Computing Laboratory

Technical Introduction

Malay Bhattacharyya

Assistant Professor

Machine Intelligence Unit
Indian Statistical Institute, Kolkata

October, 2021

Let's play a game!!!

Suppose you are playing a game in turn with the computer. Total n number of sticks are to be picked up in this game. Whoever picks the last one loses the game. Neither the computer nor you can pick up more than 3 sticks at a time. Nobody can skip a turn, i.e. at least one stick is to be picked up in a turn. Write a program to ensure that the computer wins optimally (whenever there is a chance) irrespective of the turn.

What we learnt?

Solving a problem on a computer requires:

- Data structures
  1. Storing the data effectively and efficiently.
- Algorithms
  1. Understanding the logical actions (strategies) to take on the data.
  2. Choosing the best strategy.
  3. Finding out the best implementation of the chosen strategy.

Let's take a problem

Given a bunch of 52 pages, one per student, order them following the order of roll numbers.

Data structures

- **Question:** How to store your data?

Data structures

- **Question:** How to store your data?
- **Counter-question:** What kind of operations will be needed?

Data structures

- **Question:** How to store your data?
- **Counter-question:** What kind of operations will be needed?
- **Answer:** Access the i-th element of a list.

Data structures

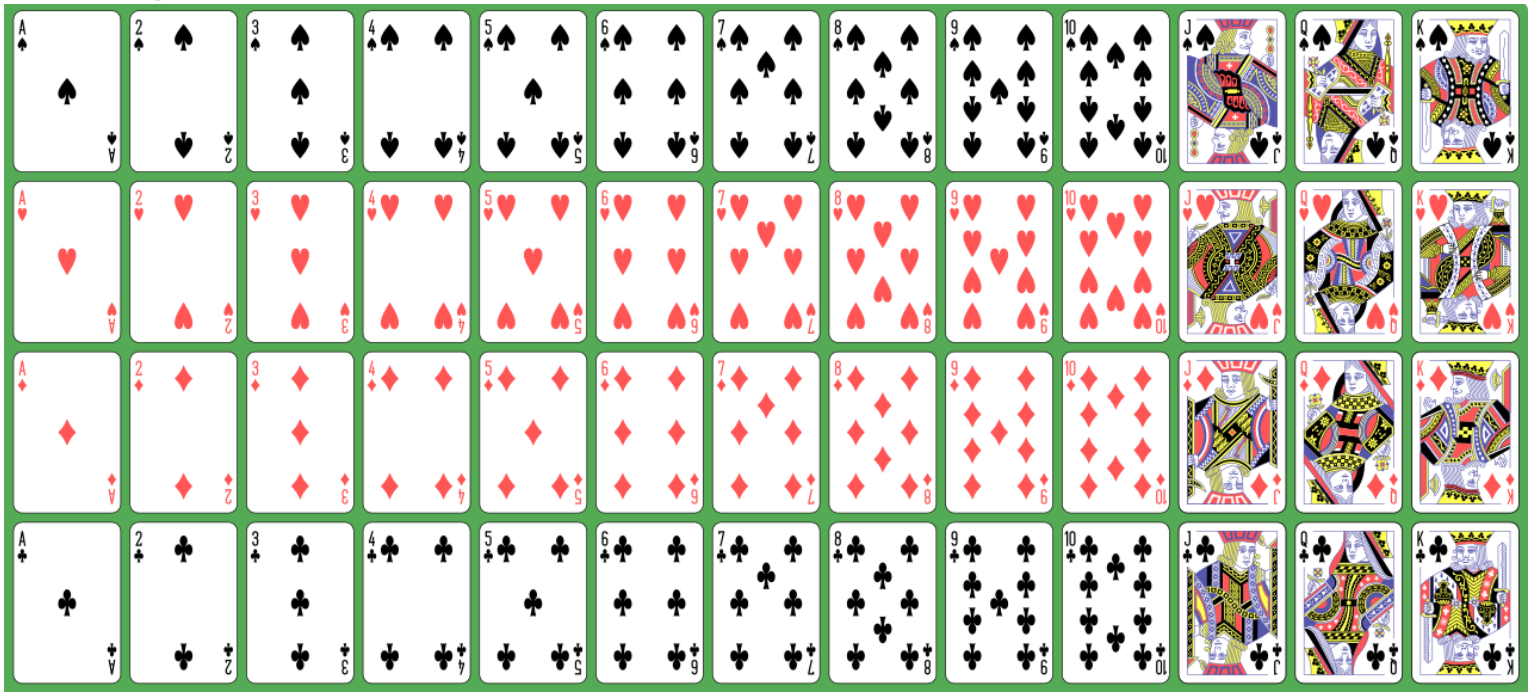
- **Question:** How to store your data?
- **Counter-question:** What kind of operations will be needed?
- **Answer:** Access the i-th element of a list.
- **Options:** unnumbered list (harder) vs. numbered list (easier)

- Ashoka
- Iltutmish
- Rajaraja Chola
- Kanishka
- Shah Jahan
- Harshavardhan

1. Ashoka
2. Iltutmish
3. Rajaraja Chola
4. Kanishka
5. Shah Jahan
6. Harshavardhan

Algorithms

- **Problem:** Given a set of 52 playing cards, arrange them in order.
- **Background:**

