CS6

Practical System Skills

Fall 2019 edition

Leonhard Spiegelberg
lspiegel@cs.brown.edu

22.98 Logistics

⇒ HW10 submission deadline extended to **next Tuesday, Dec 10th 4pm**

⇒ Final projects due 15th Dec

Last lecture: **today**

22.99 Recap DataFrames

⇒ DataFrames

- → can hold tabular-like data
- → used for small-medium sized datasets

⇒ quick manipulations, helpful for plotting, tables in Latex, and html tables in Flask

⇒ Data scientists' primary tool

23 Clusters

CS6 Practical System Skills

Fall 2019

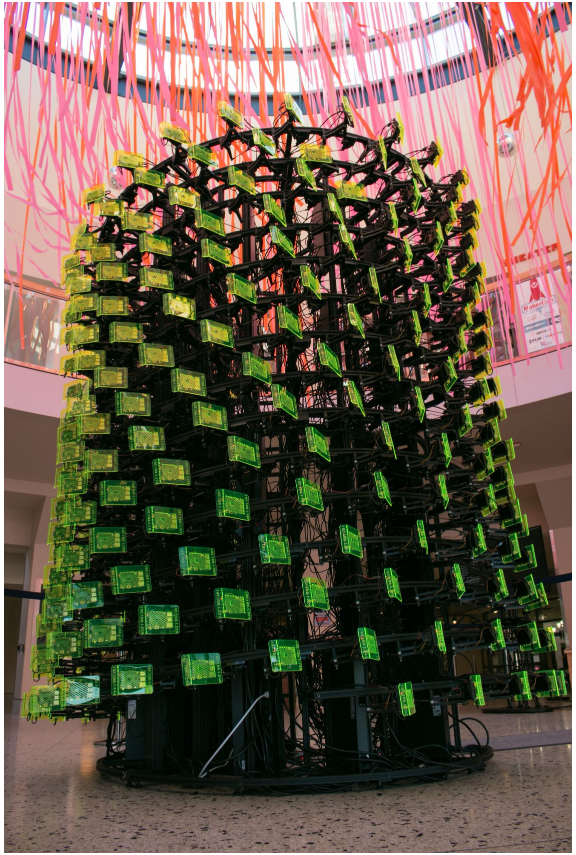
Leonhard Spiegelberg *lspiegel@cs.brown.edu*

23.01 What comes next?

So far:
Single machine,
multiple containers.

⇒ How about working with multiple, physical machines?

Virginia Tech's kinetic sculpture consisting of 256 Raspberry PIs

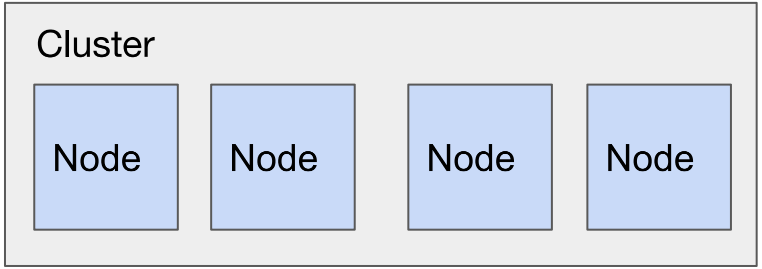


23.01 Clusters

What is a cluster?

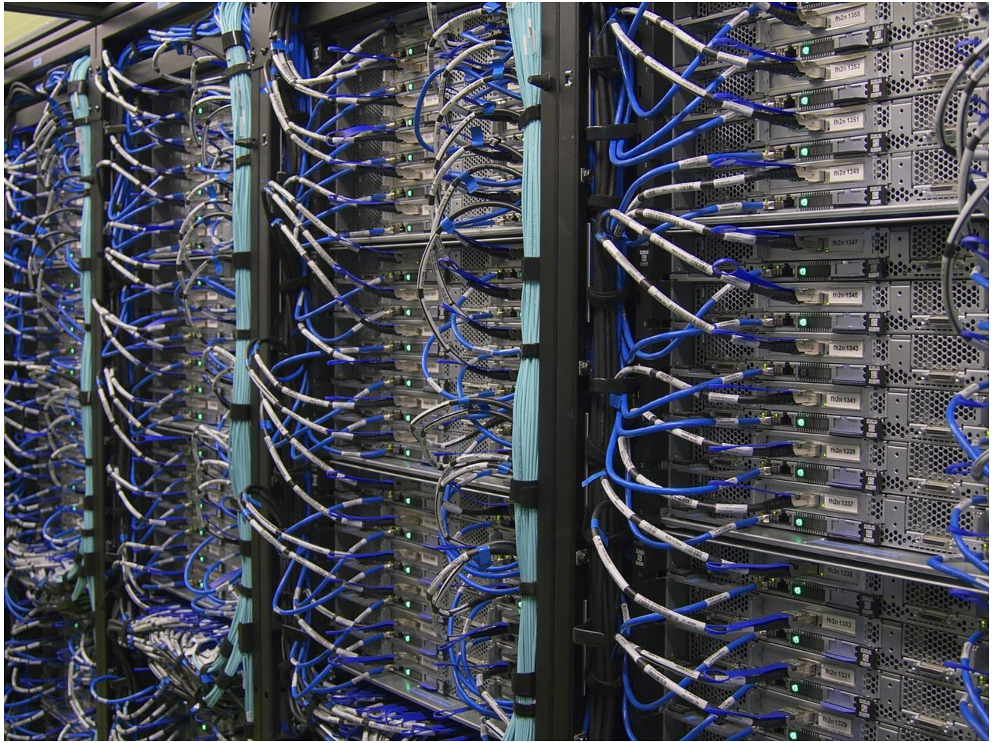
⇒ set of connected computers (servers), which can be viewed as a single system.

