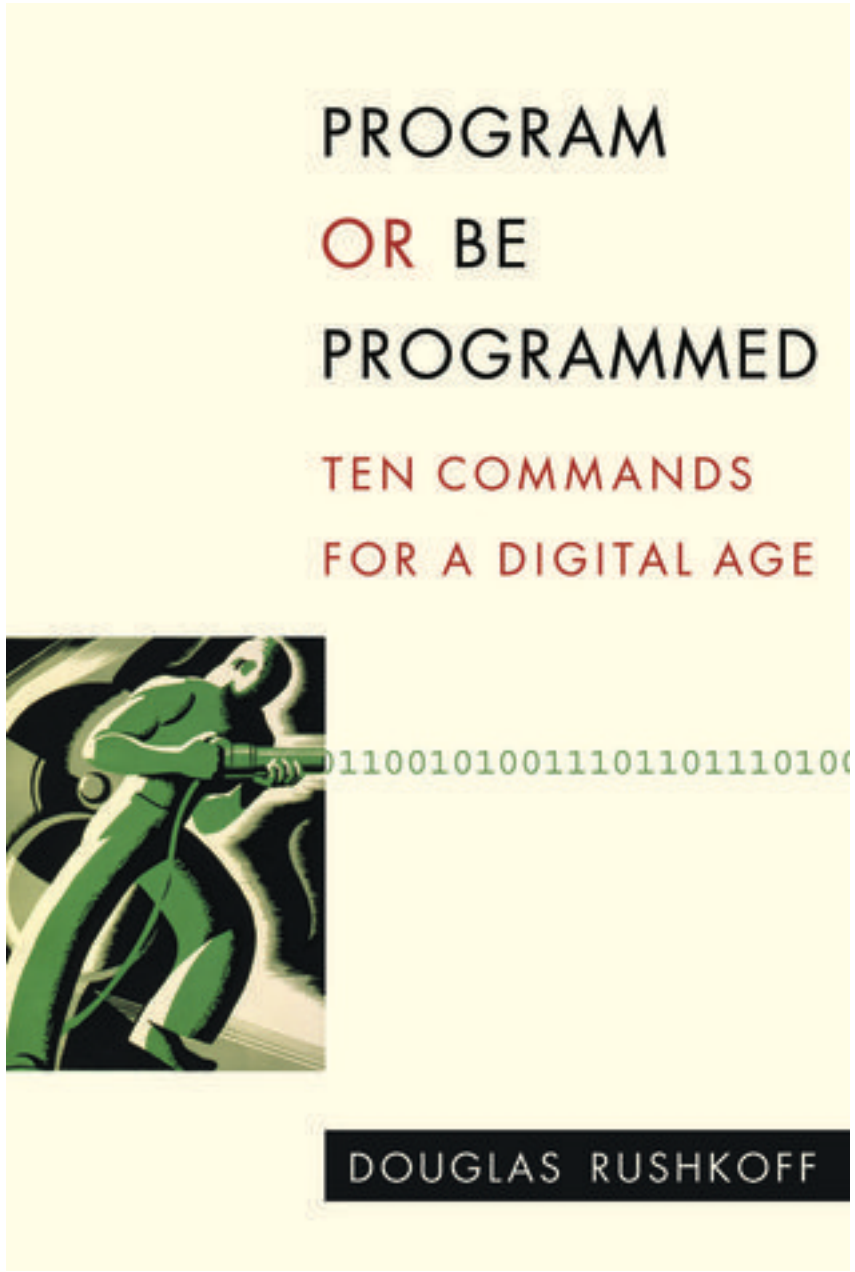
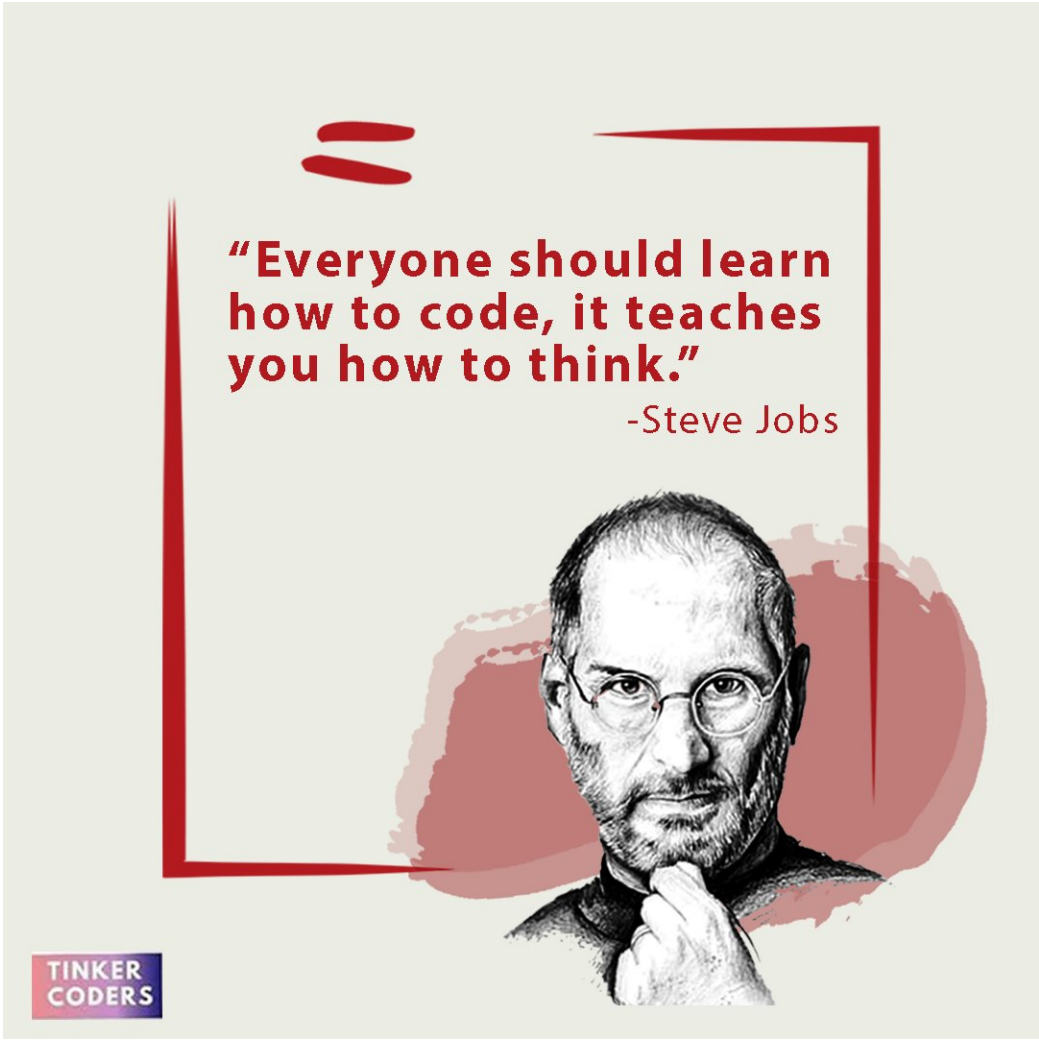
CM3-Computer Science