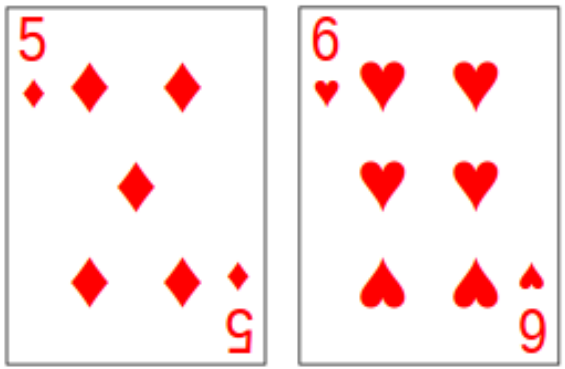


  - 4 suits: Spades, Hearts, Diamonds, Clubs
  - 13 values: 2, 3, . . ., 10, **J** (Jack), **Q** (Queen), **K** (King), **A** (Ace)

Algorithms

Strategy 1:

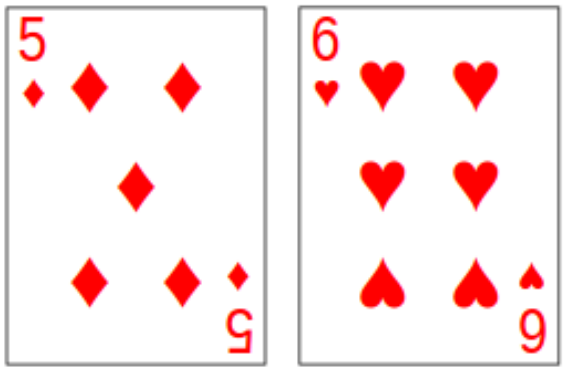
To be inserted



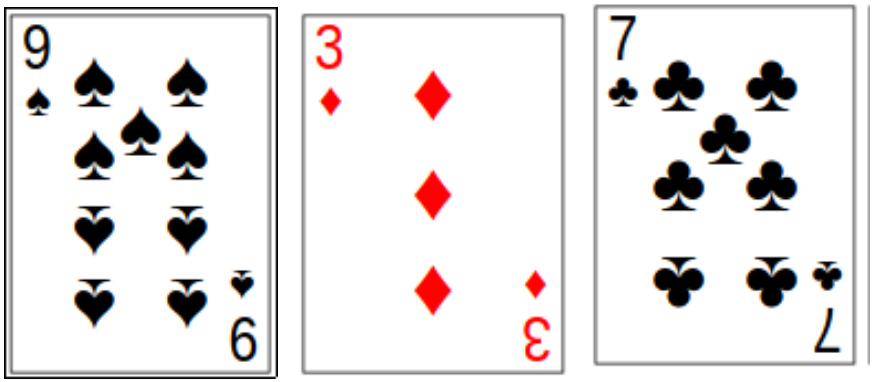
Algorithms

Strategy 1:

To be inserted



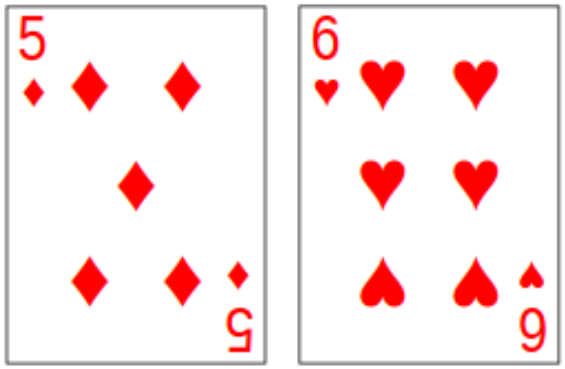
In hand



Algorithms

Strategy 2:

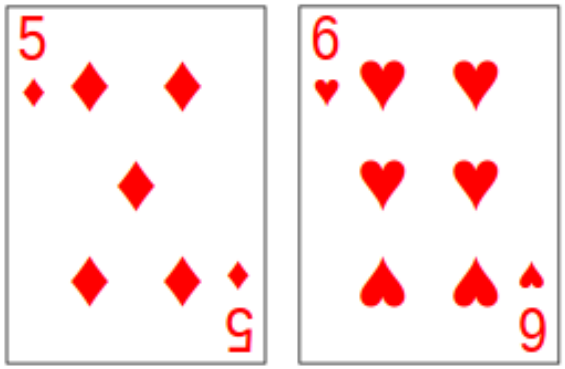
To be inserted



Algorithms

Strategy 2:

To be inserted



In data structure

Spades					Hearts			Diamonds				Clubs				
2	..	9	..	**A**	2	..	**A**	2	3	..	**A**	2	..	7	..	**A**

Algorithms

If you are not convinced that Strategy 2 is easier, try actually sorting (physically) a bunch of 52 pages, one per student, following the order of roll numbers.

Some SSH clients for Windows

- **MobaXterm**
  `https://mobaxterm.mobatek.net`
- **Solar-PuTTY**
  `https://www.solarwinds.com/free-tools/solar-putty`

CONNECTION: Connecting to ISI gateway server

1. Login to ISI's gateway server.
   - **host IP address:** 14.139.222.87, port 2222
   - **userid:** `mtcxxxx` (instead of `csxxxx` where `xxxx` is your 4-digit roll number)
   - **password:** same as your ISI Webmail/email password

Try running: `ssh mtcxxxx@14.139.222.87 -p 2222`

Note: You can change the default password with the command `yppasswd` or `passwd`.

