

# Bevezetés a számítástechnikába

## #06 – $\text{\LaTeX}$ és GNU Linux

---

2023. október 30. – november 3.

**Naszlady Márton Bese** <naszlady@itk.ppke.hu>  
**Siklósi Bálint** <siklosi.balint@itk.ppke.hu>

## **#06/1 – Ábrák**

# Ábra környezet

- ▶ Ábrák beillesztése mindig problémás: Hova kerüljön a szövegen belül? Mekkora margókat hagyjunk a kép körül? Hogyan érhetjük el, hogy minden kép az egyes oldalakon azonos pozícióba kerüljön?
- ▶  $\text{\LaTeX}$  környezetben ezeket a funkciókat a  $\text{\LaTeX}$  fordító kezeli.
- ▶ Az ábra környezet segítségével képeket, ábrákat helyezhetünk el a dokumentumban.

# Ábra környezet

Közvetlenül beszúrhatunk PDF, PNG, JPEG formátumú képeket (gyakorlatilag ezek azok, amiket a PDF formátum támogat, mint beágyazható formátumokat) a következő parancsok segítségével:

```
\begin{figure}  
  \centering  
  \includegraphics{figure.png}  
  \caption{Ez egy beszúrt kép}  
\end{figure}
```

A parancsok működéséhez a `graphicx` csomag betöltése szükséges (`\usepackage`)

# Ábrák elhelyezése oldalon belül

Alapvetően a  $\text{\LaTeX}$  fordítóra van bízva az ábrák elhelyezése, de lehetőségünk van kívánalmakat megfogalmazni:

- h A fordító törekszik a képet ott elhelyezni, ahol azt definiáltuk a dokumentumban, ha talál megfelelő méretű helyet az oldalon
- t Az adott (vagy azt követő) oldal *tetején* helyezi el az ábrát.
- b Az adott (vagy azt követő) oldal *alján* helyezi el az ábrát.
- p Olyan oldalon helyezi el, ahol nincs szöveg, csak további ábrák.
- ! Még nyomatékosabban kérjük a  $\text{\LaTeX}$ et, hogy az általunk megadottak szerint próbálja elhelyezni az ábrát.

Példa:

```
\begin{figure}[!ht]
```

# Ábra méretének meghatározása

- ▶ Az `includegraphics` paramétereként lehet megadni, hogy mekkora méretben szeretnénk beilleszteni a képünket.
- ▶ Például az `\includegraphics[width=0.48\textwidth]` megadásával a szövegszélesség 48%-a lesz a képszélesség.
- ▶ A `width=1in` esetén egy inch szélességűre állítjuk a képszélességet. (Használható cm, vagy mm is a méret megadásához.)
- ▶ Az `\includegraphics[scale=0.5]` parancs segítségével nagyíthatunk/kicsinyíthetünk. (Ebben a példában a felére kicsinyítünk.)
- ▶ A `width` mellett a `height` is használható a képméret megadására. Mindkét esetben a nem megadott oldal méretét a képarányok megőrzésével határozza meg.

## #06/2 – Táblázatok

# A tabular környezet

Táblázatok a tabular környezettel helyezhetünk el a dokumentumban:

```
\begin{tabular}{|l||rc}  
  narancs & körte & szilva  \\  
  aa & bbbbbb& ccccc \\  
  \hline  
  szőlő & KV & füge  
\end{tabular}
```

Eredménye:

narancs	körte	szilva
aa	bbbbb	cccc
szőlő	KV	füge



## A table környezet

Ha a táblázatot is szeretnénk címmel ellátni (és a `figure` környezethez hasonlóan az automatikus elhelyezést is igénybe venni), akkor a `table` környezetet kell használni:

```
\begin{table}[h!]  
  \centering  
  \begin{tabular}{|l||rc}  
    narancs & körte & szilva & \\  
    aa & bbbbbb& ccccc & \\  
    \hline  
    szőlő & KV & füge  
  \end{tabular}  
  \caption{Az első táblázatom}  
\end{table}
```

Ebben a környezetben is használhatók a `figure` környezetnél megismert elhelyezéssel kapcsolatos opciók (`h`, `t`, `!` stb.).