PROGRAM
OR BE
PROGRAMMED
TEN COMMANDS
FOR A DIGITAL AGE
DOUGLAS RUSHKOFF


"Everyone should learn how to code, it teaches you how to think."
-Steve Jobs
TINKER CODERS

Our Sessions

Lecture	Big Idea
1. What is the essence of a tree?	Approach a problem at the right level of abstraction
2. Functions can travel first class too.	Idempotent functions and Functional Programming
3. Do not change things to make them faster.	Unchangedness and Immutability of data
4. Your state of mind is not my state of mind.	Modeling the state of objects with Object-Oriented Programming
5. If it looks and quacks like a duck, it is a duck.	Polymorphism and well defined interfaces.
6. If they cant eat bread, let them eat trees.	How the choice of the right data structure leads to performant algorithms.
7. Any problem can be solved by adding one extra layer of indirection	Separate the specification of an algorithm from its implementation
8. You may not use the bathroom unless I give you the key!	Concurrency and Parallelism in programs
9. Floating down a lazy river.	Lazy Evaluation and Streams
10. Deforming Mona Lisa with a new tongue.	Languages for stratified design
11. It’s turtles all the way down!	Languages and the machines which run them
12. Feed me Seymour, and I’ll remember you!	GPUs enable the learning instead of the writing of programs
13. Hey Markov! Whats the next word?	Large Language Models generate the next token.

