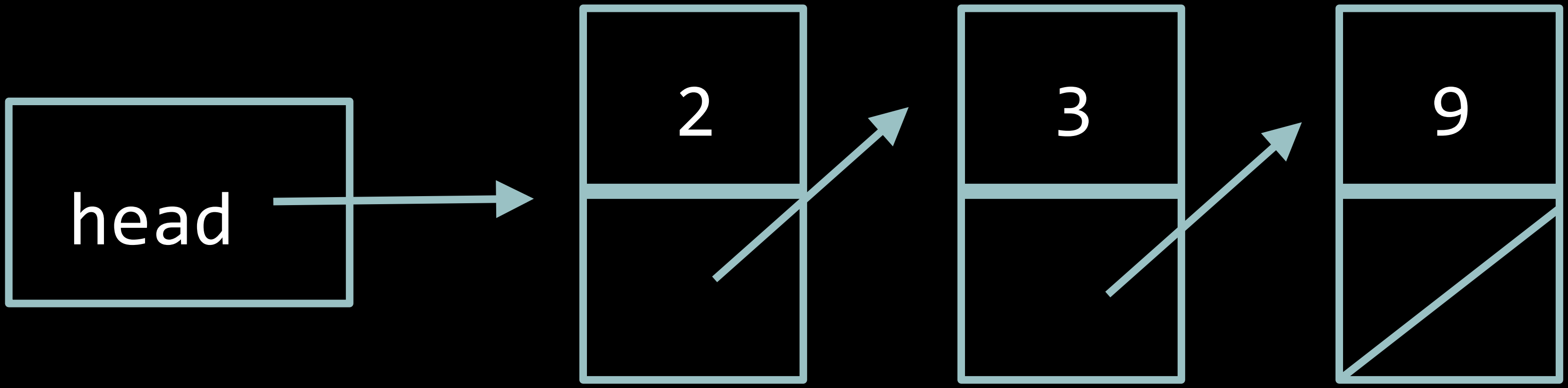
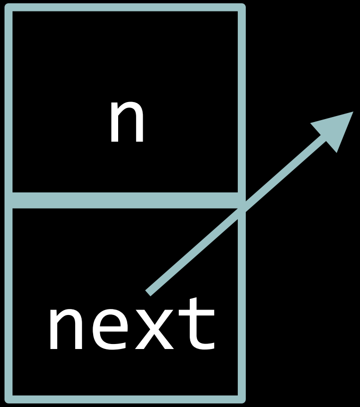
Topics (Non-exhaustive)

- Hash Tables
- Tries
- Trees
- Stacks
- Queues
- TCP/IP
- HTTP
- CSS
- PHP
- MVC
- SQL
- Javascript

Linked Lists

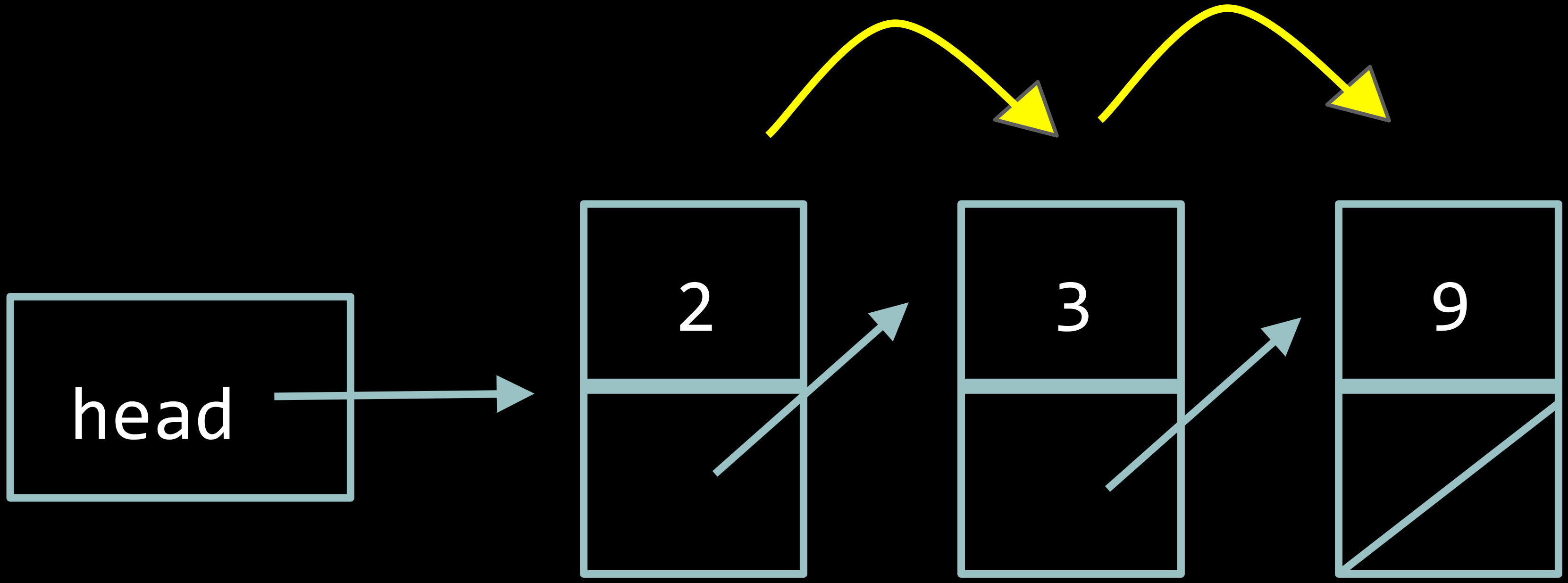


Nodes



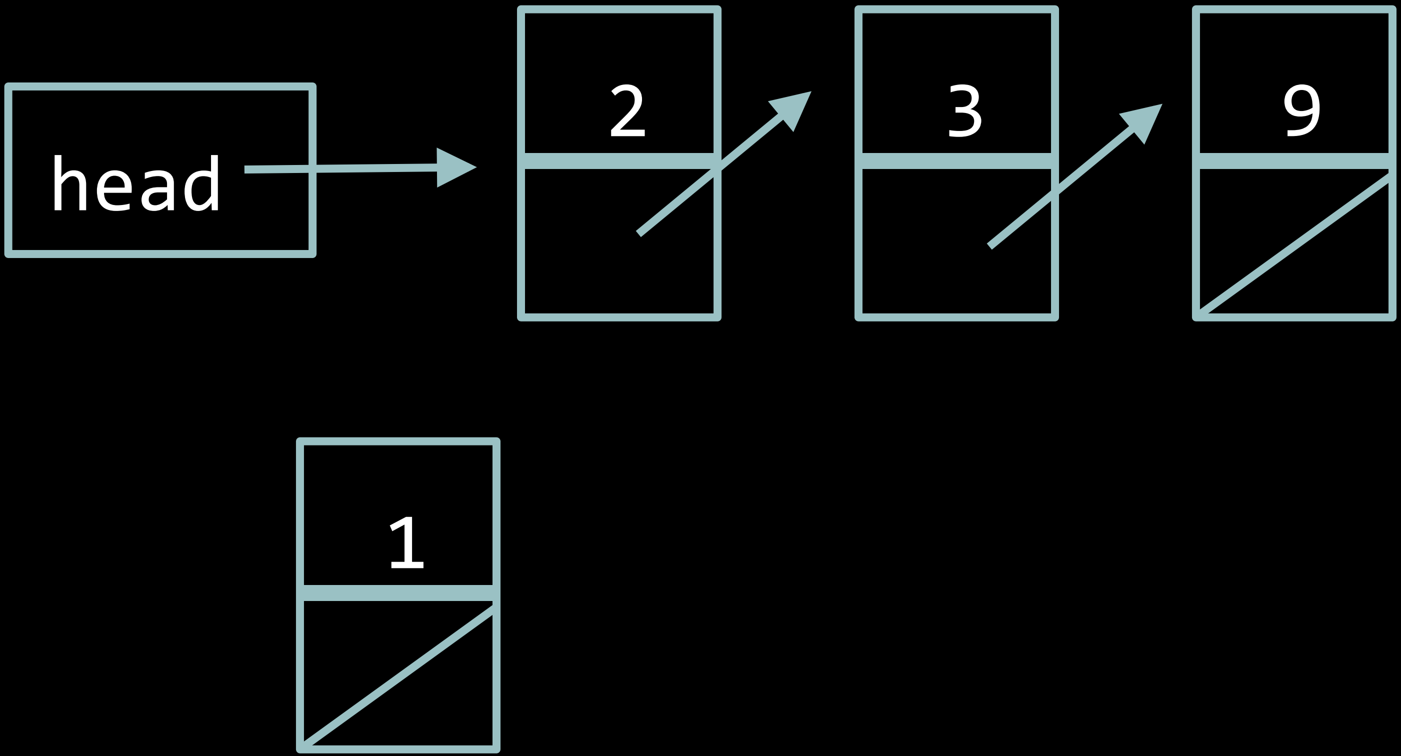
```
typedef struct node
{
        int n;
        struct node*
next;
}
node;
```

Search

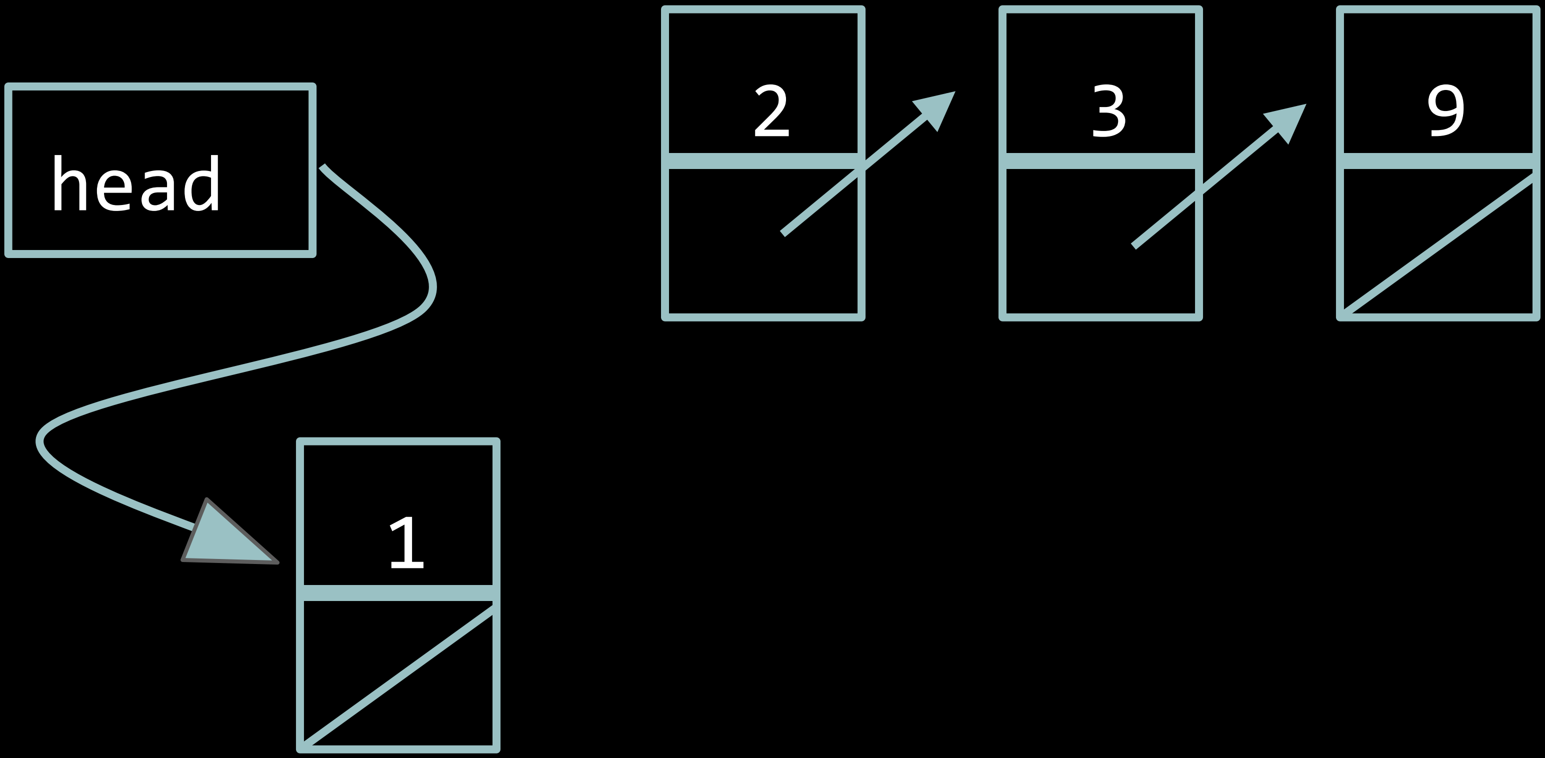


```
bool search(node* list, int n)
{
   node* ptr = list;

   while (ptr != NULL)
   {
           if (ptr->n == n)
     {
                return true;
     }
           ptr = ptr->next;
   }
   return false;
}
```