## **#06/3 – Utalások**

- ▶ Ahhoz, hogy egy struktúrális elemre (fejezetre, alfejezetre stb.), ábrára, táblázatra hivatkozni tudjunk, az adott elemnél el kell helyeznünk a `\label{cimke}` parancsot. A `cimke` bármilyen szöveg lehet, ezt fogjuk használni a későbbi hivatkozás során.
- ▶ Fontos, hogy a címkét mindig az egység belsejében, a sorszám növelése után tegyük meg. Például `\section{}` címkézése:  
`\section{Cim} \label{cimke}`
- ▶ Ez után a `\ref{cimke}` használatával tudunk hivatkozni.
- ▶ Pl.: Ehhez hasonló módszert láthatunk a 13. dián is. (És ez a sorszám akkor is helyesen készül el, ha beszúrunk közben még egy rakás egyéb diát.)

# Ábrák, táblázatok címkézése

```
\begin{figure}  
  \centering  
  \includegraphics{figure.png}  
  \caption{Ez egy beszúrt kép}  
  \label{cimke}  
\end{figure}
```

A címkézés hasonlóan tehető meg a `table`, `equation` stb. környezetek esetén is.

## aref és Aref

- ▶ A `\usepackage[magyar]{babel}` használata esetén használhatóak az `\aref{}` és `\Aref{}` referenciakezelő parancsok is, amelyek a referált szám elé kiteszik a névelőt is (kisbetővel az `\aref{}` esetében és nagybetővel az `\Aref{}` esetében), így nem kell azzal sem foglalkoznunk, hogy a hivatkozott szám elé *a* vagy *az* névelőt kell tenni.
- ▶ Gondoljunk bele mi történik, ha beszúrunk egy sorszámozott részt a dokumentumba! Megváltoznak a sorszámok, egyes esetekben *a* névelő kell *az* helyett vagy fordíva. Az `\aref{}` és `\Aref` megoldja ezt a problémát.

## Oldalszámra hivatkozás


A `\ref{}`, `\aref{}`, `\Aref{}` sorszámra hivatkozó parancsok mellett lehetőség van arra is, hogy a hivatkozott elem oldalszámát jelentsük meg, amit a `\pageref{}`, `\apageref{}`, `\Apageref{}` parancsokkal tehetünk meg. A névelős változatok ugyanúgy működnek, mint azt az előzőekben megismertük.

# Irodalomjegyzék

- ▶ Ha hivatkozni szeretnénk egy cikkre, (pl.: floating point cikk [1]) akkor keressük meg annak a bibTeX összefoglalóját.
- ▶ Pl.: cikk → cite this → BibTeX
- ▶ Az összes hivatkozni kívánt cikk BibTeX leírását másoljuk be pl. egy *hivatkozasok.bib* nevű fájlba (Mindegyik bejegyzés első eleme legyen egy leíró azonosító. pl.: *hetero\_cikk*)
- ▶ Preambulumba:

```
\usepackage[style = ieee]{biblatex}  
\addbibresource{hivatkozasok.bib}
```

- ▶ Végül: \printbibliography

 D. Goldberg, "What every computer scientist should know about floating-point arithmetic," *ACM Comput. Surv.*, vol. 23, p. 5–48, mar 1991.

## #06/4 – További $\text{\LaTeX}$ finomságok



## Akit érdekel, az nézzen utána a következőknek:

- ▶ diasor
  - 1. beamer
- ▶ vonalas ábrák
  - 1. tikz
- ▶ programkódok beillesztése
  - 1. lstlistings
  - 2. verbatim
  - 3. minted
  - 4. algpseudocode (pszeudokódokhoz)
- ▶ kémiai formulák
  - 1. mhchem
- ▶ Alábrák
  - 1. subcaption

Készítsük el az alábbi Overleaf  $\LaTeX$  dokumentum végleges verzióját:

1. Lépünk be az <https://overleaf.com> oldalra!
2. Nyissuk meg az alábbi hivatkozást:  
<https://www.overleaf.com/read/gzhwwjmpzxvc>
3. A bal felső "Menu" gomb megnyomása után használjuk a "Copy project" lehetőséget; másoljuk le magunknak a projektet.
4. Töltsük le a projekt forrásai közül a `teljes_megoldas.pdf` fájlt, és nyissuk meg azt.
5. Az előzőekben látott formázási parancsok felhasználásával formázzuk meg a projektben olvasható nyers szöveget olyanra, mint a PDF.