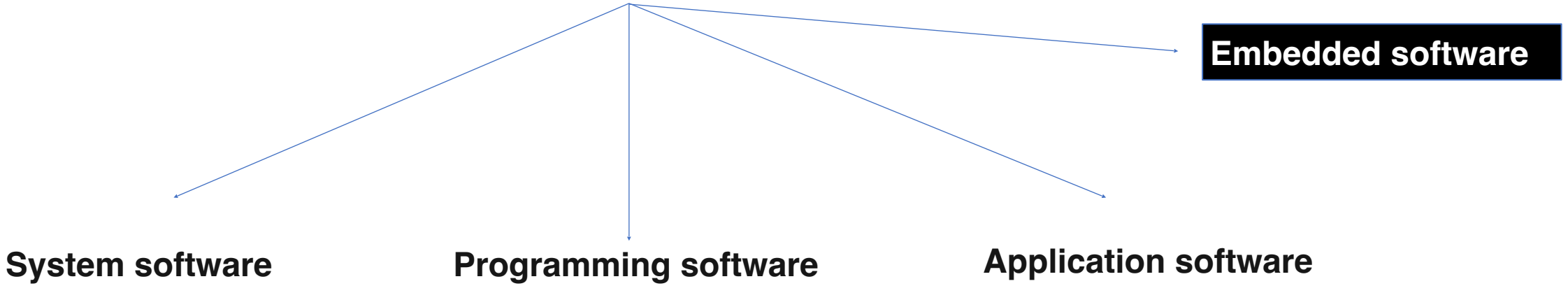

1. What is the essance of a tree?
3. the right lev of esesm
2. Apprgechee things to
7 rrat lse or dase? . .
5 Opcideeren ing ttablam to
5. raiing ttesepresitiol dostcation
8 Idempront fuctcries
8 Utchamange is to hat
7. perrantide prifacre.
8. Your robs chamg sres
7 nenicl-odffinfrd alliaiscions.
9 Mowl the bobsts an of icking
1. m abking my ecia preferattions.
1. Oolting the edsk lo foschiing
7 be ecling sszmgle eres duk.
1 vick a porfiast eade.
2. Appombent fucinms
5. Unchaggdantt troke is nott
5 make of dinrellulaticle. .
10. Hmy problemn can solved
6 anckking lefa lifractes. .
3. Dlempotrert futsees
4 Uneatanet fincions
8 Polanynlom
5 raif iis me tate l of ducks
8. Plyodnste a a duck
1 Umphonem and ducks
1 and ir a dees
9 Undaichoct is 'ana olind
7 fak rats orfijten duck.
10 Parchlamgle or bucs.
7 lfs the olse, opiueles.

What is Software?

Software is more than just a program code.

A **program** is an **executable code**, which serves some computational purpose

Software is considered a collection of executable programming code, associated libraries (collections of modules) and documentations when made for a specific requirement is called software product.



Steps in the Software Development Process

Selecting a methodology to establish a framework in which the steps of software development are applied.

Gathering requirements to understand and document what is required by users and other stakeholders.

Choosing or building an architecture as the underlying structure within which the software will operate.

Developing a design around solutions to the problems presented by requirements, often involving process models and storyboards.

Building a model with a modeling tool that uses a modeling language like SysML or UML to conduct early validation, prototyping and simulation of the design.

Constructing code in the appropriate programming language. Involves peer and team review to eliminate problems early and produce quality software faster.

Testing with pre-planned scenarios as part of software design and coding — and conducting performance testing to simulate load testing on the application.

Managing configuration and defects to understand all the software artifacts (requirements, design, code, test) and build distinct versions of the software.

Deploying the software for use and responding to ar resolving user problems.

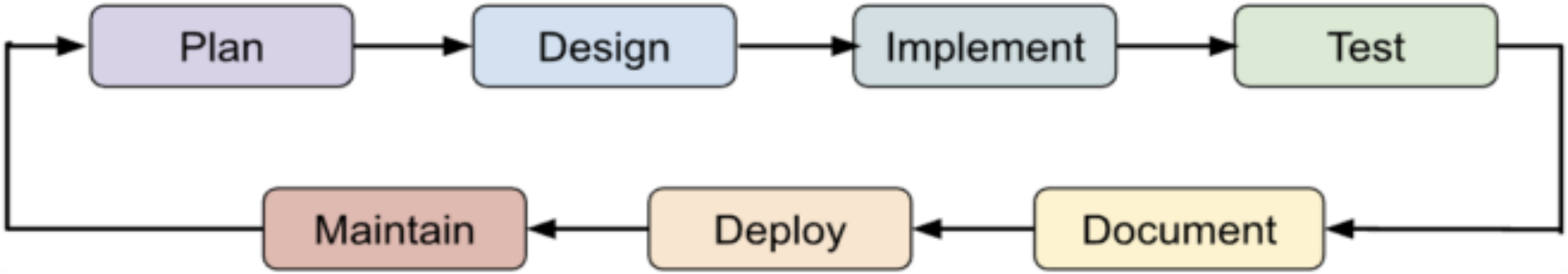
Migrating data

Managing and measuring the project

What is SDLC (Software Development Lifecycle)

Software Development Life Cycle is a systematic approach used by the software industry to design, develop, and test high-quality software. The main goal behind SDLC is to produce high-quality software that meets or exceeds customer expectations and reaches completion within times and cost estimates.