CONNECTION: Connecting to ISI gateway server (contd.)

```
Welcome to Ubuntu 16.04 LTS (GNU/Linux 4.4.0-145-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

154 packages can be updated.
10 updates are security updates.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Mon Dec  7 13:17:51 2021 from 42.110.139.124
...
mtcxxxx@gateway:~$
```

↑ You have arrived at the gateway server

CONNECTION: Connecting to ISI computational server

2. Now run: `ssh csxxxx@192.168.64.35`

   Enter your ISI Webmail/email password when prompted.

CONNECTION: Connecting to ISI computational server

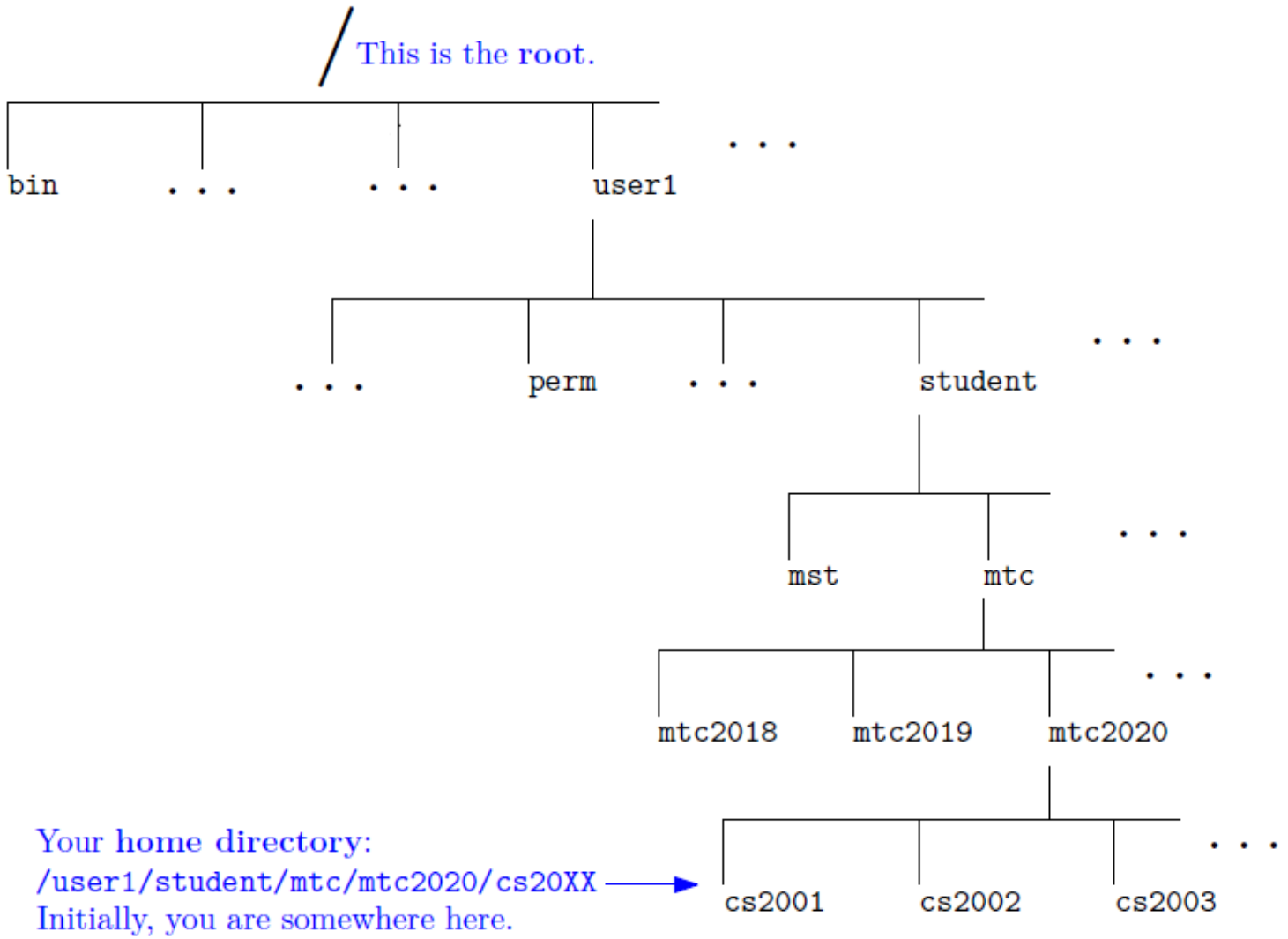
2. Now run: `ssh csxxxx@192.168.64.35`

   Enter your ISI Webmail/email password when prompted.

```
mtcxxxx@gateway:~$ ssh csxxxx@192.168.64.35
The authenticity of host '192.168.64.35 (192.168.64.35)' can't be established.
ECDSA key fingerprint is SHA256:ADmA7Y6bCVJYyVpF/mLI3F8nCM5cIKO+kANOahSjmOM.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts ...
csxxx@192.168.64.35's password:
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.4.0-141-generic x86_64)
...
Last login: Mon Dec  7 15:45:46 2021 from 192.168.143.29
csxxxx@student:~$
```

↑ You have arrived at the computational server

CONNECTION: You are here!!!