# **#06/5 – Operációs rendszerek**

# Mi az operációs rendszer?

Az operációs rendszer olyan alapszintű programgyűjtemény, amely közvetítőként működik a felhasználó és a számítógép hardvere között.

Célja:

- ▶ kényelmesen használható felületet ad a felhasználók számára,
- ▶ egységesített környezetet nyújt a futtatandó programok számára,
- ▶ elrejt a hardver körülményességét és bonyolultságát,
- ▶ a rendelkezésre álló erőforrásokat okosan (optimálisan) használja.

## Kernel (mag) és Shell (héj)

Az operációs rendszereknek van két jól azonosítható része, ezek a **kernel** és a **shell** (mag és héj).

A *kernel* áll közvetlen kapcsolatban a hardverrel, itt történik meg a perifériák kezelése, a memóriák, háttértárak ügyes használata stb.

A *shell* áll közvetlen kapcsolatban a felhasználóval, itt történik meg az ember-gép interakció grafikus felületen (GUI), menürendszeren vagy szöveges parancsok begépelése révén.

# Kernel (mag) és Shell (héj)

A kernel feladatai:

- ▶ kezeli a hardver elemeket (driverok (illesztőprogramok) révén)
- ▶ kezeli a memóriát
- ▶ kezeli a futó programokat (folyamatütemező)
- ▶ kezeli a fájlrendszereket
- ▶ kezeli a kivételeket, megszakításokat

# Kernel (mag) és Shell (héj)

A shell feladatai:

- ▶ kezel valamilyen felhasználói felületet:
  - ▶ Graphical User Interface (GUI) grafikus felület;  
egérrel, nyilakkal navigálható, ábrák, színek jelentenek valamit
  - ▶ Command Line Interface (CLI) szöveges felület;  
begépeltek parancsok, a válaszok szintén szöveggé jelennek meg
  - ▶ egyéb lehetőségek, pl.
    - nyomógombokkal (kávéautomata, játékgyép)
    - hangvezérlés (Alexa, Siri)
    - taktilis kijelzés (látássérülteknek)
- ▶ kezeli a környezeti paramétereket (pl. nyelv, időzóna, ...)

## **#06/6 – Linux**



# Linux történelem

1983-ig népszerű gép: PDP10-es  
nyíltan elérhető, a kutatók egymást segítve tudnak dolgozni rajta.

Később: különböző számítógépek, különböző módon titkosított rendszerek  
tilos volt egymásnak segíteni

1984-ben létrejön a GNU projekt (GNU is not UNIX): <http://gnu.hu>

- ▶ a program szabadon használható bármilyen célra
- ▶ a programot bárki szabadon módosíthatja igényei szerint
- ▶ a programot bárki továbbadhatja, akár ingyen, akár pénzért
- ▶ a program módosított verzióinak meg kell felelnie ugyanezen feltételeknek

# Linux történelem

A GNU projekt keretében sok hasznos alkalmazás jön létre, de kernel még nincs.

Linus Torvalds: 1991-ben a MINIX hibáit javítva dolgozni kezd egy új, szabad operációs rendszeren *to* Linux kernel

GNU + Linux → GNU/Linux → Linux

Sorra jelennek meg különféle disztibúciók:

- ▶ debian
- ▶ ubuntu
- ▶ arch linux
- ▶ [stb.](#)

# Távoli gépek használata

Gyakran fordul elő, hogy az általunk használni kívánt számítógéphez nem közvetlenül ülünk oda.

