre ajánlott)
    - Sarge kiadás
    - Etch kiadás (ez van fent a .math.bme.hu laborokban)
  - Ubuntu disztribúció (kezd?knek ajánlott)
  - UHU disztribúció (magyar fejlesztés kezd? magyaroknak ajánlott)
  - Gentoo disztribúció (profiknak ajánlott)
  - Red Hat, Fedora
  - Mandrake
- FreeBSD (szabad szoftver)
- egyéb BSD-k (szabad szoftver)
- Mac OS X
- Solaris (a Sun gyártja, egyes fajtái ingyenesen letölthet?k)
- AIX (az IBM gyártja)

· egyéb

- FreeDOS (szabad szoftver)

◊ kernel

· A kernel a gép bekapcsolása során induló, kikapcsolásig futó alapszoftver.

· A kernel a hardverer?forrásokat kezeli és teszi elérhet?vé az alkalmazások számára.

· feladatai

- memóriagazdálkodás (pl. cache, swap)
- folyamatütemezés (CPU scheduling, process scheduling)
- hardvereszközök kezelése: I/O (bemenet--kimenet), I/O ütemezés (I/O scheduling), eszközmeghajtók
- védelem (protection): jogosulatlan m?veletvégrehajtás megtagadása
- energiagazdálkodás (akkumulátor, energiatakarékos mód, halk mód)
- fájlrendszerek (filesystem) kezelése
- lemezkezelés (pl. redundancia RAID-del)
- folyamatok közti kommunikáció (interprocess communication) bonyolítása
- internetes és egyéb hálózati kommunkáció bonyolítása a megfelel? protokollokkal

· rendszerhívás (system call): olyan függvény/eljárás/programrészlet a kernelben, melyet az alkalmazás hív

- céljai, linuxos példákkal
  - nyitott fájlok kezelése: open(), creat() close(), pipe(), dup(), dup2(), read(), write(), lseek(), select(), fcntl(), ioctl(), fstat() stb.
  - fájl- és könyvtárkezelés: link(), unlink((), symlink(), rename(), mkdir(), rmdir(), mknod(), chmod(), chown(), lchown(), stat(), lstat(), utime(), access()
  - internetes kommunikáció: socket(), socketpair(), connect(), bind(), listen(), accept(), setsockopt() stb.
  - folyamatkezelés: pause(), getpid(), getppid(), exit(), kill(), fork(), execve(), wait(), waitpid(), nice(), umask(), chdir()

stb.
- ♦ memóriagazdálkodás: sbrk(), mmap(), munmap() stb.
- ♦ id?kezelés: time(), stime(), times() stb.
- ♦ signalkezelés: alarm(), sigaction(), ssetmask(), sgetmask() stb.

- Az egyes rendszerhívásokról terminálablakban leírás olvasható, pl. `man 2 pipe` paranccsal. A leírás C programozási nyelvet feltételez.

◊ beállítóprogramok
· Windows Control Panel = Vezérl?pult (Windows)
· GNOME Control Center (Unix, GNOME)
· KDE Control Center (Unix, KDE)
· Windows Registry Editor (Windows)

◊ segédprogramok (tools, utilities)

- folyamat
  - ♦ a folyamat (process) fogalma
  - ♦ a szál (thread) fogalma
  - ♦ a folyamat részei
    - ◊ memóriaterület (memory image): programkód és adatok (Neumann-elv: progam és adat ugyanabban a memóriában van)
    - ◊ futási verem (stack): épp hol tart a végrehajtás
    - ◊ a programkódot tartalmazó fájl neve (binary executable) (pl. `/bin/ls`)
    - ◊ indítási argumentumok (paraméterek, command line arguments): stringekb?l álló lista
    - ◊ környezeti változók (environment variables): kulcs=érték párok listája, pl. `HOME=/home/student/pts`
    - ◊ nyitott fájlok (open files) és egyéb fájlszer? dolgok (pl. cs?vezetékek (pipe), socket-kapcsolatok)
      - · stdin (0, <): szabványos bemenet, alapesetben a terminálablakba gépelt szöveg innen olvasható
      - · stdout (1, >): szabványos kimenet, alapesetben a program által erre írt szöveg a terminálablakban jelenik meg
      - · stderr (2, 2>): szabványos hibakimenet, alapesetben a program által erre írt hibaüzenetek a terminálablakban jelennek meg
    - ◊ egyéb er?források (resources) (Unix alatt pl. osztott memóriaterületek (shared memory), szemaforok (semaphores), signal handlerek (signal handlers))
  - ♦ folyamatok és folyamathierarchia Unix alatt
    - ◊ PID (Process ID, folyamat-azonosító): 1 és 65535 közötti egész szám
    - ◊ Rendszerindításkor az init folyamat (`/sbin/init`) indul, PID=1.
    - ◊ Egy futó folyamat lemásolhatja magát a fork() rendszerhívással.
    - ◊ Minden másolat megjegyzi az ? szül? folyamatának PID-jét (PPID, Parent PID), ezáltal egy (gráfelméleti értelemben) fa alakul ki, melynek gyökere az init folyamat.
    - ◊ gyermek folyamat (child process)
    - ◊ A gyermek folyamat (majdnem) minden tekintetben a szül? (parent process) másolata lesz, örökli pl. a fork() pillanatában nyitott fájlokat és a környezeti változókat.
    - ◊ A folyamat befejez?dhet egy kilépési kóddal (exit code, 0..127) vagy signallal (pl. SIGINT = Ctrl-C, SIGQUIT = Ctrl-Zárószöglet, SIGTERM, SIGKILL, SIGHUP = terminálablak becsukása, SIGSEGV = hibás memóriam?velet). A kilépési kódos befejez?dést a folyamat saját maga kezdeményezheti, a signalos befejez?dést akárki (kill() rendszerhívás, kill, killall és pkill parancsok).
    - ◊ A gyermek folyamat befejez?désér?l a szül? kap értesítést az operációs rendszert?l (wait() rendszerhívás). A szül? megkapja a kilépési kódot vagy a befejezést okozó