⇒ typically, a cluster is divided into nodes which do have several roles assigned.



Cluster

Node Node Node Node

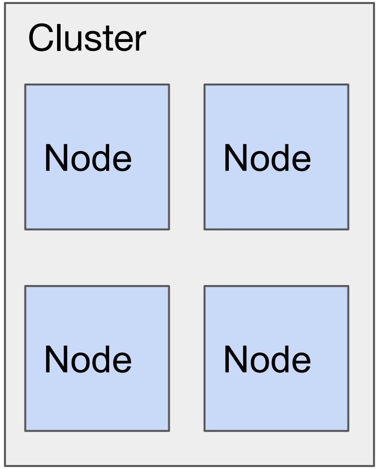


23.02 Nodes

Node = single physical machine

⇒ each node has one (or more) role(s) typically assigned, common are

1. login node
   → used to login to a cluster
2. master/manager node
   → used to coordinate a service, provides indirect access to workers
3. slave/worker node
   → executes actual work
4. data node
   → a node primarily concerned to store/provide data



23.03 Logging into a cluster

⇒ in order to protect a cluster, usually one or more machines are designated to be login nodes

→ e.g. `ssh.cs.brown.edu`

⇒ other machines are not reachable from internet, but merely from login node (→ SSH agent-forwarding)

⇒ development and testing should happen on login node

→ Don't store large files there, do not run production code on a login node

23.04 Running things on a cluster

⇒ To run a program/application over multiple nodes, you typically package it into a job
→ many frameworks do that automatically for you

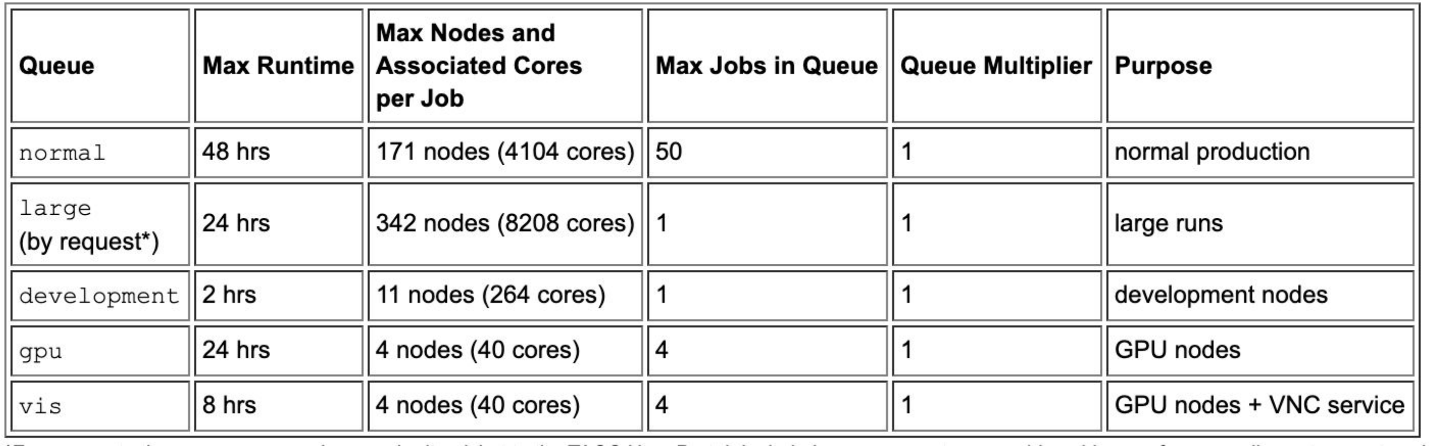
⇒ as a user you submit your Job to a queue, a scheduler then assigns resources and executes your job eventually.