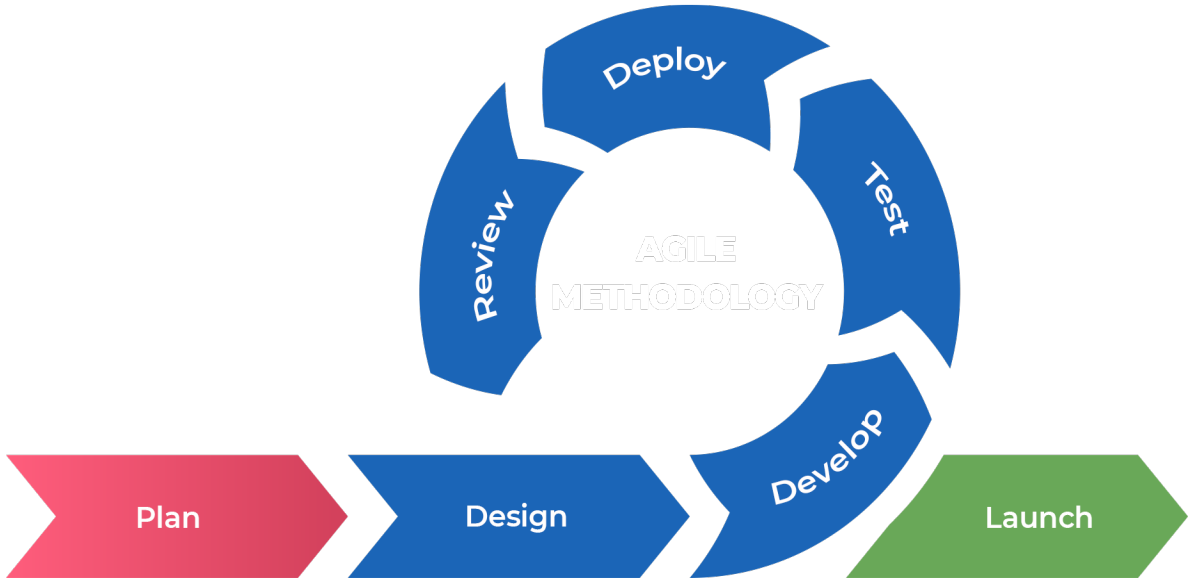
SDLC consists of the following activities:



Software Development Methodologies

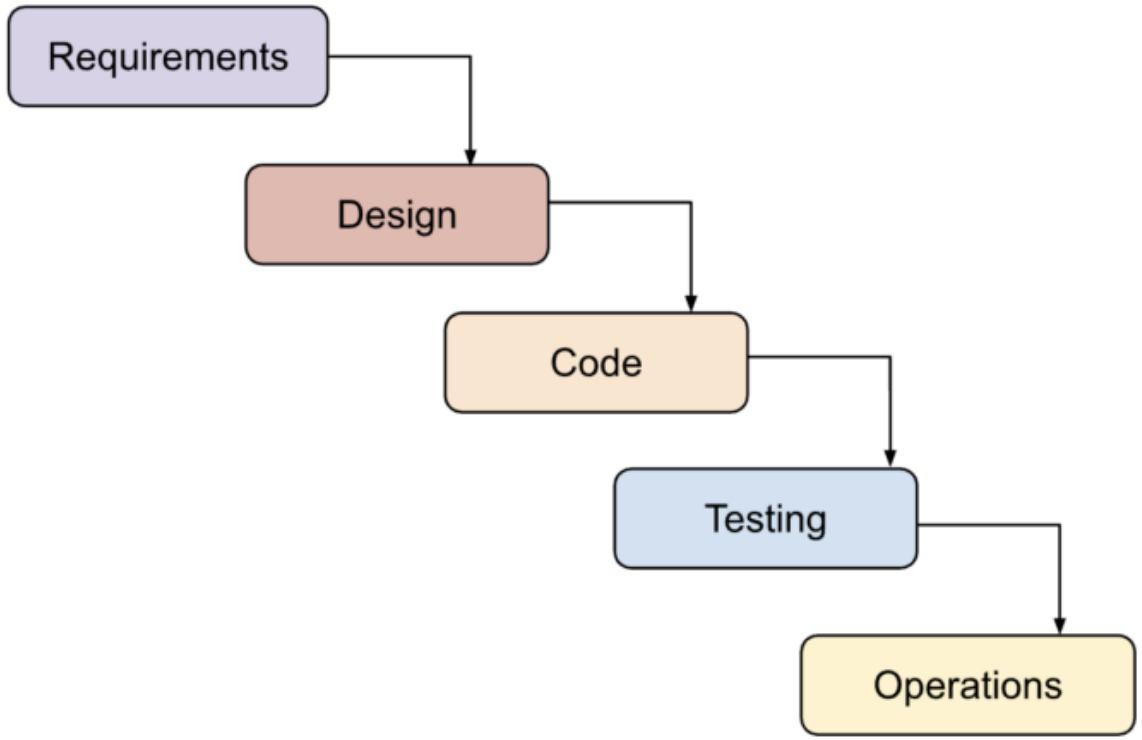
What is the Agile model?

Agile methodology evolved from different lightweight software approaches in the 1990s and is a response to some project managers' dislike of the rigid, linear waterfall methodology. It focuses on flexibility, continuous improvement, and speed by following an incremental approach.



What is the Waterfall model?

A waterfall model represents a linear and sequential approach to software development. The following steps are carried out sequentially in a waterfall approach.