signal kódját.

- ◊ zombi folyamat (zombie process): olyan folyamat, amely már befejez?dött, de a szül?je még nem kérte az értesítést a befejez?désr?l.
- ◊ Ha egy szül? folyamat befejez?dik, a még futó gyermekeit az init folyamat örökli.
- ◊ Ha az init folyamat befejez?dik, akkor a gép leáll (kernel panic-kal).
- ◊ Egy folyamat saját maga helyén egy másik programot indíthat (execve() rendszerhívás). Meg kell adni az indítandó program fájlnevét (pl. /bin/ls, /usr/bin/less), az indítási argumentumokat és a környezeti változókat.
- ◊ Másik program indításakor a folyamat legtöbb része megváltozik (pl. memóriaterület, futási verem, fájlnév, indítási argumentumok, környezeti változók), más részek pedig megváltozhatnak (pl. nyitott fájlok, signal handlerek).
- ◊ Ha program indításakor a fájlnévben van perjel (pl. `/bin/ls`, `./a.out`), akkor a megadott könyvtárban keresi a rendszer a programfájlt, ellenkez? esetben a `PATH` környezeti változóban kett?sponttal elválasztot könyvtárakat próbálja (balról jobbra) végig.
- ◊ Bonyolult példa: olyan programot írunk, amely az `ls` parancs kimenetének sorait számolja meg. Ehhez a program fork()-ol, a gyermek folyamat execve()-zi az *ls* parancsot. Ahhoz, hogy a parancs kimenete a szül?höz eljusson, a fork() el?tt a szül? egy cs?vezetéket hoz létre, melynek egyik végére a gyermek ír, a másik végét pedig a szül? olvassa. Az ellentétes végeket mindkét fél bezárja (tehát a szül? rögtön bezárja az írható véget, a gyerek pedig az execve() el?tt bezárja az olvasható véget). A gyermek az execve() el?tt a szabványos kimenetét a cs?vezetékre irányítja dup2()-vel, így az ls parancs ugyanúgy m?ködhet, mint terminálablak esetén (stdout-ra ír).
- ◊ shebang (`#!`): ha az execve()-vel futtatandó programfájl `#!` jelsorozattal kezd?dik, akkor a kernel az els? sorban megadott egy vagy két szót beszúrja a programnév és az argumentumok elé. Tehát ha például a `foo.sh` fájl els? sora `#! /bin/bash --`, akkor a `./foo.sh pikk pakk 42` parancs hatására a `/bin/bash -- ./foo.sh pikk pakk 42` parancs kezd futni, vagyis a `/bin/bash` program a `-- ./foo.sh pikk pakk 42` argumentumokkal.
- ◊ A shebang felhasználható ún. shell szkriptek (shell script) és egyéb szkriptek írására. A szkript tipikusan egy gyorsan megírt, rövid, tömör program (tipikus terjedelem: 100 .. 10000 bájt), ami egy konkrét feladatot automatizál: helyettesíti a mechanikus kattintgatást.
- ◊ multitaszk operációs rendszer (multitask operating system): Olyan operációs rendszer, amely egyszerre több folyamatot is futtatni képes (pl. Unix, Windows). Általában egyszerre több folyamat van, mint processzor, tehát az operációs rendszer kiválaszt néhány folyamatot, melyeket egy rövid ideig (néhány milliszekundum) futtat (minden processzorra 0 vagy 1 db folyamatot), majd a rövid id? letelte után esetleg más folyamatokat választ ki futásra. A folyamatok futásra kiválasztását ütemezésnek (CPU scheduling) nevezzük.
- ◊ system load (http://en.wikipedia.org/wiki/System_load): a load egy pozitív valós mér?szám arra, hogy hány aktív folyamat fut a rendszerben. Linux alatt aktívnak számít egy folyamat, ha épp ? fut, futásra kész (de várnia kell, amíg sorra kerül), vagy I/O (bemenet--kimenet) m?velet befejez?désére vár. A load azt is megmondja, hogy hány százalékos a processzorok kihasználtsága. Ha egy rendszeren pl. tartósan 6.0 közelében van a load, akkor a gépbe 6 processzort téve a teljesítmény fokozható, de 6-nál több processzor fölösleges, mert úgysem lesz mind kihasználva. Másik példa: ha egy 2 processzoros rendszeren tartósan 2.0 alatt van a load, akkor a processzorok nincsenek teljesen kihasználva, valamelyik processzor az ideje nagy részét várakozással tölti, hogy ? is kapjon feladatot. Ha egy asztali Linux rendszerhez percekig nem nyúlunk hozzá, akkor 0-hoz közeli, 0.1 alatti értékre csökken a load, mivel a gép nem kapott feladatot.