⇒ Popular schedulers are:
SLURM (academia/science)
Mesos, YARN or Kubernetes (industry)

23.05 Queue / Jobs example - science/academia

⇒ TACC is a cluster from the U of Texas (https://portal.tacc.utexas.edu/user-guides/lonestar5#running-queues)

⇒ Brown also has a cluster, oscar https://docs.ccv.brown.edu/oscar/!



Queue	Max Runtime	Max Nodes and Associated Cores per Job	Max Jobs in Queue	Queue Multiplier	Purpose
`normal`	48 hrs	171 nodes (4104 cores)	50	1	normal production
`large` (by request*)	24 hrs	342 nodes (8208 cores)	1	1	large runs
`development`	2 hrs	11 nodes (264 cores)	1	1	development nodes
`gpu`	24 hrs	4 nodes (40 cores)	4	1	GPU nodes
`vis`	8 hrs	4 nodes (40 cores)	4	1	GPU nodes + VNC service

⇒ To submit a job, you write a bash-like SLURM script and submit it via

```
sbatch script.sh
```

23.06 Queue / Jobs example - industry

⇒ Whereas scientists write typically SLURM scripts and explicitly submit jobs, schedulers used in industry are usually integrated with frameworks for more convenience.

Example:

```
spark-submit --master yarn --deploy-mode cluster
             --queue production
             ingest-job.jar conf.yml
```

Spark is a popular big data framework

23.07 Practical tips when working on a cluster

⇒ typically you don't have admin rights, i.e. can't install additional software

- → to solve this, use "user" mode, i.e. install software in some directory, setup paths, zip dependencies and ship them
- → many users use `$HOME/.local`
- → `pip3 install <package> --user` to install to `.local`
- → `./configure --prefix=$HOME/.local` for software requiring a local build/compilation

Software for clusters

23.08 Distributed file storage

⇒ allow to store (large) files distributed to get several benefits

→ faster reads/writes when chunked/partitioned

→ fault-tolerance through replication

→ store more data

There are several kinds of distributed file storage, popular are:

1. Object stores, e.g. Amazon S3
2. Distributed file systems, e.g. Ceph or HDFS (Hadoop FileSystem)

⇒ In production scenarios you'll typically work with a distributed system

23.09 Compute frameworks

⇒ Can use a distributed database, ingest data into it and perform analytics

→ Popular solutions are Vertica, OmniSci(MapD), …

⇒ Sometimes you just want to compute over input files, no need for a database. Distributed programming frameworks provide this functionality:

- Science: MPI
- Industry: Spark, Hadoop MapReduce, Flink, Storm, Presto, ...

23.10 WimPi

⇒ It's a *NIX world

⇒ Research project for next-gen system on a Raspberry PI cluster

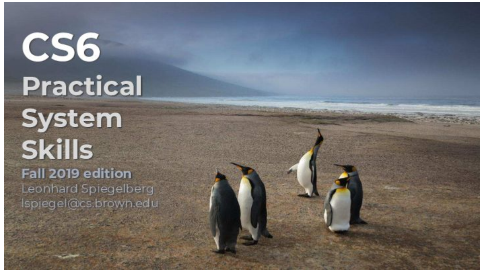
⇒ 25 nodes

End of new content.

Course recap - what did we learn?

23.11 Week 1

- ⇒ Working with a CLI, REPL style
- ⇒ File paths: `/absolute` and `../relative/..`
- ⇒ Navigating the file system in a shell (`cd, ls, pwd`)
- ⇒ Working with files (`mv, cp, rm, cat, hexdump`)
- ⇒ Wildcard patterns (`ls otp_fl?ight_*.csv`)
- ⇒ Brace expansion (`mv *.{csv,json} folder/`)



CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.12 Week 2

⇒ user permissions (`chmod g+x,u=rw,o= file.txt`)

⇒ Links (`ln -s target link_name`)

⇒ Streams and Pipes (`cmd1 | cmd2`, redirection e.g. `cmd > out.txt`)

⇒ Stdin(0), Stdout(1), Stderr(2)

⇒ Stream redirection (`cmd > out.txt 2>&1`, `cmd 2>&1 | tee out.txt`)