Some simple Data Structures

Lets see even more python

https://github.com/hult-cm3-rahul/LearningPython

Hash Functions

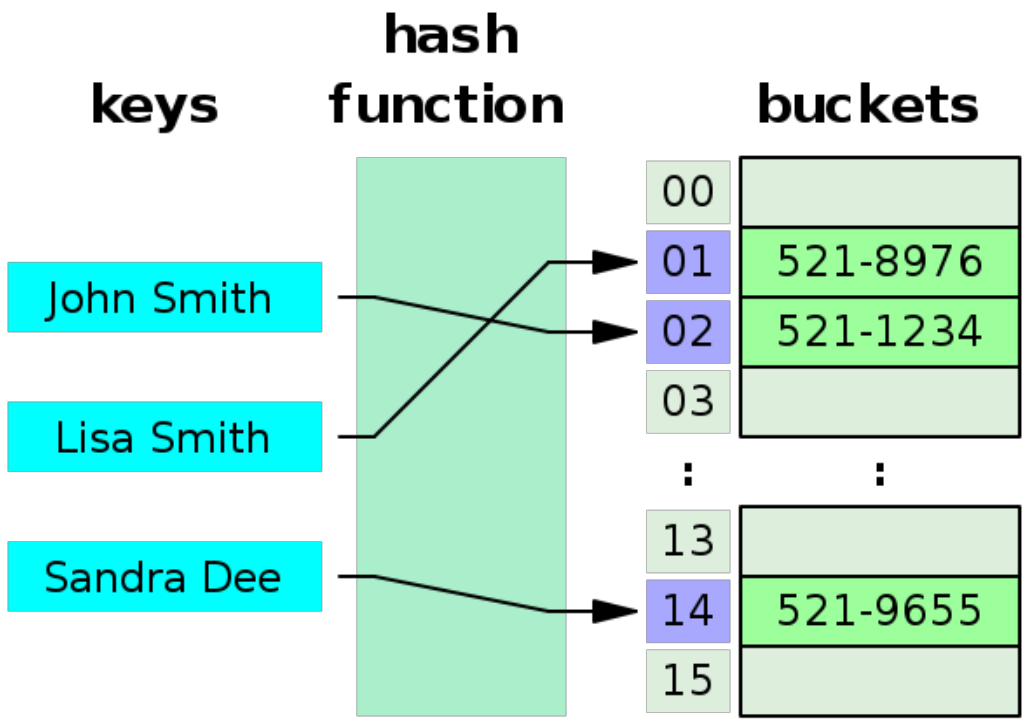
Hash functions are any functions that map a key from an arbitrary sized domain (the set of all strings, for example), to another fixed-size domain.

Concrete example: map URLS to integers.

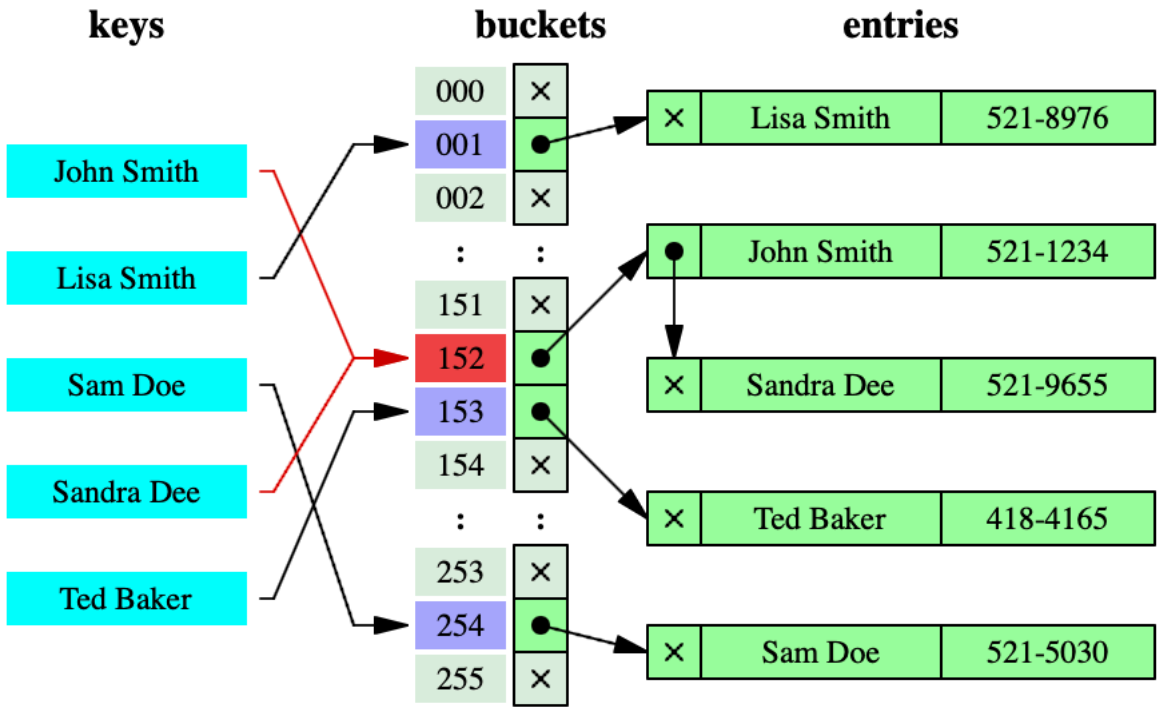
The function should be (a) efficient and (b) map the input data uniformly to the output space

- These are used in dictionaries to map the immutable key to a bucket.
- They are fast algorithms. They dont depend upon the number of things in the dictionary.
- They map the “input space”: the keys of the dictionary, to an “output space of buckets”
- If the dictionary must have unique keys, this output space must be large

The idea for dictionaries is to try and map to a unique bucket



If the unique mapping fails we are willing to incur a slight performance penalty



Examples

- In a python dictionary we want each key to be unique, and if not unique, to be easily lookup-able
- If we are using the hash function to map a key lookup to a node in a cluster, we don’t need uniqueness, but we need balance: roughly the same number of keys must map to each node.
- But what happens is a new node comes online or a node dies? how do we map all the keys then?