TRANSFER: Transferring files to the server

1. From Mobaxterm / Terminal *on your computer at home*, run

   ```
   ssh -L 1234:192.168.64.35:22 mtcxxxx@14.139.222.87
   -p 2222
   ```

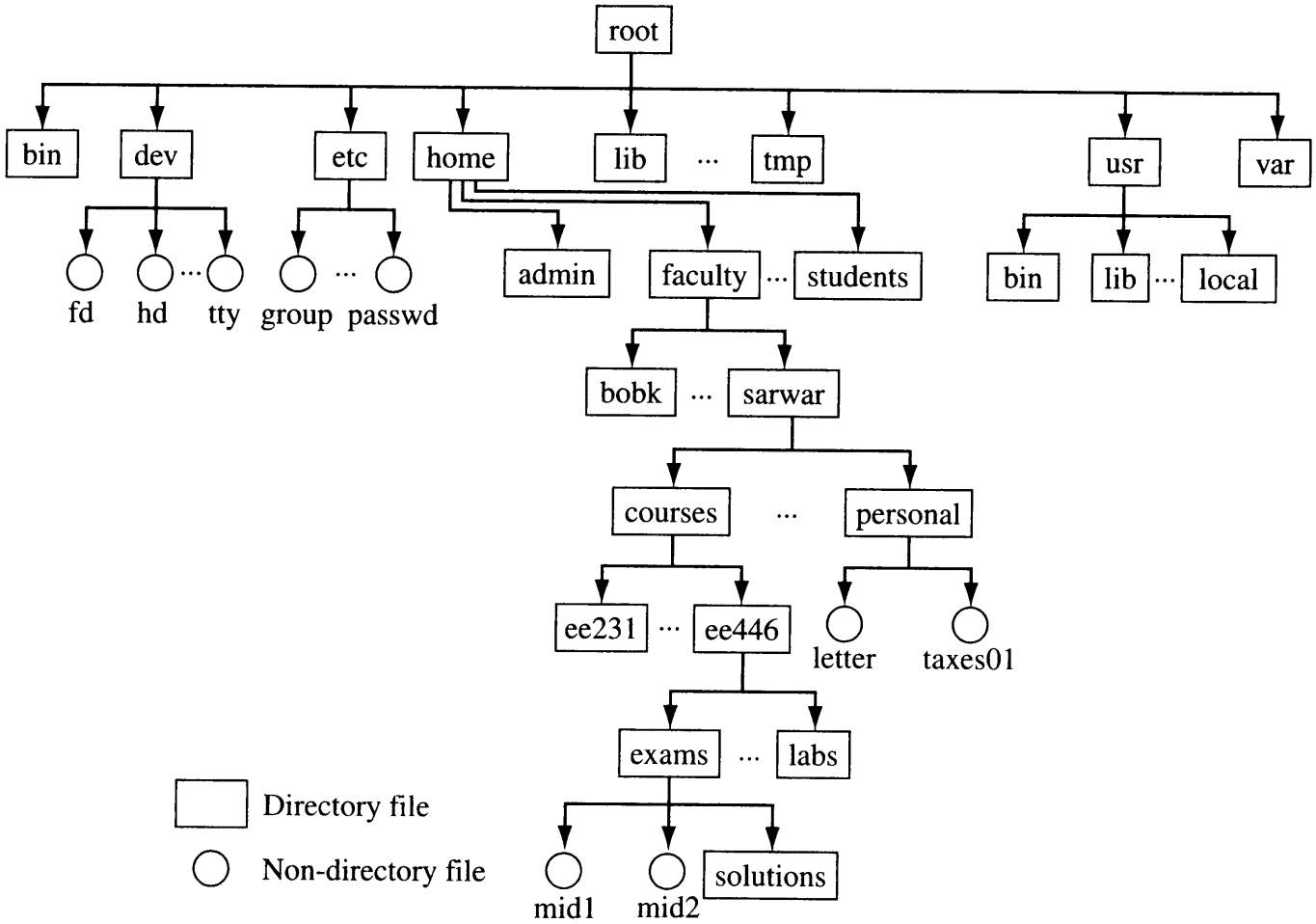
   Respond to the CAPTCHA and enter your email password when prompted. Keep this terminal open and go to next step.
2. Use the following information to connect via `filezilla` (`https://filezilla-project.org`) / `Tunnelier` / `Bitvise` or from command line:
   - **host:** `sftp://localhost`
   - **port:** 1234
   - **username, password:** `csxxxx`, and your email password

   Example:

   ```
   sftp -p 1234 csxxxx@localhost
   ```

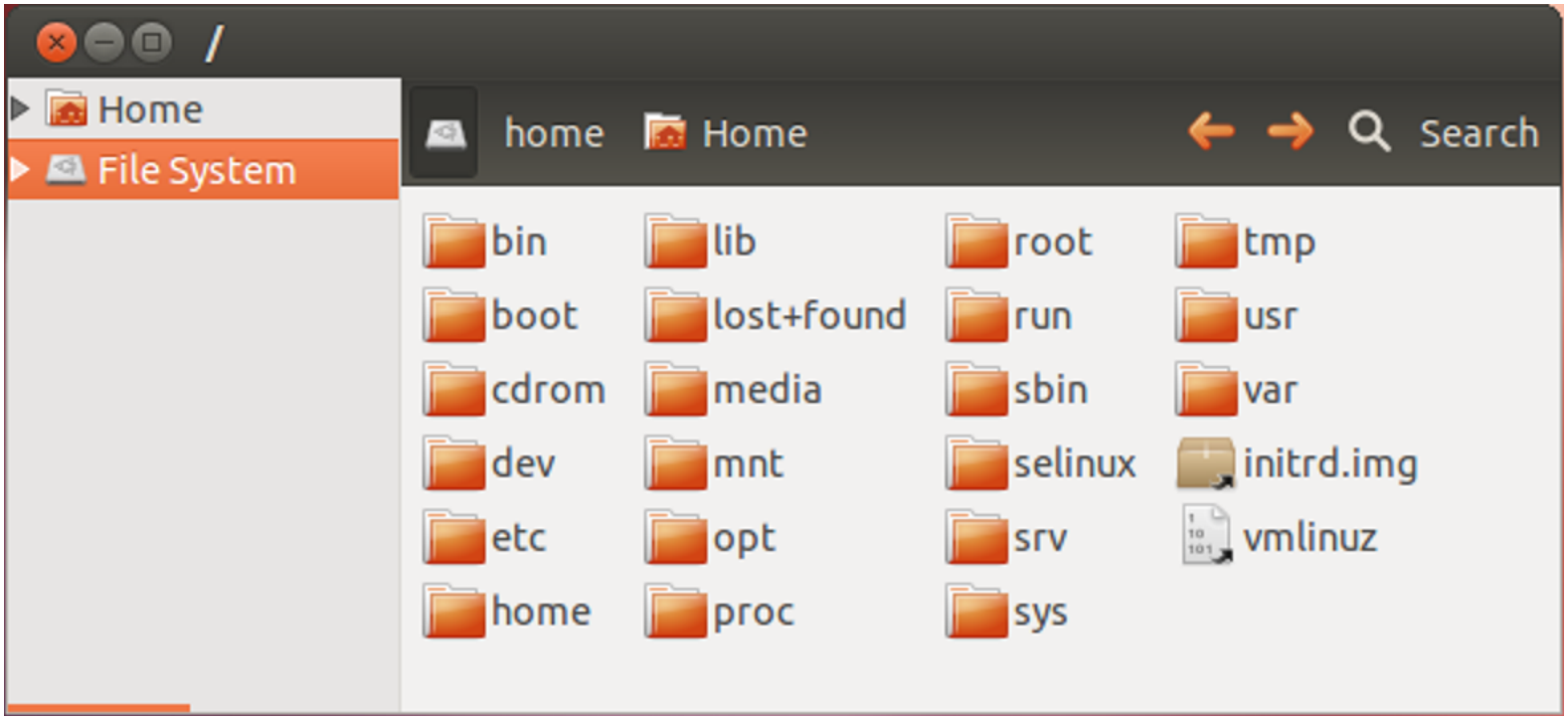
Note: This is an independent process. No need to set up CONNECTION prior to this separately.