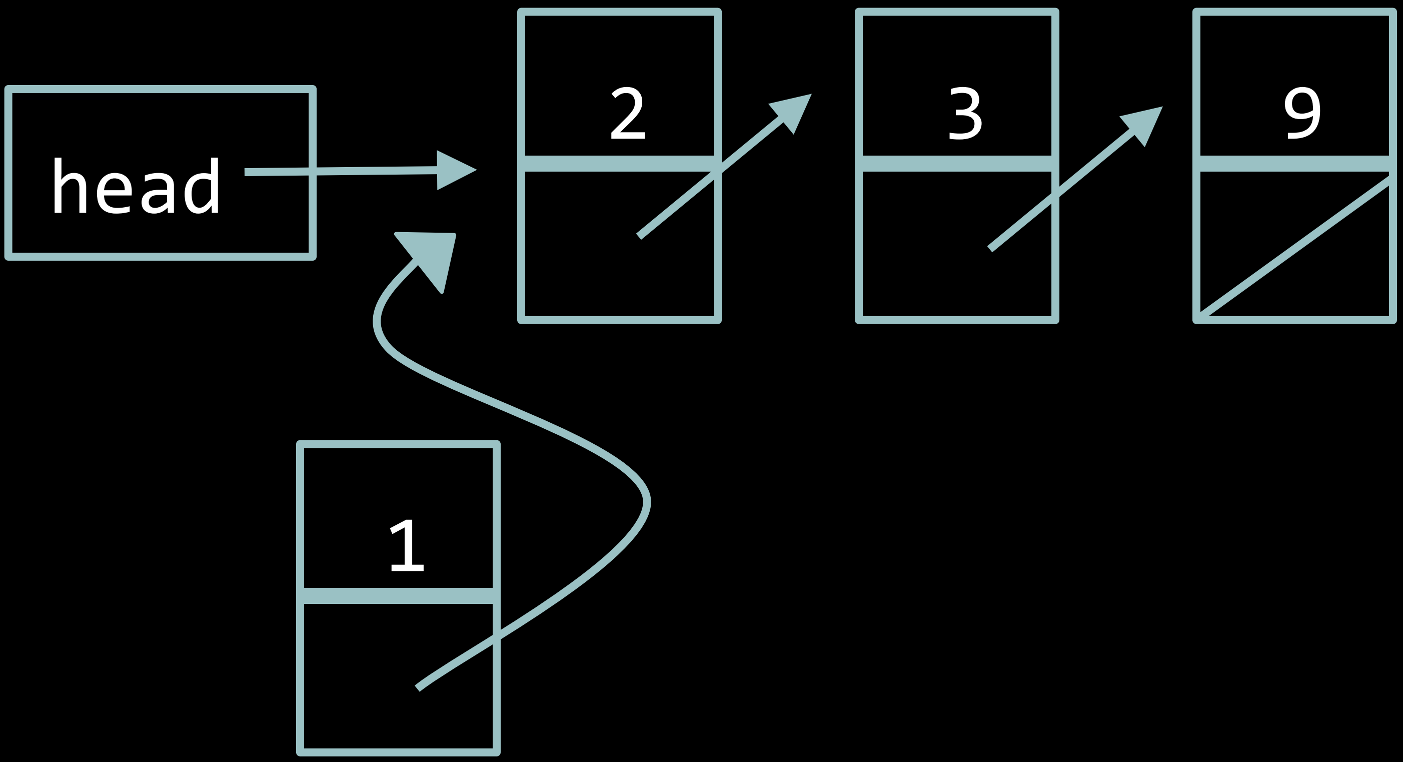
Insertion



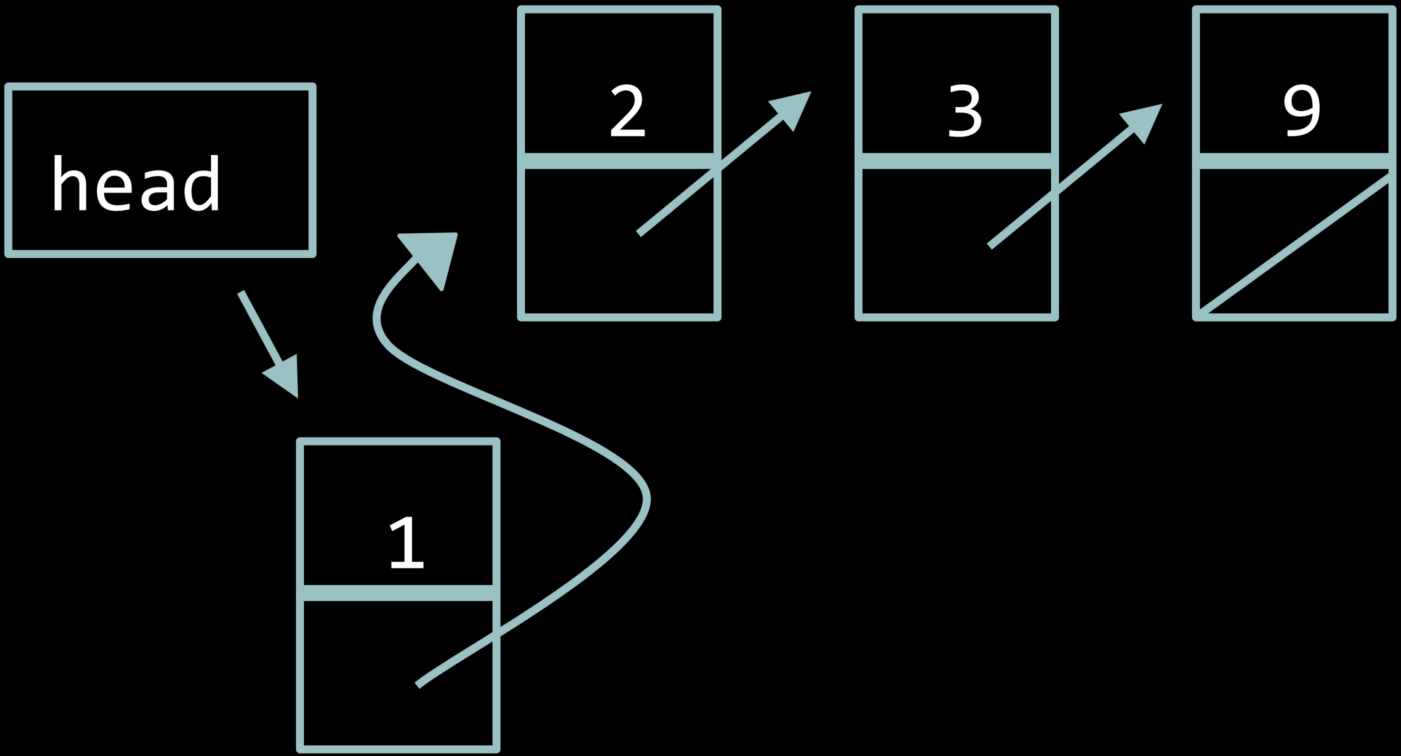
Insertion



Insertion

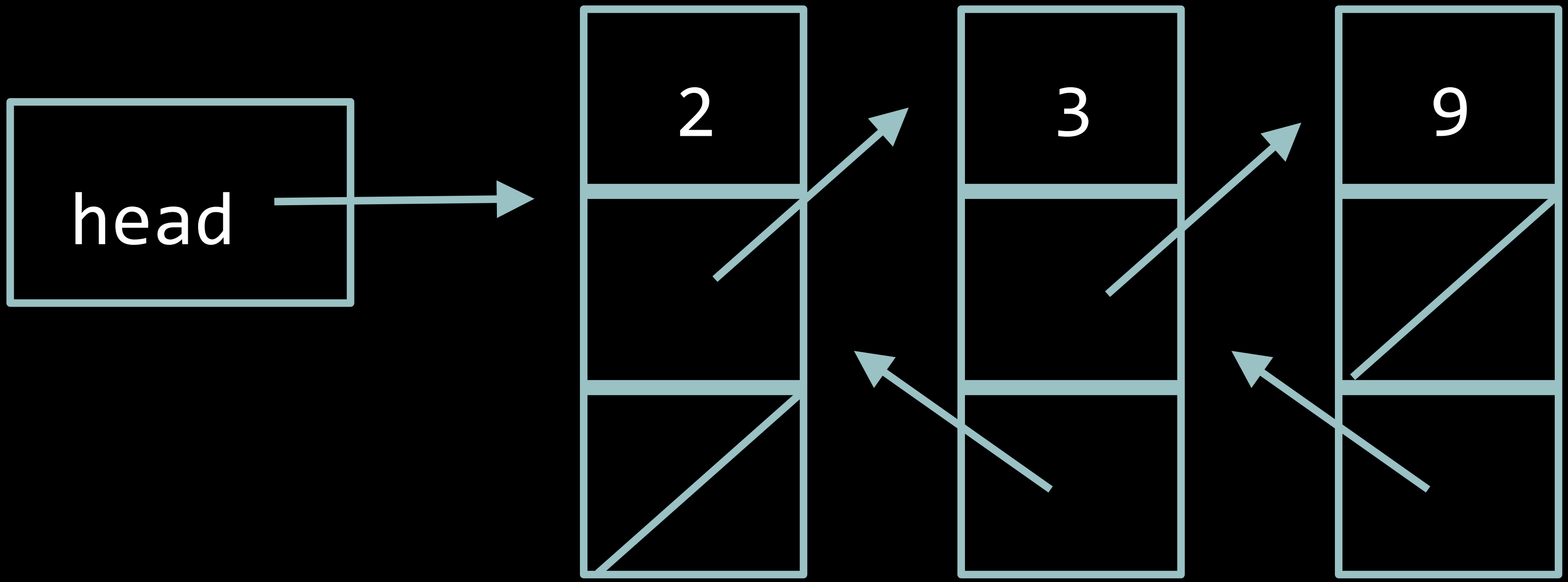


Insertion

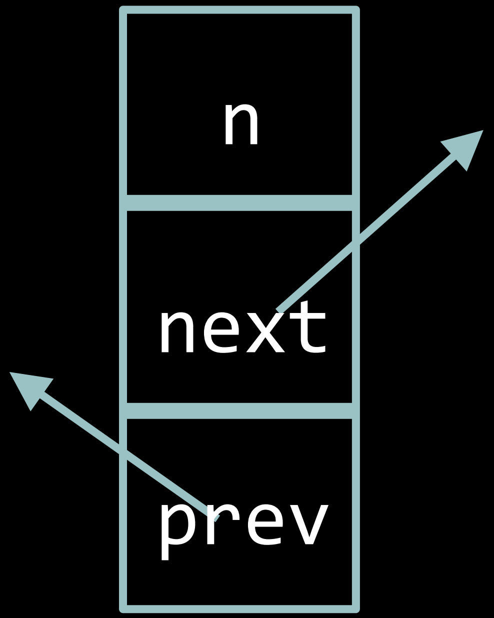


```c
void insert(int n)
{
    // create new node
    node* new = malloc(sizeof(node));

    // check for NULL
    if (new == NULL)
    {
        exit(1);
    }
    // initialize new node
    new->n = n;
    new->next = NULL;