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.13 Week 3

⇒ Bash scripting

⇒ Shell variables, environment variables

⇒ Passing parameters to scripts
(stdin, parameters, environment, read)

⇒ Arithmetic expansion `(( x *= 7 ))`

⇒ Quoting (Difference between ', " and `)

⇒ command expansion via `` `cmd` `` or `$(cmd)`

⇒ return/status codes, `&&` and `II`

⇒ control flow via `if`

⇒ tests, i.e. `[[ … ]]`, `[ … ]`, and `test`

⇒ arrays and dictionaries (`ARR=(1 2 3)` or `declare -a d`)

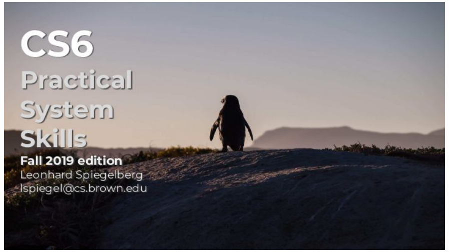
CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.14 Week 4

- ⇒ SSH
- ⇒ hostnames, URLs, URIs
- ⇒ Practical public key cryptography via SSH keys
- ⇒ SSH config (`~/.ssh/config`, `~/.ssh/known_hosts`, `~/.ssh/authorized_keys`)
- ⇒ `scp` and `rsync`
- ⇒ Tape archives (`tar`)
- ⇒ Processes (`ps`, `kill`, `fg`, `bg`) and Signals (`Ctrl + C`, `Ctrl + \`, ...)

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.15 Week 5

⇒ String processing (`wc`, `uniq`, `sort`, `tr`)

⇒ CSV files (`cut`, `paste`)

⇒ process substitution (`<(echo "Hello world")`)

⇒ `diff`

⇒ `xargs`

⇒ Regular expressions, `grep`

⇒ `sed` and `awk`

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.16 Week 6

- ⇒ HTML (`<html> … </html>`)
- ⇒ HTTP requests (GET/POST/...)
- ⇒ Using cURL to issue HTTP requests
- ⇒ CSS

CS6

Practical System Skills

Fall 2019 edition

Leonhard Spiegelberg

lspiegel@cs.brown.edu

CS0060

Philipp Eichmann

HTML/CSS/JS

23.17 Week 7

⇒ Git, version control

⇒ Git areas (working dir, staging area, repository)

⇒ creating commits, pushing them to a remote

⇒ Checking out old versions, detached HEAD

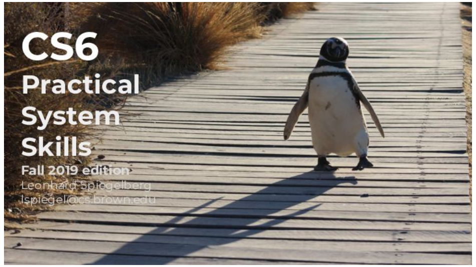
⇒ Branching and Pull requests

⇒ merge conflicts

⇒ rebasing vs. merging

⇒ Git workflows

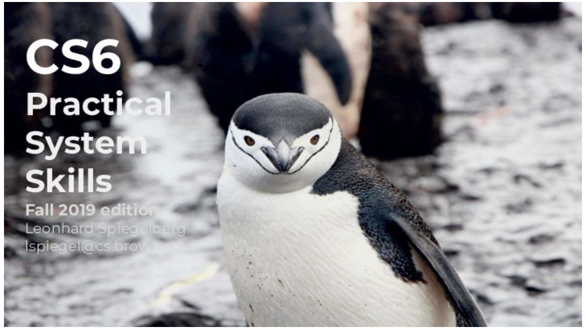
CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.18 Week 8

⇒ Python



CS6

Practical

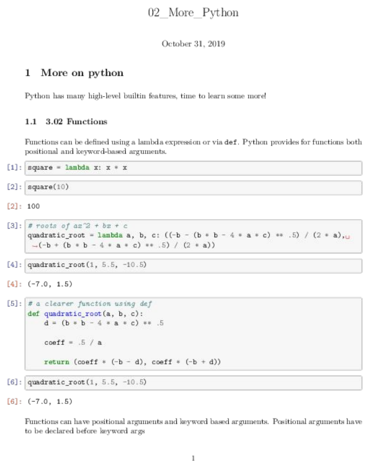
System

Skills

Fall 2019 edition

Leonhard Spiegelberg

lspiegel@cs.brov



02_More_Python

October 31, 2019

1 More on python

Python has many high-level builtin features, time to learn some more!