Organization of Linux

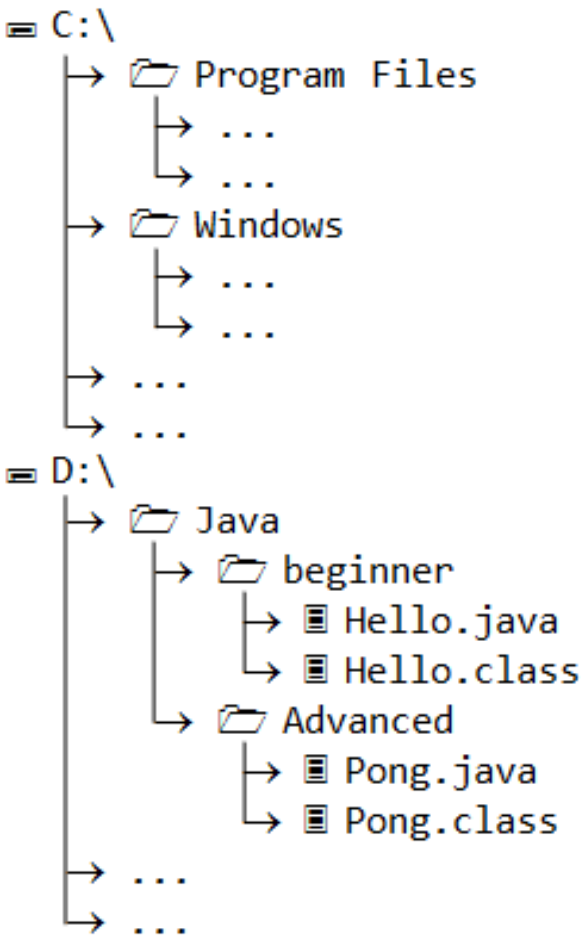


Hierarchical structure of directories, sub-directories, and files

Linux directories

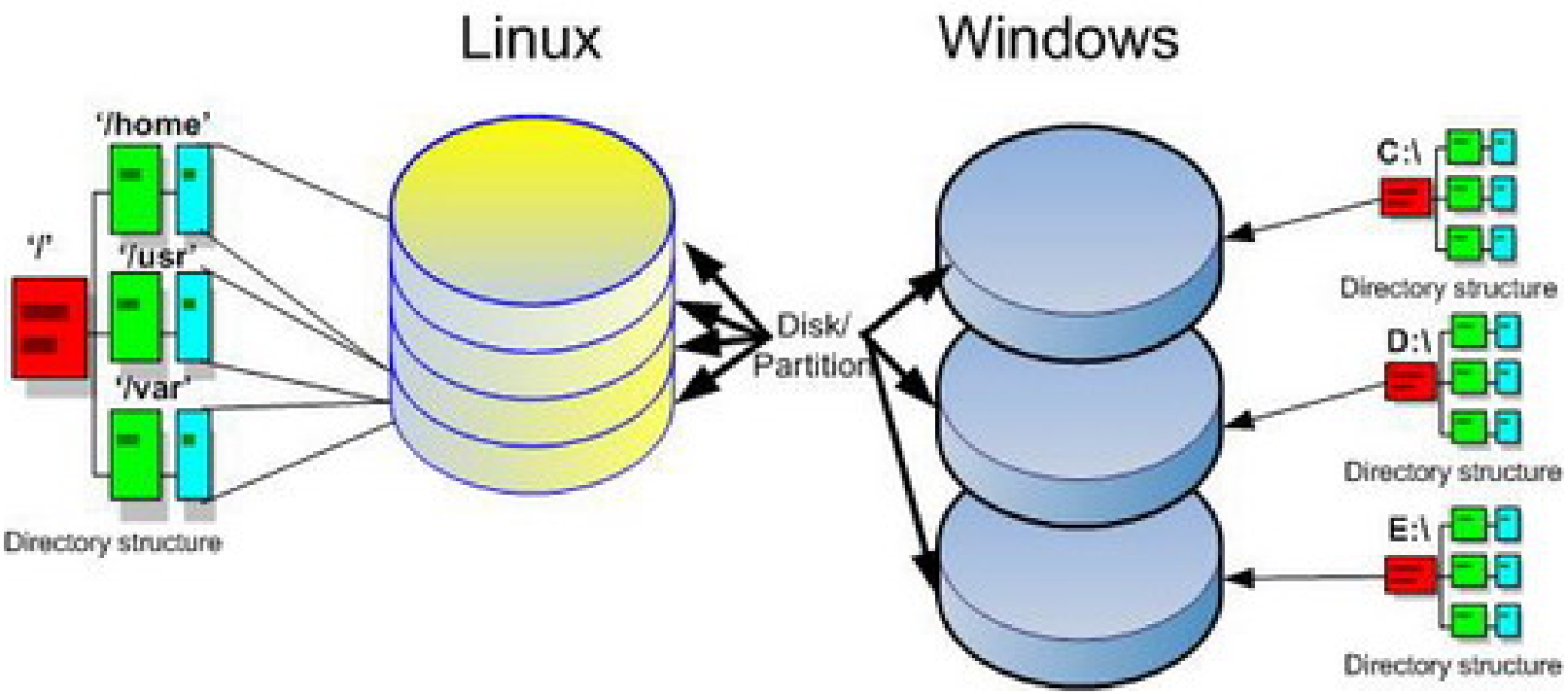


Organization of Windows



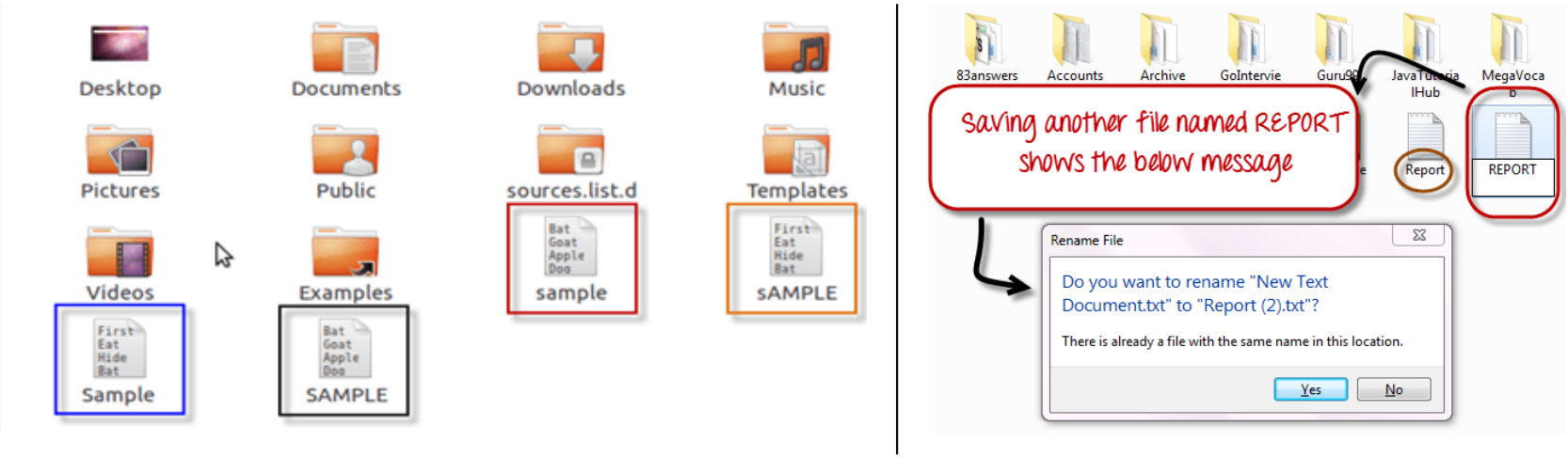
Hierarchical structure of drives, folders, sub-folders, and files