    // insert new node at head
    new->next = head;
    head = new;
}
```

Doubly Linked Lists

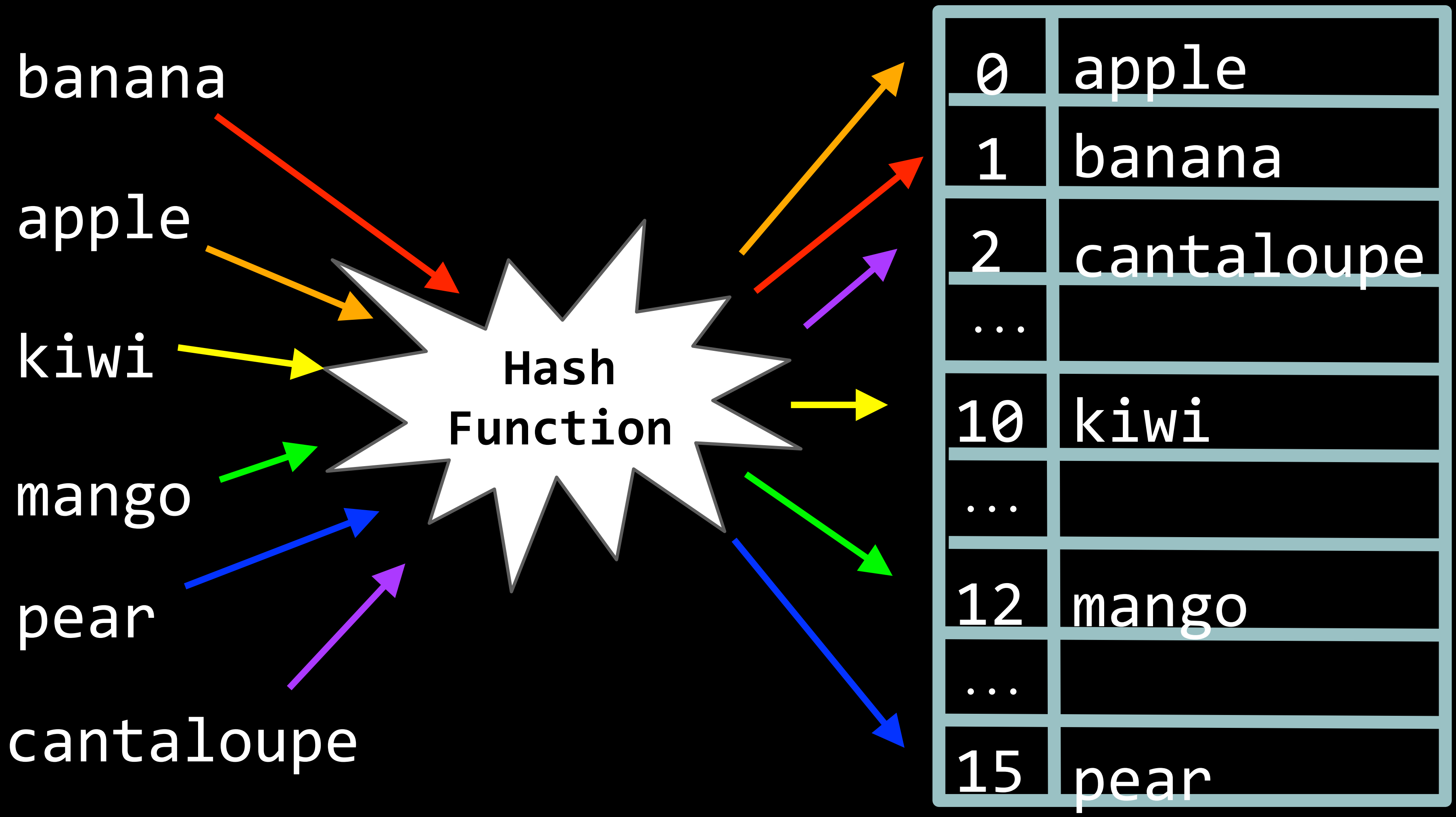


DLL Nodes

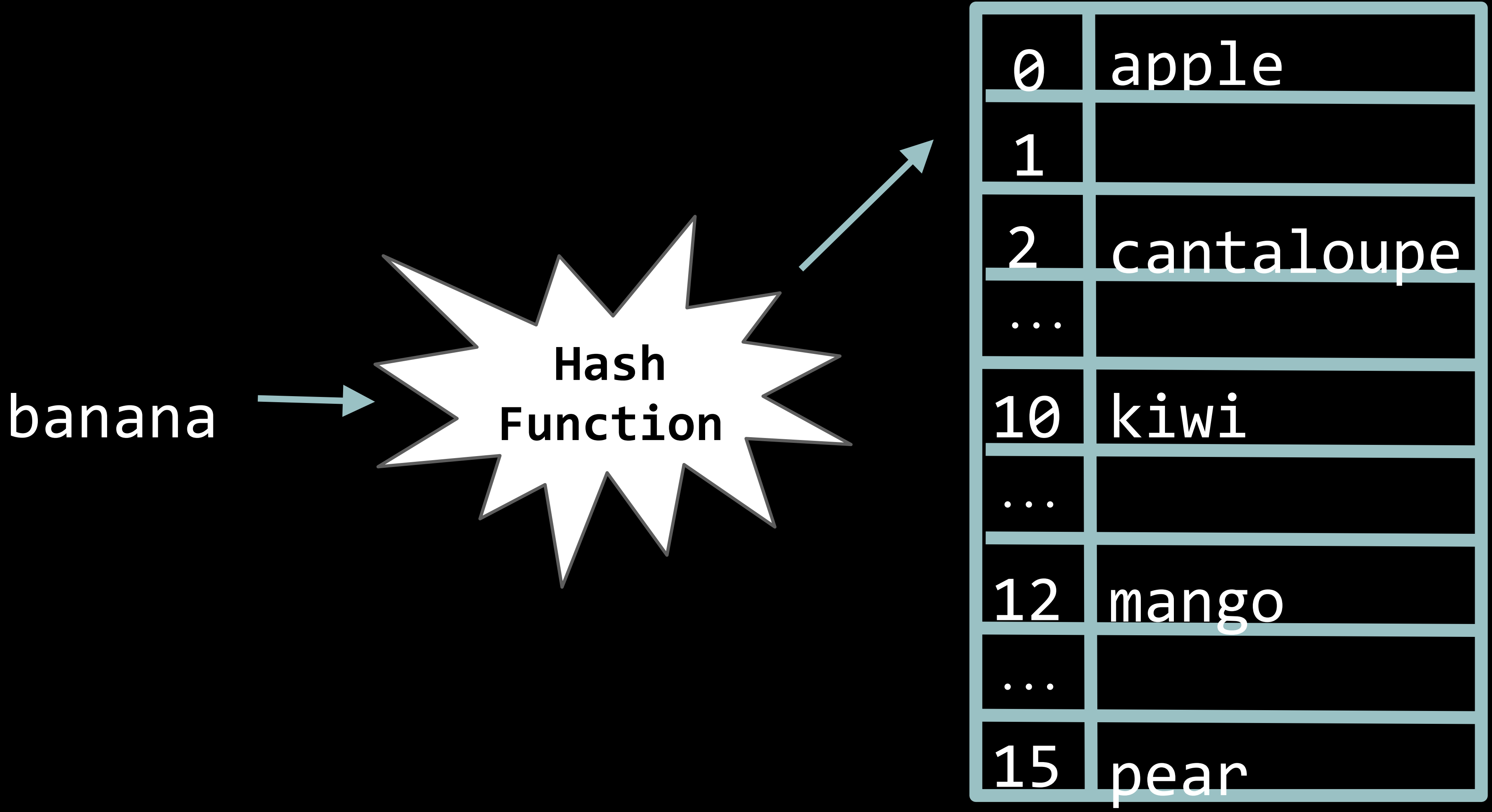


```
typedef struct  node
{
        int n;
        struct node* next;
        struct node*
prev;
}
node;
```

Hash Tables


banana
apple
kiwi
mango
pear
cantaloupe
Hash Function
0 apple
1 banana
2 cantaloupe
...
10 kiwi
...
12 mango
...
15 pear