- shell-praktikák
  - a shell (burok, héj) f? feladatai
    - parancssor beolvasása (nyílbillent?k, Backspace stb. kezelése)
    - parancs értelmezése, a megfelel? program elindítása
    - stdin, stdout és stderr és egyéb fájlok átirányítása
    - folyamatok összekapcsolása cs?vezeték-rendszerben
    - fájlnevek kiegészítése (pl. `*.txt`)
    - a futó programok listázása, háttérbe és el?térbe rakása
    - programozási lehet?ségek (pl. változó, feltétel, ciklus, függvény) biztosítása, shell szkriptek futtatása
  - Részletes információ a Bash shellr?l a terminálablakban kiadott `man 2 bash` paranccsal.
  - Terminálablak nyitásakor a `~/.bashrc` fut le, ssh és egyéb belépés után a `~/.bash_profile`.
  - Az ékezetes bet?k parancssorba írásához és a PageUp-pal és PageDown-nal történ? history-kereséshez az ~/.inputrc fájlukba írjuk bele:

```
set meta-flag on
set input-meta on
set convert-meta off
set output-meta on
"http://wiki.math.bme.hu\e[5~"http://wiki.math.bme.hu: history-search-backward
"http://wiki.math.bme.hu\e[6~"http://wiki.math.bme.hu: history-search-forward
```

-
  - Ezután vagy nyissunk új terminálablakot, vagy adjuk ki a `bind -f ~/.inputrc` parancsot.
  - Az aktuális terminálablakban használhatjuk a *history* parancsot, egy már becsukott terminálablak esetén pedig a ~/.bash_history fájlt. Érdemes e fájlból a legfontosabb parancsokat másik fájlba írni, mert a bash a régi bejegyzéseket eldobja. A limit megnövelhet?, ha a ~/.bashrc-nkbe elhelyezzük az alábbi sort `export HISTSIZE=9999`.
  - Ha az ékezetes bet?k nem stimmelnek, akkor az alábbiak egyikét tegyük:
    - Adjuk ki az `export LC_CTYPE=hu_HU.UTF-8 LANG=hu_HU.UTF-8` parancsot, és pl. a Gnome terminál menüjében válasszuk az UTF-8 (Unicode) karakterkódolást.
    - Adjuk ki az `export LC_CTYPE=hu_HU.ISO8859-2 LANG=hu_HU.ISO8859-2` parancsot, és pl. a Gnome terminál menüjében válasszuk az ISO-8859-2 (Latin 2, Central European) karakterkódolást.
  - példák a shell szintaxisára
    - A `tac` program a bemenetét beolvassa, majd fordított sorsorrendben kiírja a sorokat.
    - A `rev` program a bemenetét soronként beolvassa, és minden sort fordott bet?sorrendben kiír.
    - Ctrl-C-vel kilépve megszakítjuk a program futását (tehát a *tac* nem ír ki semmit, és kilép).
    - Üres sorba írt Ctrl-D-vel kilépve jelezzük, hogy a bemenetnek vége (tehát a *tac* kiír mindent fordítva, majd kilép).
    - `ls /bin/bash /bin/cat`
    - `ls "http://wiki.math.bme.hu/bin/bash /bin/cat"http://wiki.math.bme.hu`
    - `ls '/bin/bash /bin/cat'`
    - `ls "http://wiki.math.bme.hu/bin/bash /bin/cat $PATH"http://wiki.math.bme.hu`
    - `ls '/bin/bash /bin/cat $PATH'`
    - `echo 'foo bar' | rev`
    - `(echo 'foo bar'; echo baz) | rev | tac`