Linux versus Windows – Disk partitions



A view toward the disk partitions in Linux and Windows

Linux versus Windows – Naming conventions



File names are case sensitive in Linux but not in Windows

Terminologies

- Folders $\equiv$ *directories*
- Top of the hierarchy: *root directory* (/)
- Location of a file or directory: specified by *path*
- Current location in terminal or file browser: *current working directory*
- Normal (or default) start location: *home directory*
- Paths: *absolute* or *relative*
  - absolute path: from root (starts with /)
    Example: `/usr/bin/firefox`, `/tmp`, `/user1/student`
  - relative path: from current working directory (does not start with /)
    Example: `clab/assignment1/hello.c`

Terminologies

- Folders $\equiv$ *directories*
- Top of the hierarchy: *root directory* (/)
- Location of a file or directory: specified by *path*
- Current location in terminal or file browser: *current working directory*
- Normal (or default) start location: *home directory*
- Paths: *absolute* or *relative*
  - absolute path: from root (starts with /)
    Example: `/usr/bin/firefox`, `/tmp`, `/user1/student`
  - relative path: from current working directory (does not start with /)
    Example: `clab/assignment1/hello.c`

Note the difference between (forward) slash (/, used in Unix-like systems) and backslash (\, used in Windows-like systems).

Essential commands: directories

- `mkdir` : create a directory

  Examples:

  ```
  mkdir clab
  mkdir clab/assignment1
  ```

Essential commands: directories

- `mkdir` : create a directory

  Examples:

  ```
  mkdir clab
  mkdir clab/assignment1
  ```

  OR

  ```
  mkdir -p clab/assignment1 clab/assignment2
  ```

Essential commands: directories

- `mkdir` : create a directory

  Examples:

  ```
  mkdir clab
  mkdir clab/assignment1
  ```

  OR

  ```
  mkdir -p clab/assignment1 clab/assignment2
  ```

  ↑

  Create parent directories if required

Essential commands: directories

- `mkdir` : create a directory
  Examples:
  `mkdir clab`
  `mkdir clab/assignment1`

  OR

  `mkdir -p clab/assignment1 clab/assignment2`
  ↑
  Create parent directories if required
- `rmdir` : remove an (empty) directory
  Example: `rmdir assignment2`, `rmdir clab/programs`

Essential commands: navigating the file system

- `cd` : change directory
  Example:
  `cd /user1/student/mtc1999`
  `cd clab/assignment1/`
  `cd` ⟵ go to home directory
- `pwd` : print current working directory

Essential commands: navigating the file system

- `cd` : change directory
  Example:
  `cd /user1/student/mtc1999`
  `cd clab/assignment1/`
  `cd` ⟵ go to home directory
- `pwd` : print current working directory

Special directory names

- ~ : home directory
  Example: `cd ~/clab`
- `.` : current working directory
  Example: `./program1`
- `..` : parent directory (one level up)
  Example: `cd ..`, `cd ../assignment2`

Essential commands: file listing

- `ls` : view list of files in current directory
- `ls <path>` : view list of files in specified path
- `ls -l` : view detailed list of files
- `ls -lt` : view detailed list of files sorted by modification time

Essential commands: file listing

- `ls` : view list of files in current directory
- `ls <path>` : view list of files in specified path
- `ls -l` : view detailed list of files
- `ls -lt` : view detailed list of files sorted by modification time

Example:

```
$ /bin/ls -l
total 68
drwx------ 2 malay malay  4096 Sep 19 00:45 assignments
drwx------ 2 malay malay  4096 Sep 22  2016 exams
-rw-r--r-- 1 malay malay 13521 Sep 19 00:41 index.html
drwx------ 2 malay malay  4096 Sep 19 00:45 lectures
```

Essential commands: permissions

drwx------ 2 malay malay 4096 Jul 19 00:45 lectures

↑ Permissions

↑ Size

↑ Modification time

Essential commands: permissions

drwx------ 2 malay malay 4096 Jul 19 00:45 lectures

↑ Permissions ↑ Size ↑ Modification time

Permissions:

- 9 possible permissions:
  { read, write, execute } × { user (owner), group, other (everyone else) }
- 9 bits (1 ≡ permission granted)

ur	uw	ux	gr	gw	gx	or	ow	ox

Essential commands: permissions

drwx------ 2 malay malay 4096 Jul 19 00:45 lectures

↑ Permissions ↑ Size ↑ Modification time

Permissions:

- 9 possible permissions:
  { read, write, execute } × { user (owner), group, other (everyone else) }
- 9 bits (1 ≡ permission granted)

ur	uw	ux	gr	gw	gx	or	ow	ox

- `chmod`: changing permissions

  Example:

  `chmod g+wx <path>`
  `chmod og-wx <path>`
  `chmod 644 <path>` ⟵ 644 ≡ 110 100 100 ≡ `rw-r--r--`
  `chmod 700 <path>` ⟵ 700 ≡ 111 000 000 ≡ `rwx------`