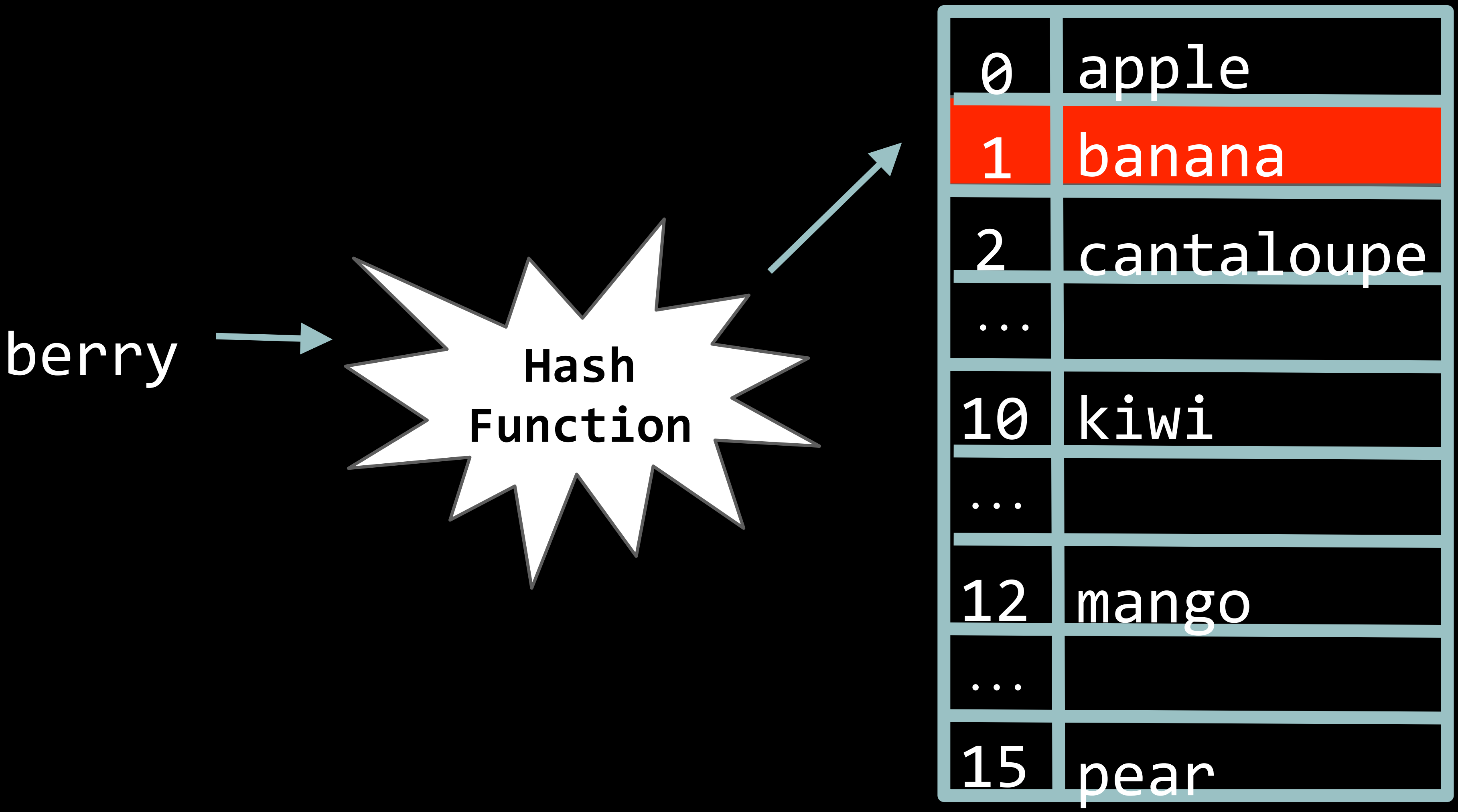
Hash Function



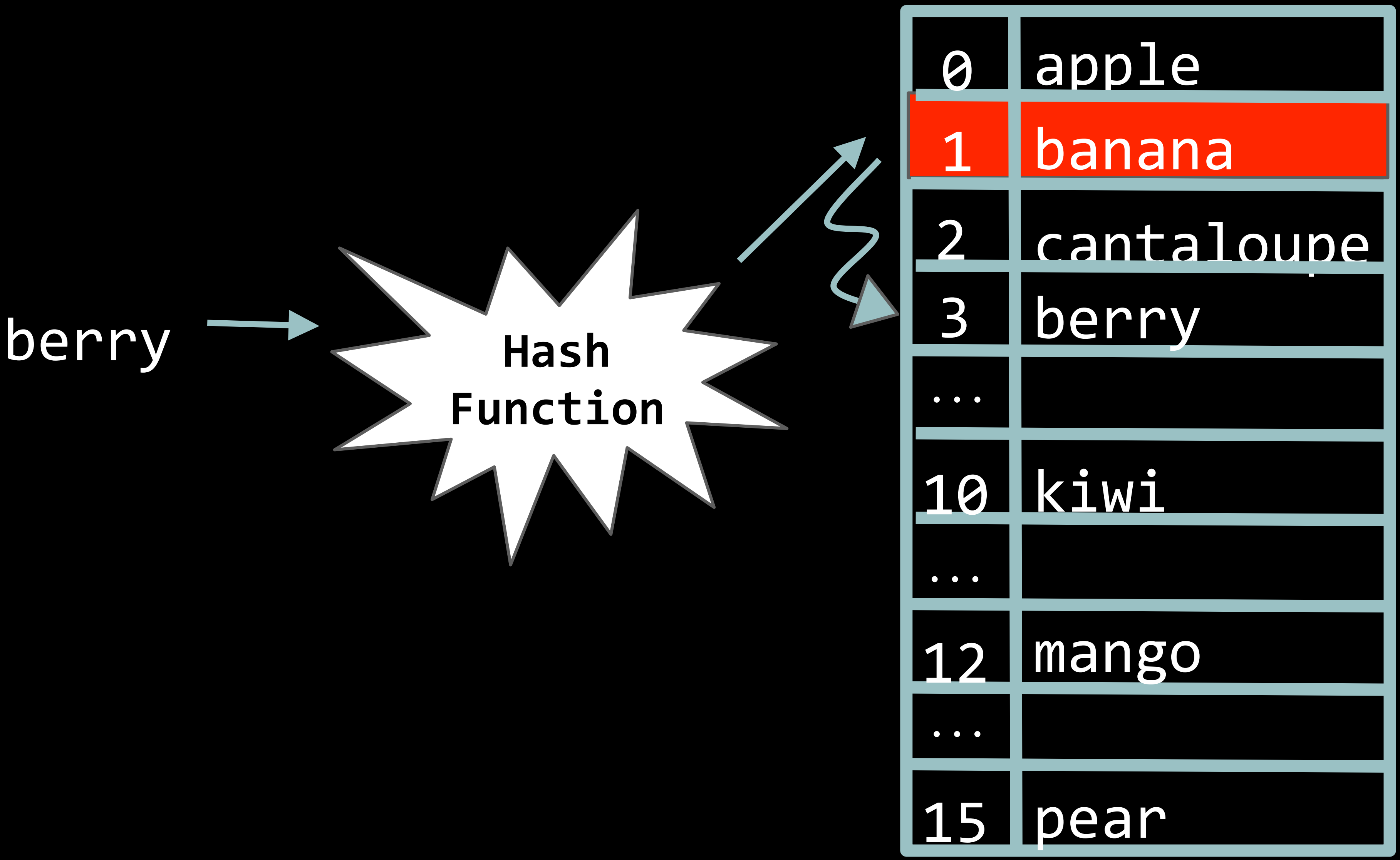
Hash Function Example

```
int hash_function(char* key)
{
        // hash on first letter of string
        int hash = toupper(key[0]) - 'A';

    return hash % SIZE;
}
```

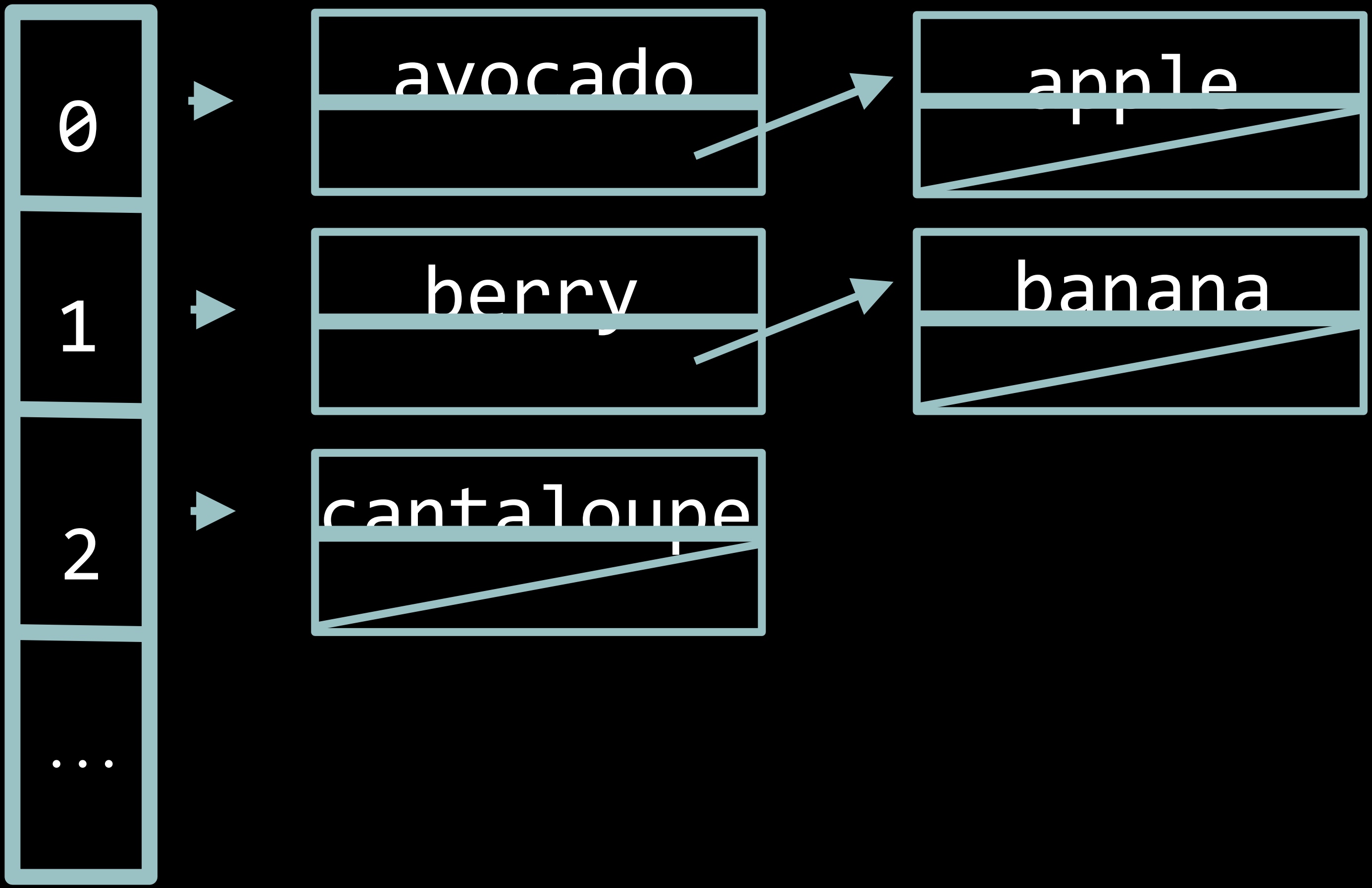
Collisions


berry
Hash Function
0
apple
1
banana
2
cantaloupe
...
10
kiwi
...
12
mango
...
15
pear