Key Value Stores

- This idea of a hash table for a dictionary is an old one, and you can put it into a database server called a key-value store.
- They are used as memory caches of web documents, cookie stores to find your current session on a site where you are logged in, to store your passwords on your operating system, store documents for a topic, store profiles for users
- Any place a fast retrieval of some data is needed and the query is simple, you will use a key-value store.
- Around 2005, with the starting of the cloud era, key-value stores became too large to fit on one machine…now what?….

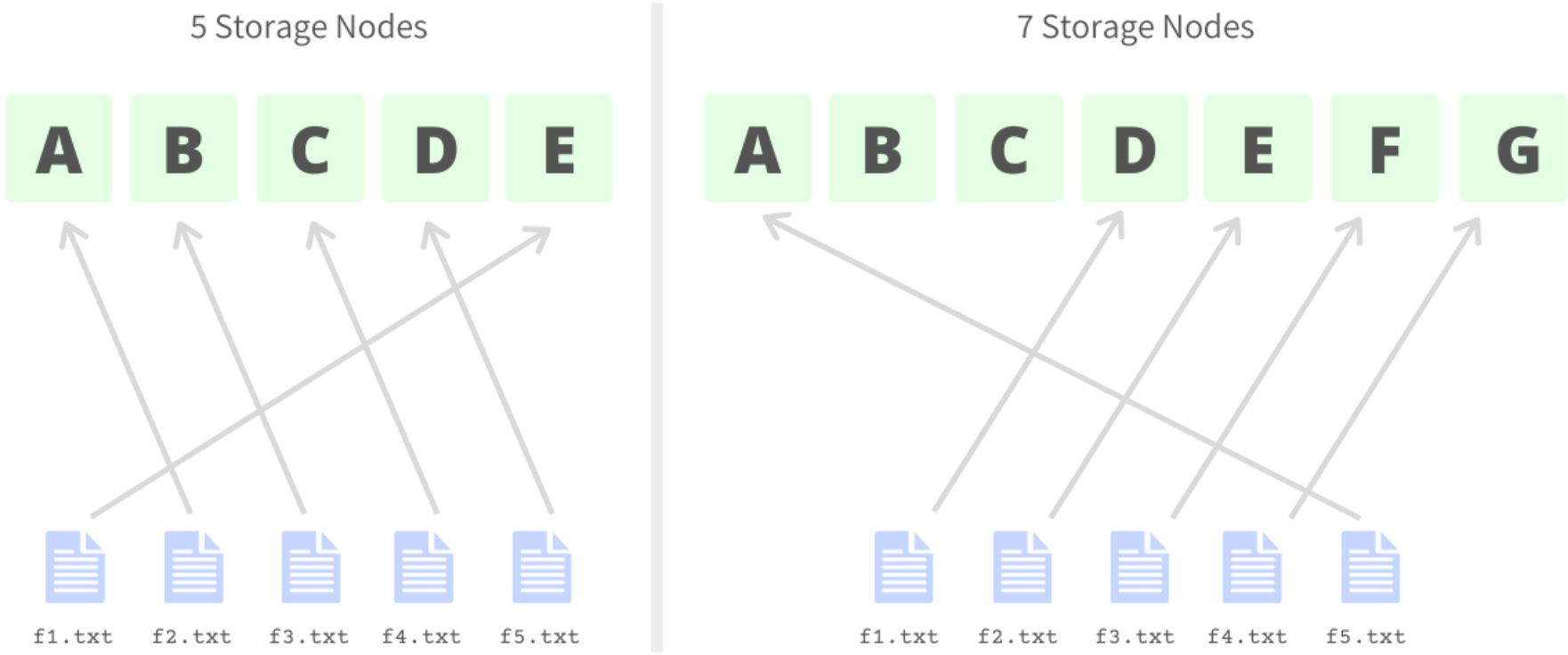
```python
# storage_nodes holding instances of actual storage node objects
storage_nodes = [
    StorageNode(name='A', host='10.131.213.12'),
    StorageNode(name='B', host='10.131.217.11'),
    StorageNode(name='C', host='10.131.142.46'),
    StorageNode(name='D', host='10.131.114.17'),
    StorageNode(name='E', host='10.131.189.18'),
]


def hash_fn(key):
    """The function sums the bytes present in the `key` and then
    take a mod with 5. This hash function thus generates output
    in the range [0, 4].
    """
    return sum(bytearray(key.encode('utf-8'))) % 5


def upload(path):
    # we use the hash function to get the index of the storage node
    # that would hold the file
    index = hash_fn(path)

    # we get the StorageNode instance
    node = storage_nodes[index]

    # we put the file on the node and return
    return node.put_file(path)


def fetch(path):
    # we use the hash function to get the index of the storage node
    # that would hold the file
    index = hash_fn(path)

    # we get the StorageNode instance
    node = storage_nodes[index]

    # we fetch the file from the node and return
    return node.fetch_file(path)
```