Essential commands: files

- `cp` : copy a file
  Example:

```
cp program1.c program2.c
cp -i source-file target-file
cp -i source-file target-directory
```

(`-i` ≡ interactive, for confirmation)

Essential commands: files

- `cp` : copy a file
  Example:
  ```
  cp program1.c program2.c
  cp -i source-file target-file          (-i ≡ interactive,
  cp -i source-file target-directory      for confirmation)
  ```
- `mv` : rename (move) a file
  Example:
  ```
  mv program1.c program2.c
  mv -i source-file target-file          (-i ≡ interactive,
  mv -i source-file target-directory      for confirmation)
  ```

Essential commands: files

- `cp` : copy a file
  Example:
  ```
  cp program1.c program2.c
  cp -i source-file target-file          (-i ≡ interactive,
  cp -i source-file target-directory      for confirmation)
  ```
- `mv` : rename (move) a file
  Example:
  ```
  mv program1.c program2.c
  mv -i source-file target-file          (-i ≡ interactive,
  mv -i source-file target-directory      for confirmation)
  ```
- `rm` : remove (delete) a file
  Example:
  ```
  rm program1.c
  rm -i file1 file2.c *.bak
  rm -r some-directory (remove directory and everything inside it)
  ```

Viewers / pagers

- Useful for quickly viewing a file (not editing)
- Use `less`

  Example: `less cs19xx-day0-prog1.c`
  - space: move forward one page
  - backspace or b: move backward one page
  - q : exit the pager
  - / : search for a string in the file
  - run `man less` for more information

Editors/IDEs

Choose any one that you like. Some options:

- emacs, nano, vim
  - terminal based
  - can use them after logging in to the ISI server
- atom, geany, sublime text, vscode, etc.

Editors/IDEs

Choose any one that you like. Some options:

- emacs, nano, vim
  - terminal based
  - can use them after logging in to the ISI server
- atom, geany, sublime text, vscode, etc.

If you expect to do a substantial amount of programming, take the time and trouble to get familiar with a powerful editor / IDE. It will probably save you time and trouble in the long run.

Editors/IDEs

Choose any one that you like. Some options:

- emacs, nano, vim
  - terminal based
  - can use them after logging in to the ISI server
- atom, geany, sublime text, vscode, etc.

If you expect to do a substantial amount of programming, take the time and trouble to get familiar with a powerful editor / IDE. It will probably save you time and trouble in the long run.

Some random opinions / guides:

- `http://lifehacker.com/five-best-text-editors-1564907215`
- `http://www.techradar.com/news/the-best-free-text-editor-2017`
- `https://www.codementor.io/mattgoldspink/best-text-editor-atom-sublime-vim-visual-studio-code-du10872i7`
- `http://blog.liveedu.tv/10-best-text-editors-programming-2016/`

Other commands

- `man`
- `help [-dms]`
  Example: `man ls`, `man cp`, `help -d rm`

Other commands

- `man`
- `help [-dms]`
  Example: `man ls`, `man cp`, `help -d rm`

Find out more about these on your own.

- `alias` (giving your own, easy-to-remember names to commands)
- `wc` (counting characters, words, lines)
- `sort`
- `head`, `tail` (first few / last few lines)
- `cmp`, `diff` (comparing two files)
- `ps`, `top`, `kill` (checking what programs are running)
- `find` (finding files or directories)
- `grep` (searching for patterns)
- `awk`, `sed` (programming)

Useful references / cheat-sheets

https://phoenixnap.com/kb/linux-commands-cheat-sheet
http://www.math.utah.edu/lab/unix/unix-commands.html
https://www.cmu.edu/computing/services/comm-collab/collaboration/afs/how-to/unix-commands.pdf
https://ubuntudanmark.dk/filer/fwunixref.pdf
https://cs.brynmawr.edu/Courses/cs206/fall2015/Setup/UNIXCheatSheet.pdf
https://www.loggly.com/wp-content/uploads/2015/05/Linux-Cheat-Sheet-Sponsored-By-Loggly.pdf

Directory naming conventions

- Location

```
$ cd                    Go to your home directory.
$ mkdir -p clab/dayX    Create a directory for day X
$ cd clab/dayX          Go to the directory corresponding
                        to the class on day X.
```

File naming conventions

File names = roll number + assignment/test number + program number + "password"

- Assignments: `cs21xx-assignz-progy.c`
- Programming tests: `cs21xx-testz-progy.c`
- `xx` = your roll number
  `z` = assignment/test number (1, 2, 3, ...)
  `y` = program number (1, 2a, 2b, 3, ...)
- Insert a single alpha-numeric string of your choice, 6-8 characters long, in the name given above.
  Examples
  - `cs21XX-test3-abcdef-prog1.c`
  - `mnopqr-cs21XX-test3-prog2.c`
  - `cs21XX-test3-prog2-uvwxyz.c`

Submission header

Include following header at the beginning of **any** submitted program file (assignment / exam).

```
/*-------------------------------
Name:
Roll number:
Date:
Program description:
Acknowledgements:
-------------------------------*/
```

Compiling and running

- Compiling a C program
  ```
  gcc -g -Wall -o prog1 cs21xx-day1-prog1.c
  ```
- Running a C program
  ```
  ./prog1
  ```