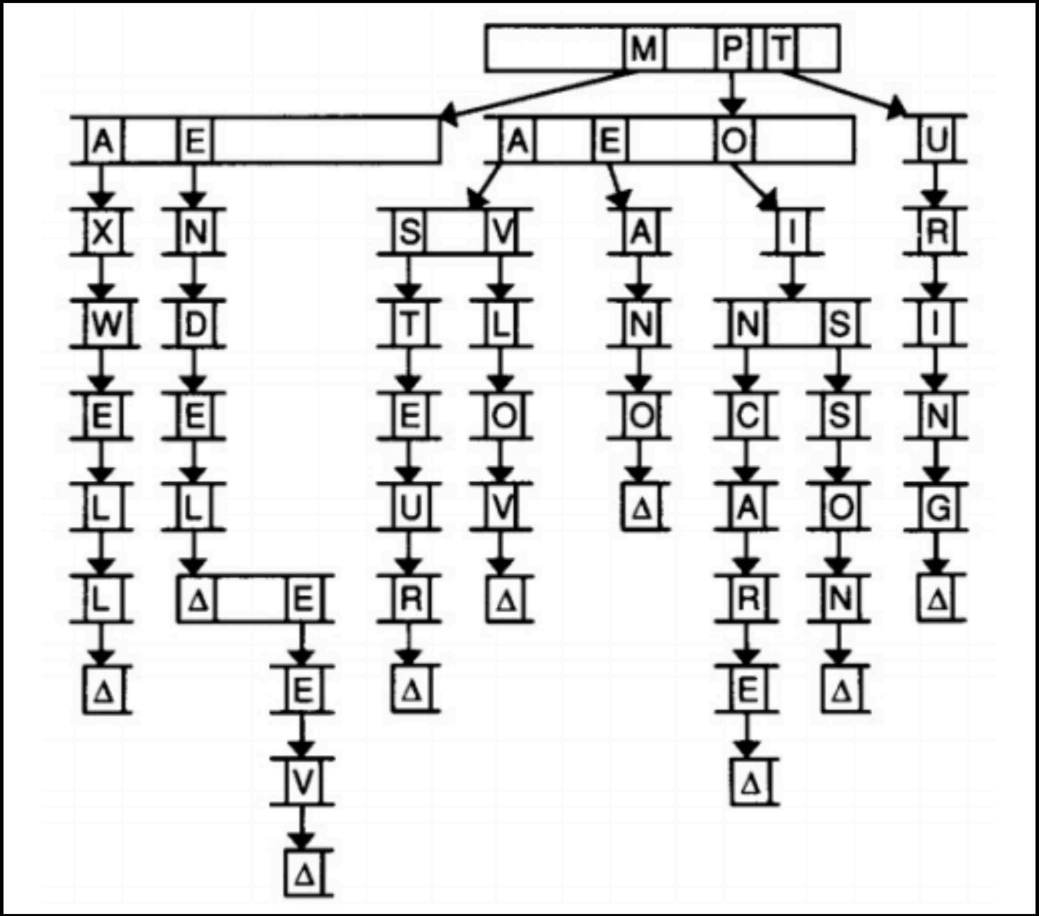
Linear Probing



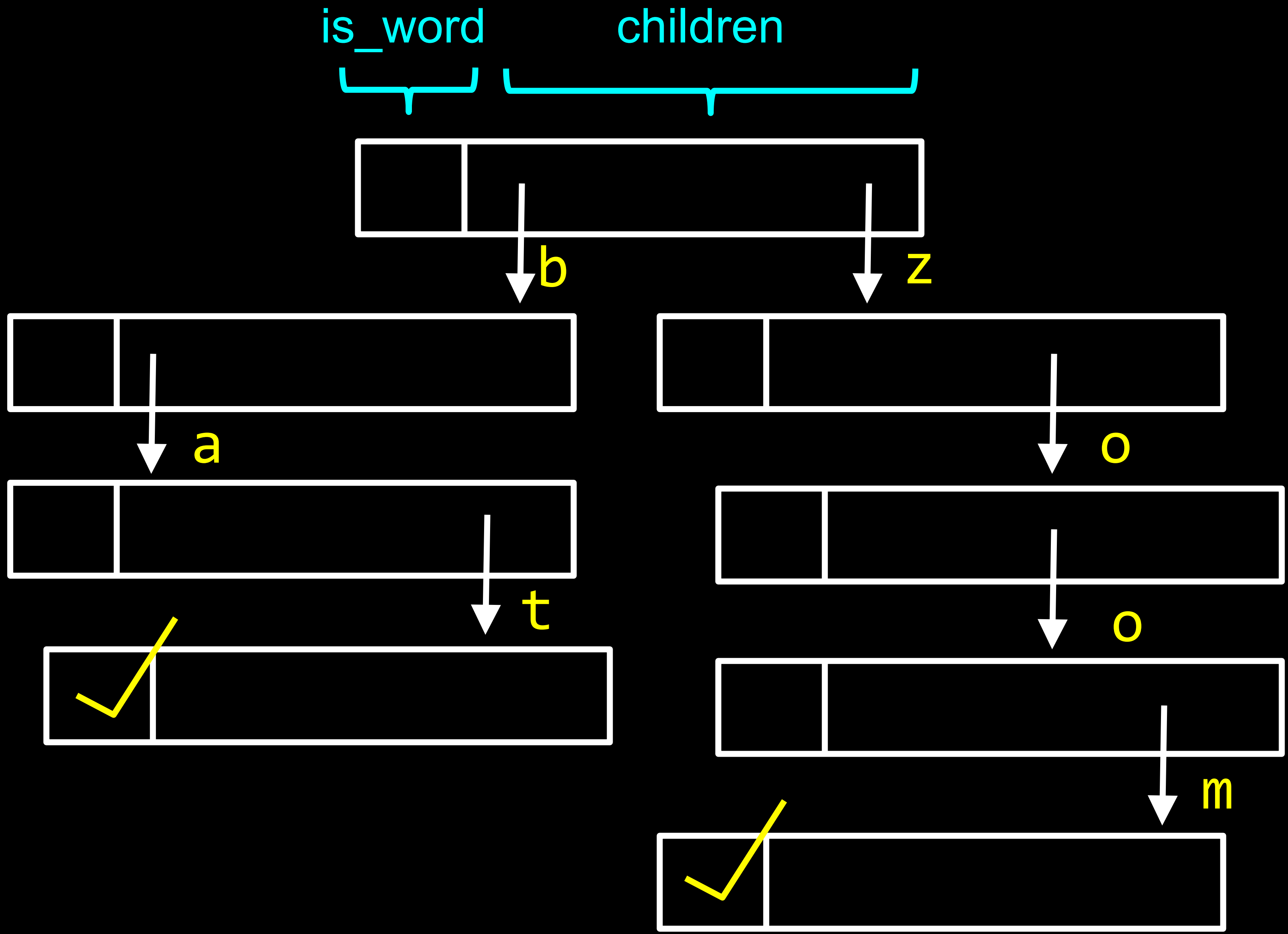
Separate Chaining

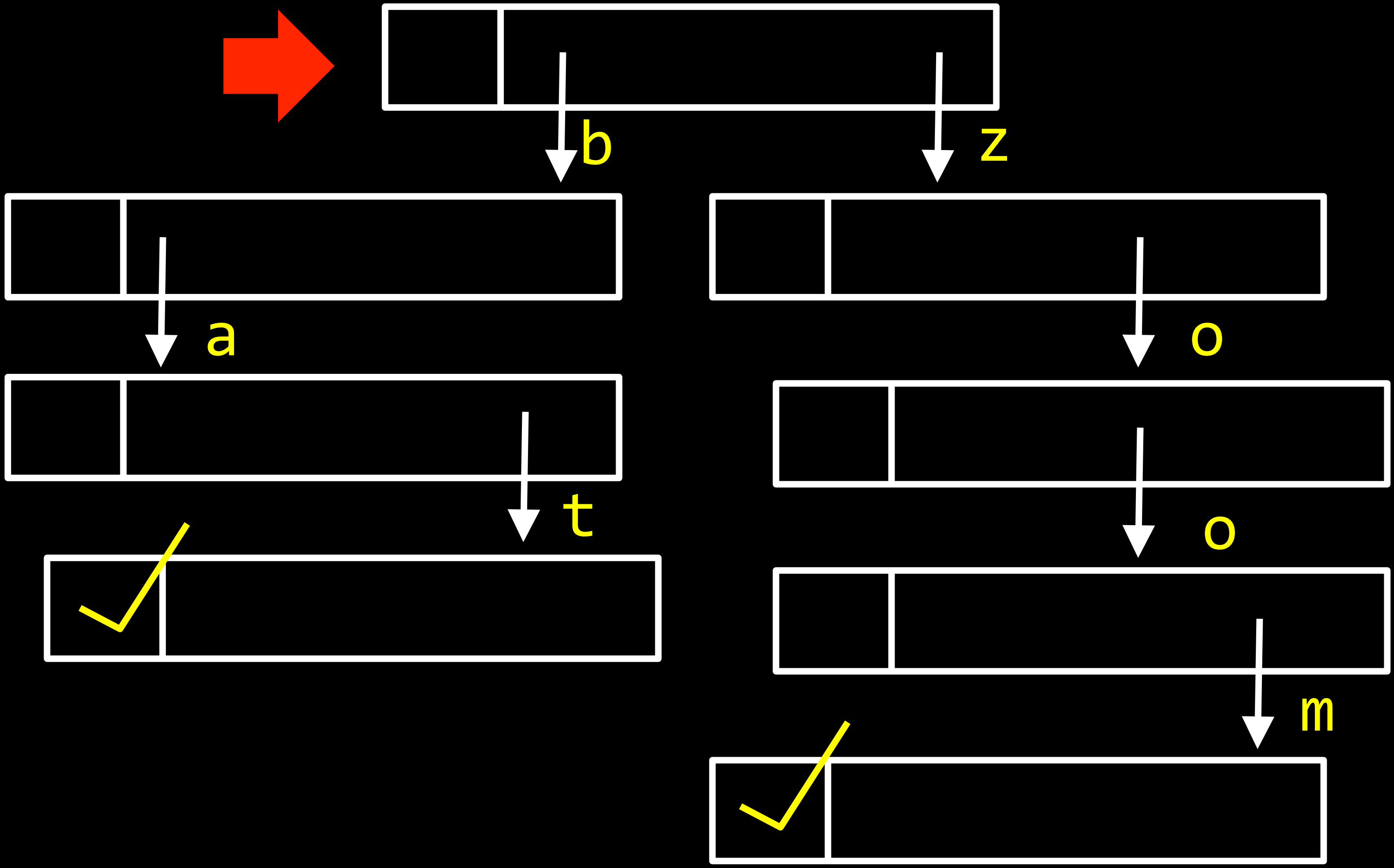


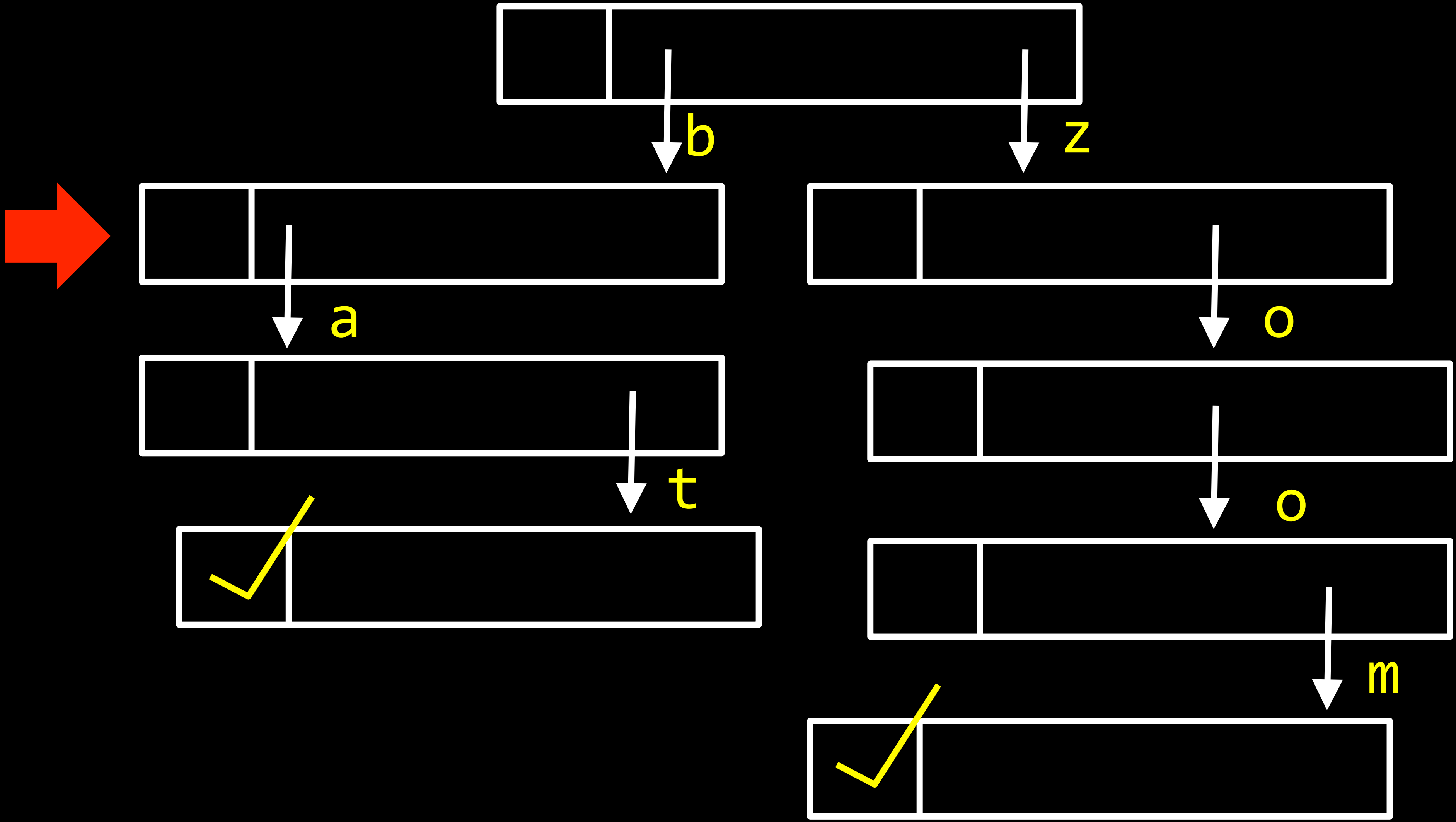
Tries

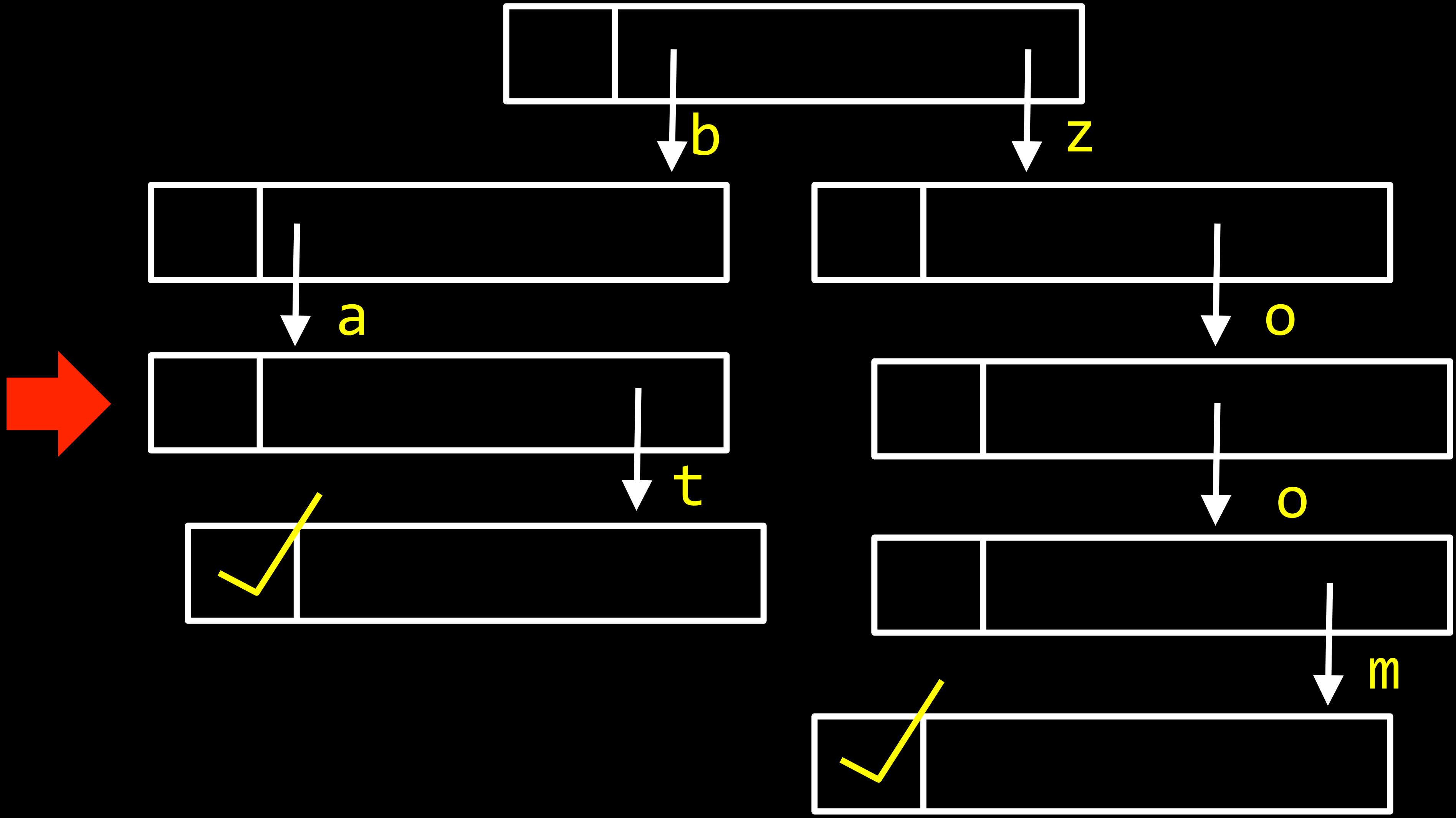

M P T
A E
A E O
U
X N
S V
A