1.1 3.02 Functions

Functions can be defined using a lambda expression or via **def**. Python provides for functions both positional and keyword-based arguments.

```
[1]: square = lambda x: x * x
```

```
[2]: square(10)
```

```
[2]: 100
```

```
[3]: # roots of ax^2 + bx + c
     quadratic_root = lambda a, b, c: ((-b - (b * b - 4 * a * c) ** .5) / (2 * a),␣
      ↪(-b + (b * b - 4 * a * c) ** .5) / (2 * a))
```

```
[4]: quadratic_root(1, 5.5, -10.5)
```

```
[4]: (-7.0, 1.5)
```

```
[5]: # a clearer function using def
     def quadratic_root(a, b, c):
         d = (b * b - 4 * a * c) ** .5

         coeff = .5 / a

         return (coeff * (-b - d), coeff * (-b + d))
```

```
[6]: quadratic_root(1, 5.5, -10.5)
```

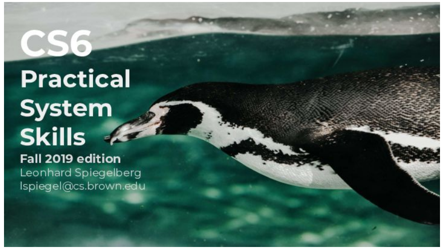
```
[6]: (-7.0, 1.5)
```

Functions can have positional arguments and keyword based arguments. Positional arguments have to be declared before keyword args

23.19 Week 9

- ⇒ Flask, developing a web backend using python
- ⇒ dynamic vs. static websites
- ⇒ routes(`/blog/<int:year>/<int:month>`) and requests
- ⇒ Templating using Jinja2
- ⇒ HTML forms
- ⇒ Javascript / JSON / REST

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



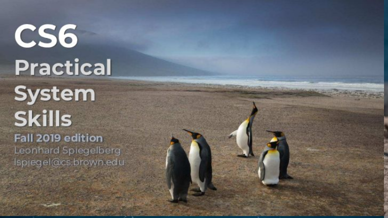
CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

23.20 Week 10

- ⇒ Databases
- ⇒ relational databases (Postgres)
- ⇒ Document stores (MongoDB)
- ⇒ `SELECT, INSERT, CREATE TABLE, UPDATE, DELETE, …`
- ⇒ Transactions
- ⇒ Aggregations

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical
System
Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



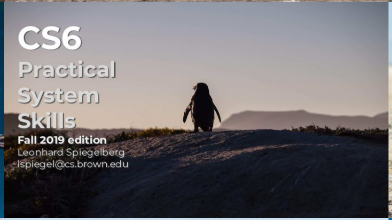
CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

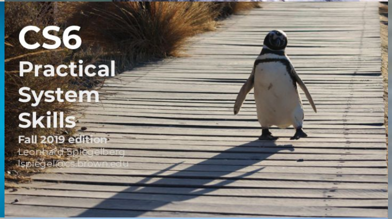
CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

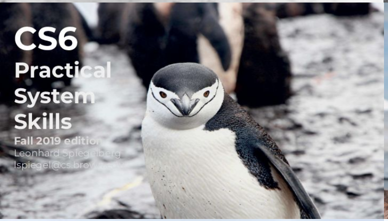
CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

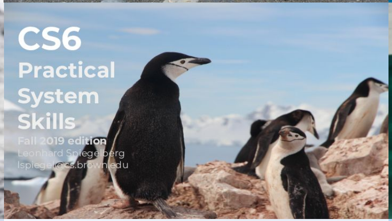
CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



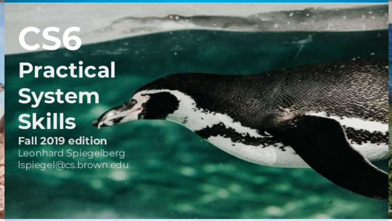
CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu



CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

CS6
Practical System Skills
Fall 2019 edition
Leonhard Spiegelberg
lspiegel@cs.brown.edu

What comes next?

23.21 Life after CS6

Courses for Spring 2019/2020, if you liked…

- … UNIX/programming/systems ⇒ **CS131: Fundamentals of Computer Systems**
- … Databases ⇒ **CS127: DB Management Systems**
- … DataFrames/Analytics ⇒ **CS1951A: Data Science**
- … Websites ⇒ **CS132: Creating Modern Web Applications**
- … Regular expressions ⇒ **CS101: Theory of Computation**
- … Programming/Javascript ⇒ **CS32: Intro to SE**

TAing

Research

Internships

Build cool stuff!

End of lectures.

Final Projects: Sun 15th Dec, 3-5pm