Mapping nodes

Suppose Amazon S3 wants to serve some files to you. The files as we know are stored on multiple nodes. By hashing the path to the file, the S3 URL that is, the file will be stored on a particular node, and you can upload the file there or retrieve it

From https://arpit.substack.com/p/consistent-hashing-with-binary-search

But if we change the number of nodes, the order gets all messed up, and we have to move files from one node to another. This will not do!



Scaling up the Storage Nodes

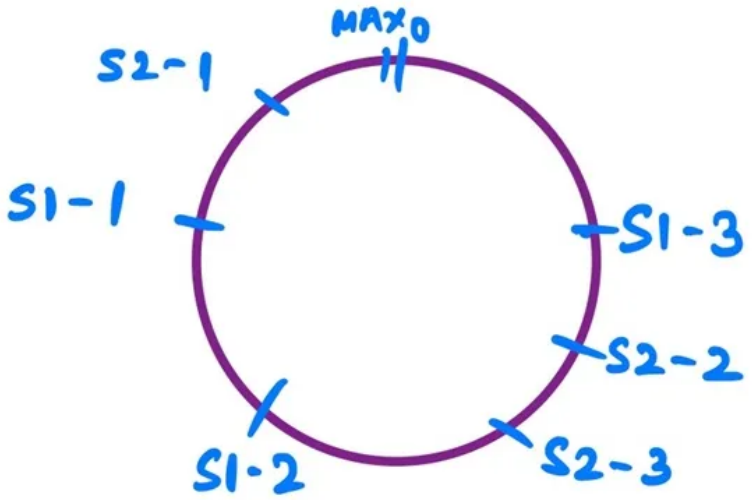
If we go from 5 servers to 7, then the new servers should be filled with 2/7th of the keys. Keys should be evenly chosen from the 5 “old” servers. Keys should only move to the new servers, never between the old servers. Similarly, if we need to remove a server (say, because it crashed), then the keys should be evenly distributed across the remaining live servers.

From https://dgryski.medium.com/consistent-hashing-algorithmic-tradeoffs-ef6b8e2fcae8

Consistent Hashing

A brilliant idea comes to the rescue. Map the hash function's target to a circle of radius 1. Map the server names there first. Actually to create more maps, create virtual nodes per server and map these instead, so that they are mapped to the circle more uniformly.

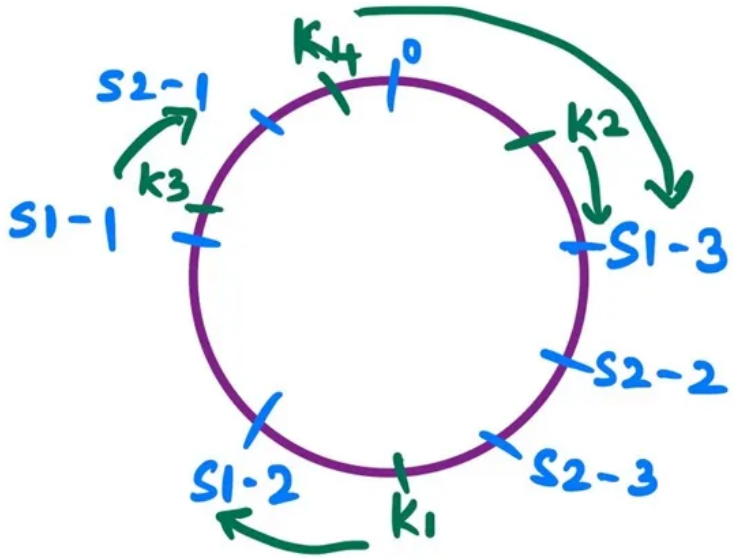
Consistent Hashing: Hashing virtual nodes