I
R
W D
T L
N
N S
I
E E
E O
O
C S
N
L L
U V
Δ
A O
G
L
Δ E
R
Δ
R N
Δ
Δ
E
Δ
E
Δ
Δ
V
Δ

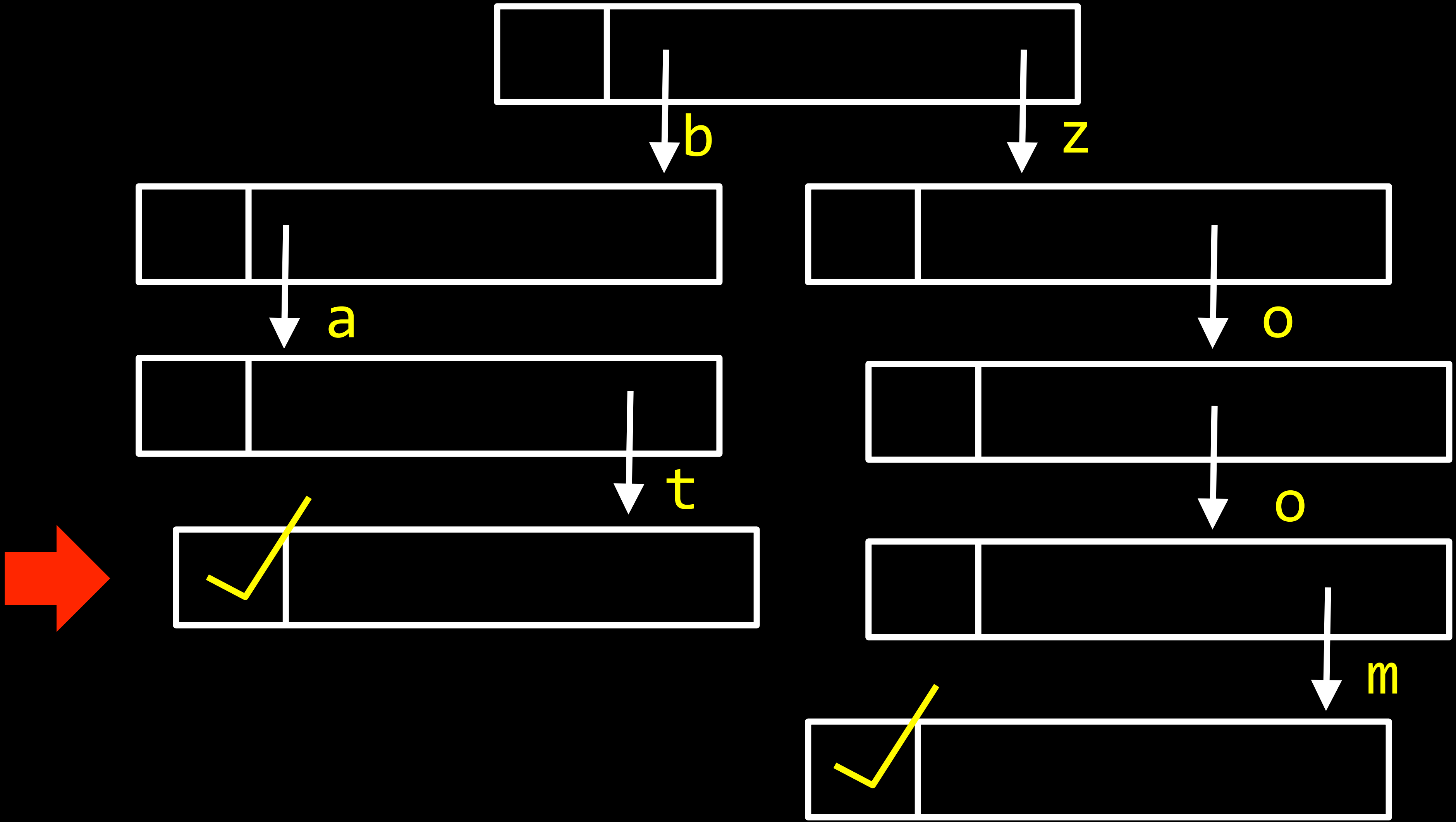
```
typedef struct node
{
    // marker for end of word
     bool is_word;

     // pointers to other nodes
     struct node* children[27];
}
node;
```