Ilyenkor egy közvetítő számítógépet használunk fel, ami a tényleg használt gép operációs rendszeréhez csatlakozik, és segít a shell közvetett elérésében:

- ▶ például átküldi a monitor képét (TeamViewer, távoli asztal)
- ▶ például átküldi a szöveges interfészre beírt parancsokat és a kapott válaszokat.

**Az egyik ilyen távoli kapcsolódási módszert fogjuk SSH-nak (secure shell) hívni.**

# Távoli gépek használata

## A közvetlenül a birtokunkban lévő gépen

- ▶ fut valamilyen operációs rendszer  
*pl. Windows 10*
- ▶ van valamilyen felhasználónk  
*pl. Pista*
- ▶ vannak fájlok  
*pl. C:\Windows\System32\mspaint.exe*
- ▶ vannak futó programok  
*pl. webböngésző*

## A távoli, tényleg használni akart gépen

- ▶ fut egy **másik** operációs rendszer  
*pl. GNU Linux (Debian 11)*
- ▶ van valamilyen **másik** felhasználónk  
*pl. István*
- ▶ vannak **más** fájlok  
*pl. /var/log/access.log*
- ▶ vannak **más** futó programok  
*pl. webszerver*

A két gép között egy alkalmas programmal (pl. Putty) teremtünk kapcsolatot. Így lehetővé válik a fájlok mozgatása oda-vissza és a távoli gép shell-jének használata.

# Távoli gépek használata

Hallgatók által is használható ITK-s szerverek:

- ▶ `users.itk.ppke.hu` kari felhasználók egyszerű weboldalai számára
- ▶ `cortex.itk.ppke.hu` shell scriptek és cpp fordítások kipróbálására való szerver

Minden ilyen szerverhez az egyetemi azonosítóval (vezetéknévből + keresztnévből (+ számból) képzett felhasználónév) és az ahhoz tartozó jelszóval lehet belépni.

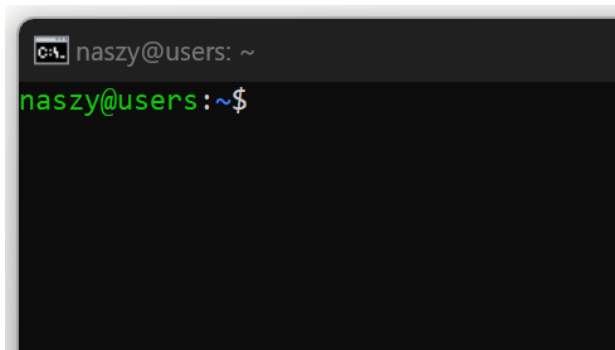
Windows alól javasolt a natív ssh program vagy a Putty program használata; Linux és MacOS alól az openssh kliens használata.

Bővebben lásd még: [csatlakozási lehetőségek ITK-s szerverekhez](#)

# Linux parancsok

**Terminál:** egy olyan program, amiben a shell által elérhetővé tett interfész használható.

A linux parancssori shell-jének használata közben az ún. **prompt** után tudjuk begépelni a parancsot. A prompt mutatja, hogy milyen felhasználóval, milyen gépen és a fájlrendszeren belül hol vagyunk.

A screenshot of a Linux terminal window. The window has a dark background. At the top left, there is a small icon of a terminal window followed by the text 'naszy@users: ~'. Below this, the prompt 'naszy@users: ~\$' is displayed in a green color. The rest of the terminal area is empty.

# Linux parancsok

A terminálon parancsok gépelhetők be. A begépelés során megadható a parancs (program) neve, és ha van illetve szükséges, akkor a parancs vagy program által használandó paraméterek, bemeneti adatok.

A begévelt parancs értelmezése az Enter lenyomása után kezdődik.

Néhány egyszerű program, aminek elég csak a nevét begépelni ahhoz, hogy a shell lefuttassa őket:

- ▶ `pwd` – kiírja az aktuális mappa elérési útját
- ▶ `who` – kiírja, hogy kiknek van aktív munkamenete a számítógépen
- ▶ `hostname` – kiírja a használt számítógép nevét
- ▶ `whoami` – kiírja, hogy ki a jelenlegi felhasználó

# Linux parancsok

Vannak olyan programok, melyek (opcionális) bemeneti paraméterekkel dolgoznak; a futás eredménye ezektől függ.