Assumption:
- 3 virtual nodes per server
- 2 servers: S1 and S2

Consistent Hashing in Action

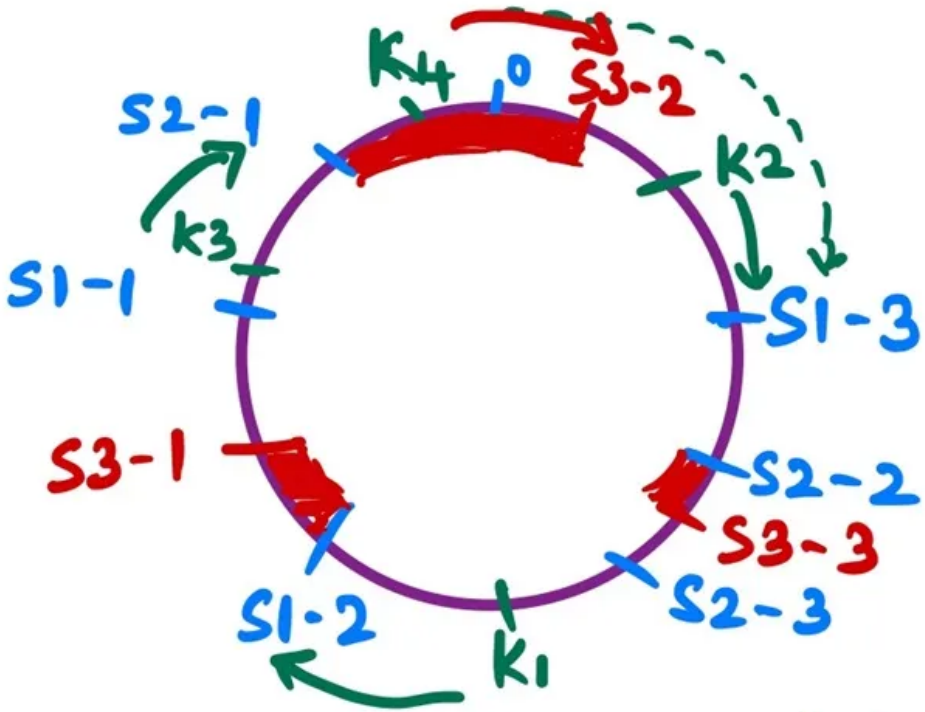
Now when you map a key like a S3 URL, you map it to the same circle, and assign it the server immediately to its right.



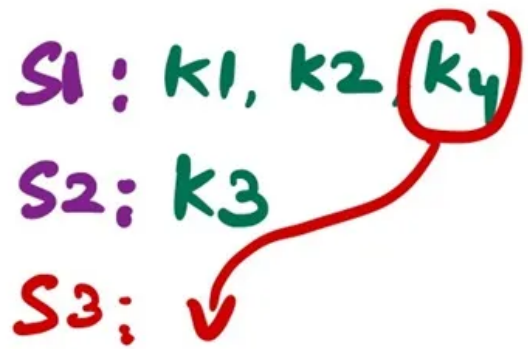
S1: K1, K2, K4
S2: K3

From https://blog.techlanika.com/consistent-hashing-an-introduction-264beb5e871d

Consistent Hashing: When a server is added...



Range of impacted keys



Now your S3 becomes popular so you need to add more nodes to keep up with demand. So you add 3 virtual nodes the new server. Only K4 has to move. The more virtual nodes you have per server, the less keys have to move.

AWS Dynamo-DB

- Amazon was using relational databases, a lot of which failed in the holiday season of 2004
- they replaced these by dynamo, but at first it had no automatic cluster and node management
- Once this was added, internal uptake soared, and it is now a very valuable external, managed product

During the last 10 years, hundreds of thousands of customers have adopted DynamoDB. It regularly reaches new peaks of performance and scalability. For example, during the last Prime Day sales in June 2021, it handled trillions of requests over 66 hours while maintaining single-digit millisecond performance and peaked at 89.2 million requests per second. Disney+ uses DynamoDB to ingest content, metadata, and billions of viewers actions each day. Even during unprecedented demands caused by the pandemic, DynamoDB was able to help customers as many across the world had to change their way of working, needing to meet and conduct business virtually. For example, Zoom was able to scale from 10 million to 300 million daily meeting participants when we all started to make video calls in early 2020.

From https://aws.amazon.com/blogs/aws/happy-birthday-dynamodb/

Why was content hashing invented?

- It was created to cache web pages in 1997 in Tom Leighton's lab by him and Daniel Lewin, down the road in MIT, so as to handle flash crowds, such as a Taylor Swift ticket booking or a Kardashian "Break-the-Internet" moment.
- It works by caching a set of web-pages on different servers, out in different countries, and modifying the name resolution of web-sites to dynamically hit these nodes.
- It dynamically switches nodes according to traffic, country, node aliveness, etc. It was commercialized as the company Akamai.
- IPO'ed Fall 1999 at $26, was $140 by end of day, $327.63 by end of year. Yahoo was a charter customer.

Tragedy and Redemption

- Then 2000 struck. Dot-Com Bust. Even so the stock managed to keep around 100 for a long time, because of the quality of the algorithm and the company.
- But by 2001 the economy was really bad and the stock went below $3. Companies were dying left and right and things could not get worse. But they did. On Sep 11, on AA flight 11 from Boston, Daniel Lewin tried to foil the hijacking and was killed 1/2 hour before the plane crashed into the WTC.
- Ironically news customers using Akamai were able to handle the frash traffic on Sep 11. Customers returned and by 2004 Akamai was fully profitable again. See http://www.fundinguniverse.com/company-histories/akamai-technologies-inc-history/.