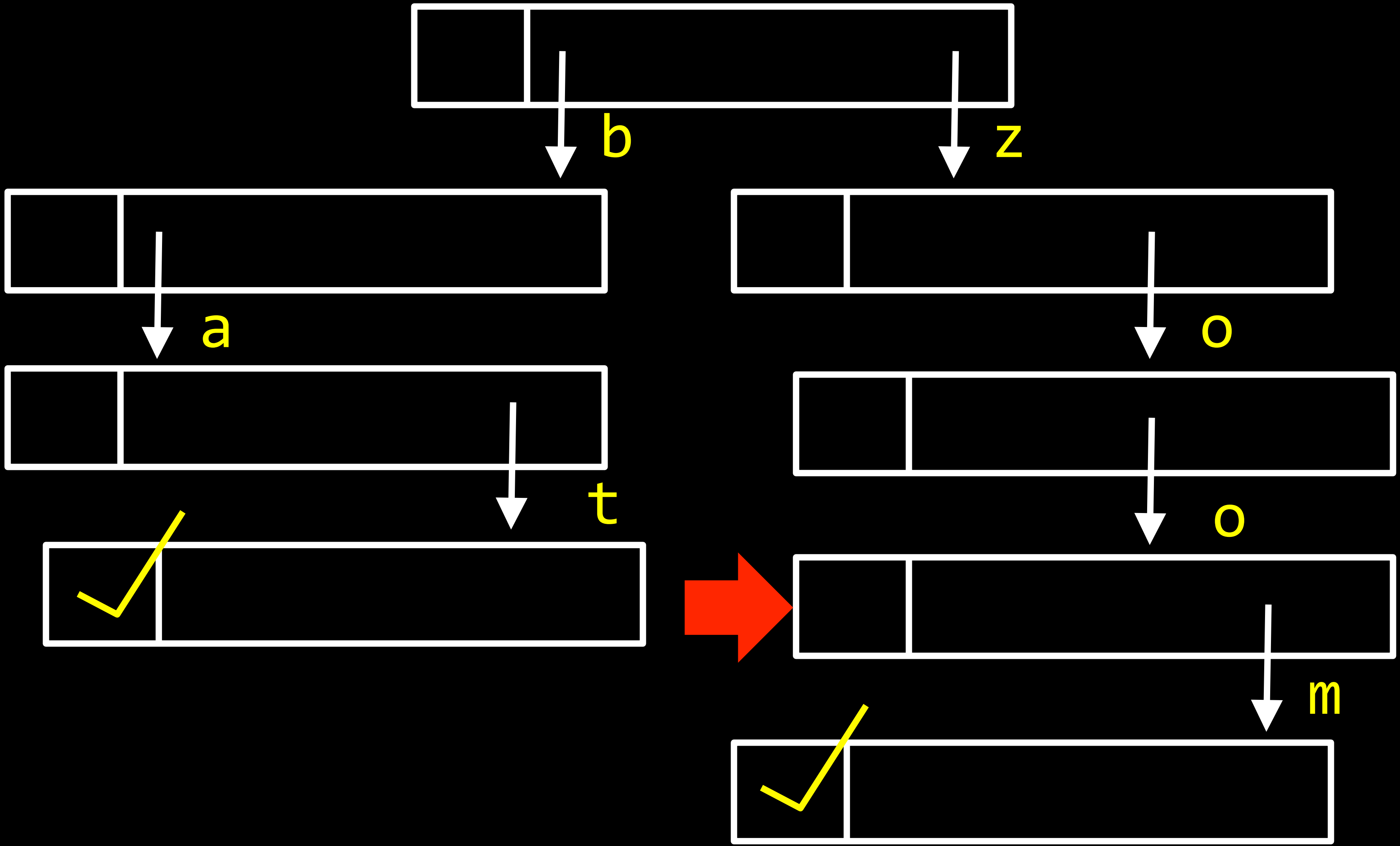

is_word
children
b
z
a
o
t
o
m

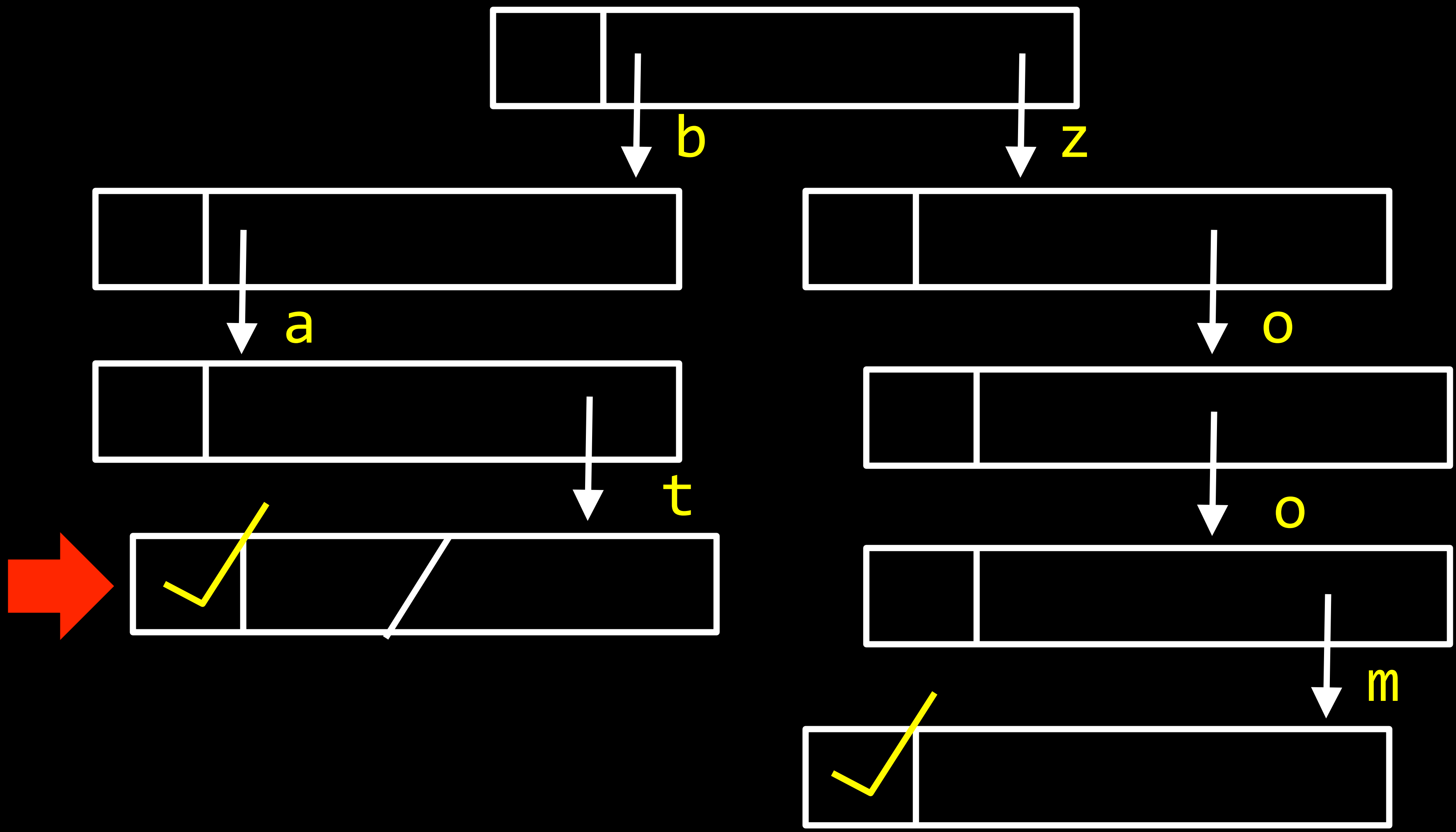

b
z
a
o
t
o
m

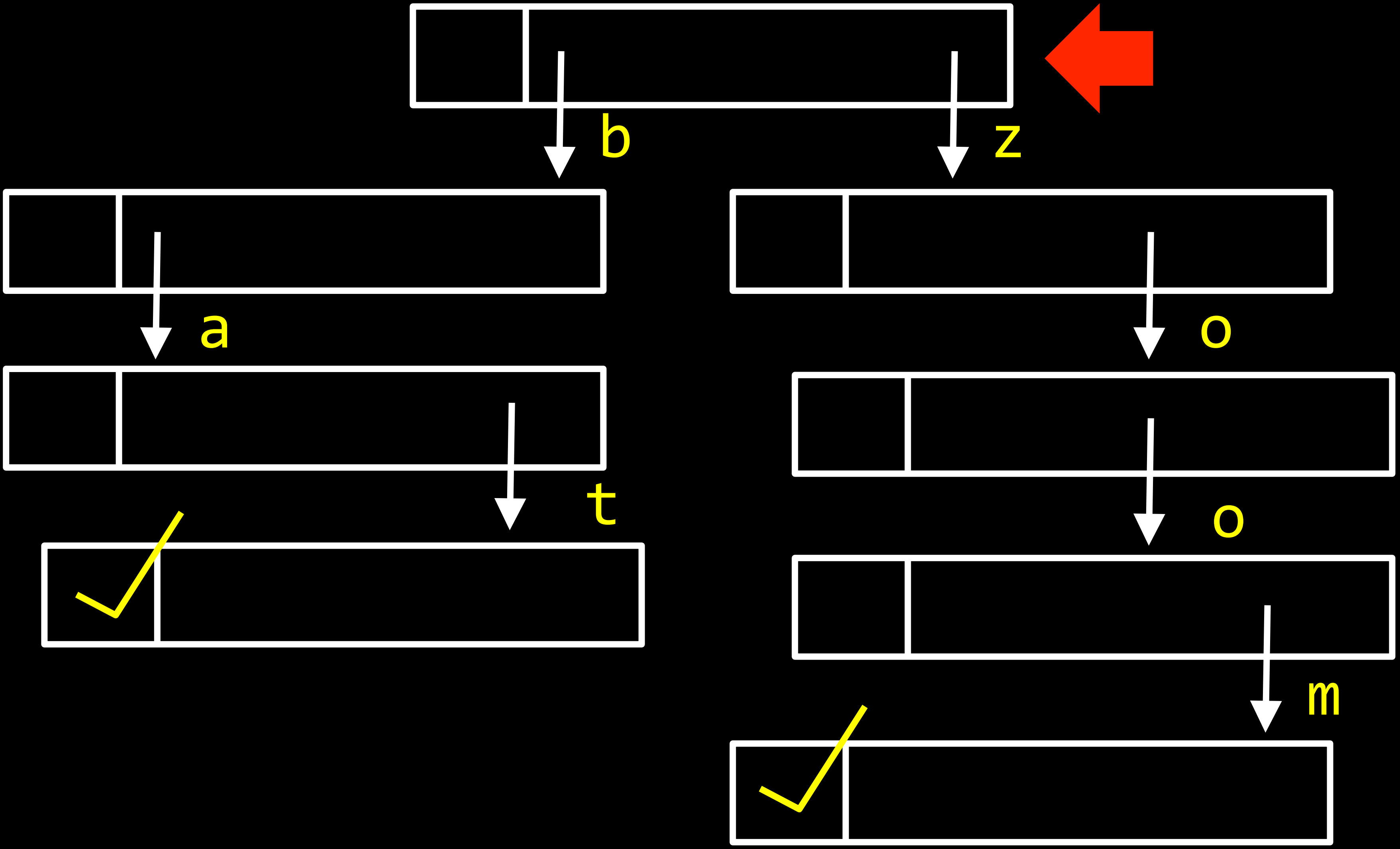

b
z
a