Néhány olyan program, ami paraméterek használatával működik

- ▶ `echo` – kiírja a paraméterként kapott értéket a terminálra
- ▶ `man` – a paraméterként megadott program leírását (manual) jeleníti meg
- ▶ `ls` – listázza a megadott mappa tartalmát
- ▶ `mkdir` – létrehozza a megadott nevű mappát

A paramétereket (argumentumokat) a program neve után szóközzel elválasztva írjuk be, és ha minden argumentumot megadtunk, akkor ezt követően nyomunk entert.



# Linux fájlrendszer

A GNU Linux operációs rendszer fájlrendszere egy **fa gráf**.

A gyökérelem neve: /

A gyökérelem alatt néhány igen fontos mappa van:

- ▶ `bin` – alapvető programok
- ▶ `boot` – a bootoláshoz való anyagok
- ▶ `etc` – beállítások
- ▶ `home` – felhasználók saját könyvtárai
- ▶ `tmp` – átmeneti fájlok

stb...

# Abszolút és relatív elérési útvonalak

## Abszolút elérési út

Egy fájl (erőforrás) abszolút elérési útja a gyökérelemtől kiindulva történik, és minden közben érintett mappa nevét felsoroljuk '/' jellel elválasztva.

```
/home/naszy/public_html/index.html
```

## Relatív elérési út

Az erőforrás eléréséhez azt adjuk meg, hogy a jelenlegi helyről hogyan érjük azt el: a fában hány szintet kell először felfelé menni, aztán pedig a másik ágon milyen mappákon keresztül kell lefelé szállni.

```
../../public_html/index.html
```

A relatív elérési utak esetében van két kitüntetett "mappanév":

- ▶ . ← a jelenlegi mappát jelenti
- ▶ .. ← a jelenlegi mappa szülőjét jelenti

# Linux parancsok

Vannak olyan parancsok, amik a fájlrendszerben való navigációt segítik. Ezeknek általában valamilyen (opcionális) elérési utat kell megadni, és adott esetben bizonyos (opcionális) kapcsolókkal is befolyásolható a működésük.

- ▶ `cd` ← az adott elérési út szerinti helyre visz
  - ▶ `cd ..` ← egy mappával feljebb visz
  - ▶ `cd /tmp` ← bárhol is vagy, a /tmp mappába teleportál
  - ▶ `cd ~` ← bárhol is vagy, a saját felhasználód home könyvtárába visz
- ▶ `ls` ← listázza a megadott mappa tartalmát
  - ▶ `ls -l` ← listaszerűen listáz
  - ▶ `ls -a` ← mindent is kilistáz (rejtett fájlokat is)
  - ▶ `ls -la` ← az előző kettőt összevonva csinálja

További [Linux parancsok és rövid leírásuk itt található](#).

## **#06/7 – Feladatok**

# Feladatok

1. Lépj be a `users.itk.ppke.hu` szerverre valamilyen terminálon keresztül
2. Hozd létre a következő oldalon lévő struktúrát, ahol:
  - ▶ a mappákat neked kell létrehozni
  - ▶ `sajat_nevem.txt` fájlt is neked kell létrehozni, tartalma legyen a saját neved
  - ▶ a többi fájlt másolhatod a `/home/naszy/bevszamtech` mappából

# Struktúra

```
gyakorlas
├── kepek
│   ├── allatok
│   │   ├── tukan
│   │   └── cat
│   └── emberek
│       ├── balint.rajz
│       ├── robotlab.rajz
│       ├── saját_nevem.txt
│       └── Mona Lisa.txt
└── irodalom
    ├── lorem_ipsum.txt
    └── vogon_vers.txt
```

# VÉGE



## PÁZMÁNY

Pázmány Péter Katolikus Egyetem  
**Információs Technológiai és Bionikai Kar**