        - `(echo 'foo bar'; echo "http://wiki.math.bme.huEz itt a $PATH"http://wiki.math.bme.hu) | tac`
        - `ls | wc`
        - `ls -a | wc -c`
        - `echo /bin/*x*`
        - `echo /bin/?a*`
        - `echo /bin/?x*`: mivel nincs egyezés, benne marad a kérd?jel
        - `ls /bin/?x*`
        - `echo ~`
        - `ls -d ~wettl/public_html/x*`
        - `ls -l --sort=size >csokkeno_lista.txt`
        - `ls -l --sort=size >/dev/null`
        - `ls /nincs.ilyen >/dev/null 2>hiba; cat hiba; rm -f hiba`
        - `ls /nincs.ilyen >/dev/null 2>&1 || echo hiba volt`
        - `ls / >/dev/null 2>&1 || echo hiba volt`
        - `<be.txt sort | rev >ki.txt`
        - `find ~ -type f | perl -pe 's@^.*/@@' | sort | uniq -cd | sort`: Gyakoriság szerinti csökken? sorrendben listázza azokat a fájlneveket, melyek 1-nél többször fordulnak el? az aktuális felhasználó home könyvtárában. A fenti parancs pipeline demonstrálja a Unix shellprogramozás erejét.
    - az *ls* program fontosabb kapcsolói
        - `-l`: részletes, többoszlopos kimenet
        - `-a`: a ponttal kezd?d? fájlneveket is
        - `-d`: a parancssorban megadott könyvtáraknak csak a nevét, ne a tartalmát
        - `-R`: a könyvtárakba rekurzívan belemenni
- gyakorlati feladatok Linux, GNOME környezetben
    - Milyen folyamatok futnak éppen?
        - `ps aux` parancs
            - a: ne csak az aktuális felhasználó folyamatait mutassa
            - u: mutassa a felhasználónevet is
            - x: ne csak a terminálon (terminálablakban stb.) futó folyamatokat mutassa
        - `top` terminálablakos program: alapból CPU-használat szerinti növekv? sorrendbe teszi a folyamatokat, máshogy is tudja rendezni
        - grafikus eszköz
    - Az aktuális felhasználó milyen folyamatokat futtat éppen?
        - `ps x` parancs
    - Mennyi a system load?
        - `uptime` parancs mutatja az utolsó 1 perc, 5 perc és 15 perc átlagát
        - a `top` is mutatja a fels? sorban
        - grafikus eszköz
    - Melyik folyamatunk ragadt be, és zabálja a sok processzorid?t?
        - a `top` rögtön fölül írja ki
    - Melyik folyamatunk ragadt be, és zabálja a sok memóriát?
        - a `top`-ban Shift-O, N, Enter, majd rögtön fölül írja ki
    - Állítsuk le (l?jük ki) a 42-es PID-? folyamatot!
        - `kill 42`: ha a folyamat beragadt, akkor lehet, hogy a killnek nincs hatása
        - `kill -9 42`: szinte tuti kilövi (mert SIGKILL signalt küld)
    - L?jük ki az összes saját IceApe-et!
        - `ps x | grep -i iceape`: listázza
        - `kill -9 ...`: PID alapján kilövi
        - `killall -9 iceape-bin`: parancsnév egyezése alapján kilövi
        - `pgrep -u $EUID '^iceape-bin$'`: csak a PID-eket listázza
        - `ps u `pgrep -u $EUID '^iceape-bin$'`: mindent listáz

- ◊ `pkill -9 -u $EUID '^iceape-bin$'`: parancsnév illeszkedése kilövi
- ♦ Keressük meg az összes foo-t tartalmazó fájlnevet!
  - ◊ `find / 2>/dev/null | grep '/[^/]*foo[^/]*$'`
    - · Ctrl-Z-vel állítsuk le
    - · `jobs`-szal listázzuk ki az adott shellb?l indított feladatokat
    - · `fg %1`-gyel folytassuk (1 helyébe esetleg mást írva)
    - · Ctrl-Z-vel állítsuk le
    - · `bg %1`-gyel folytassuk a háttérben
    - · `fg %1`-gyel folytassuk (az el?térben)
    - · Ctrl-C-vel vagy Ctrl-Zárószöglet-tel állítsuk le
    - · Ha beragad, se Ctrl-C, se Ctrl-Zárószöglet, se Ctrl-Z, se Ctrl-D nem m?ködik, akkor zárjuk be a terminálablakot, és szükség esetén egy másik terminálablakból l?jük ki a megadott folyamatot.
- ♦ Milyen környezeti változók aktívak?
  - ◊ `env | sort` megmutatja a környezeti változókat
  - ◊ `set | sort` megmutatja a környezeti változókat és a shell bels? változóit is