o
t
o
m

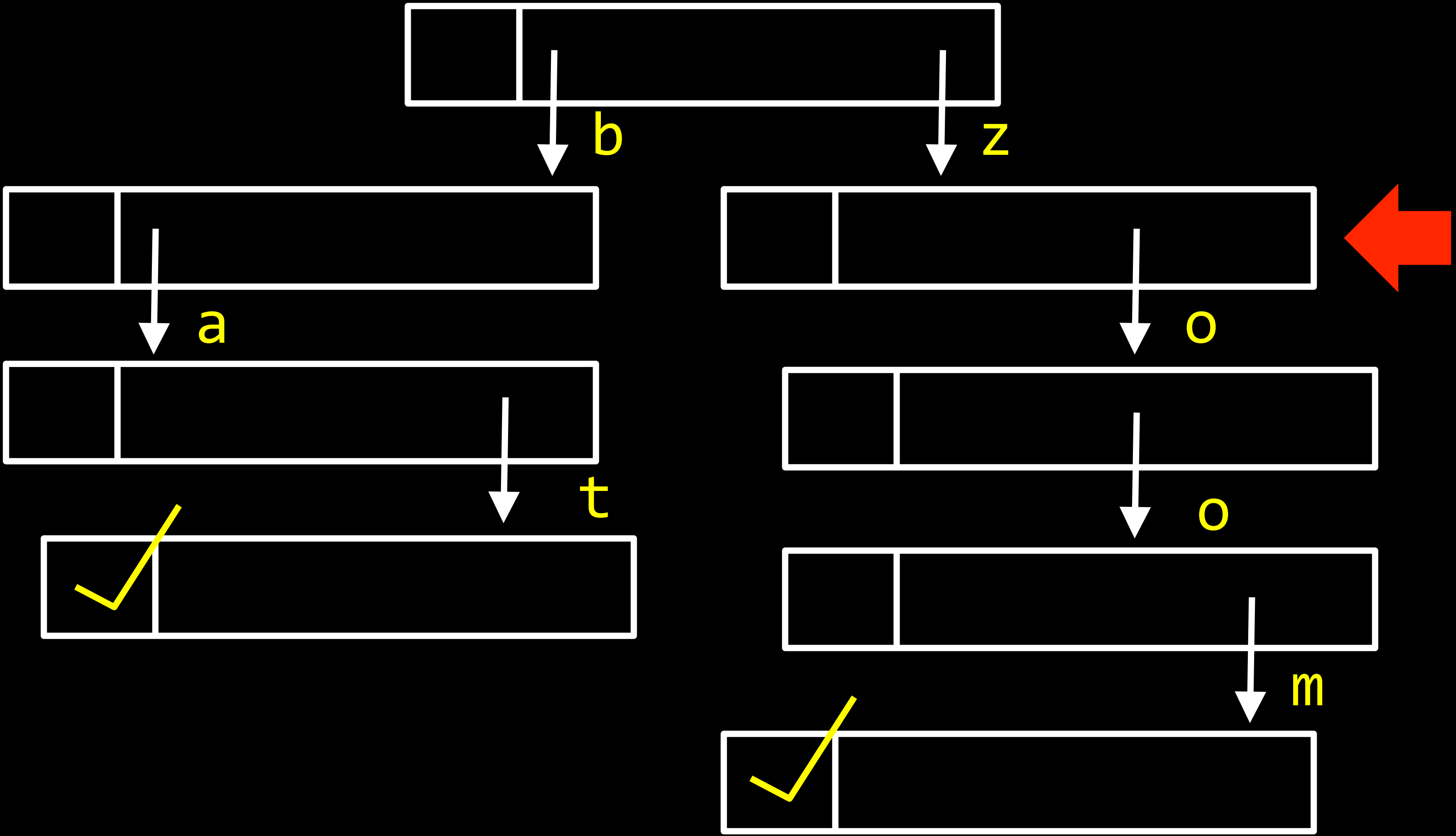

b
z
a
o
t
o
m

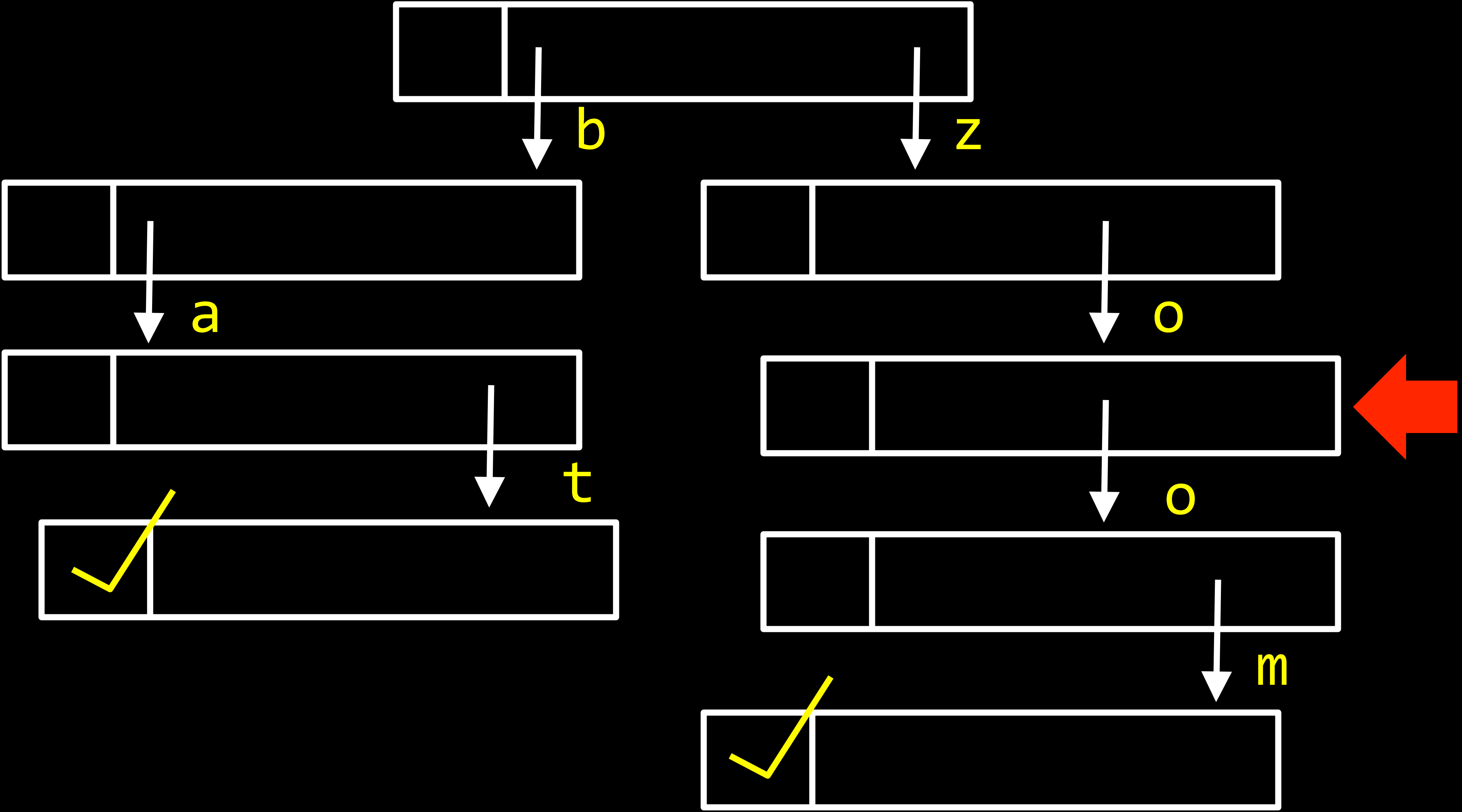

b
z
a
o
t
o
m

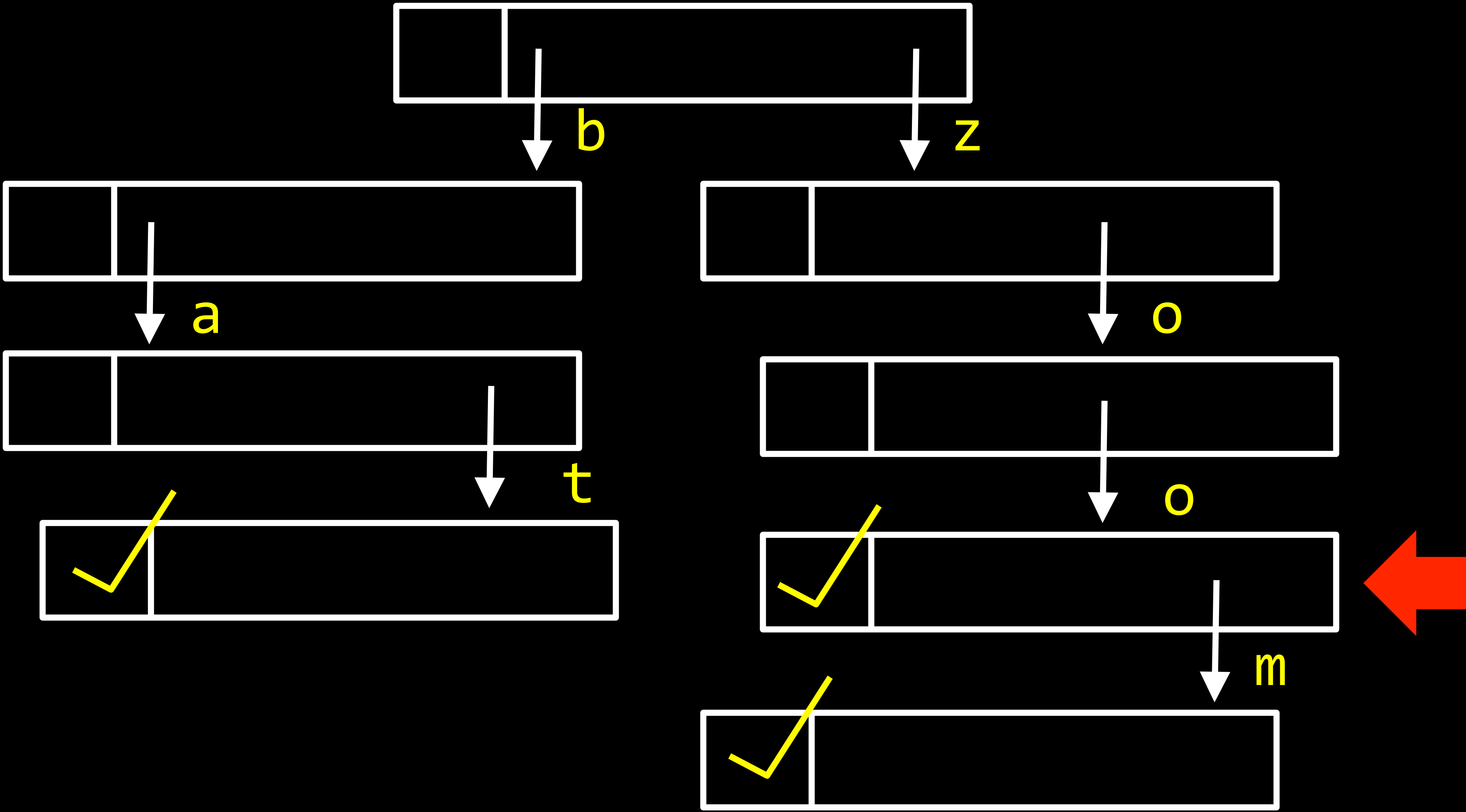

b
z
a
o
t
o
m

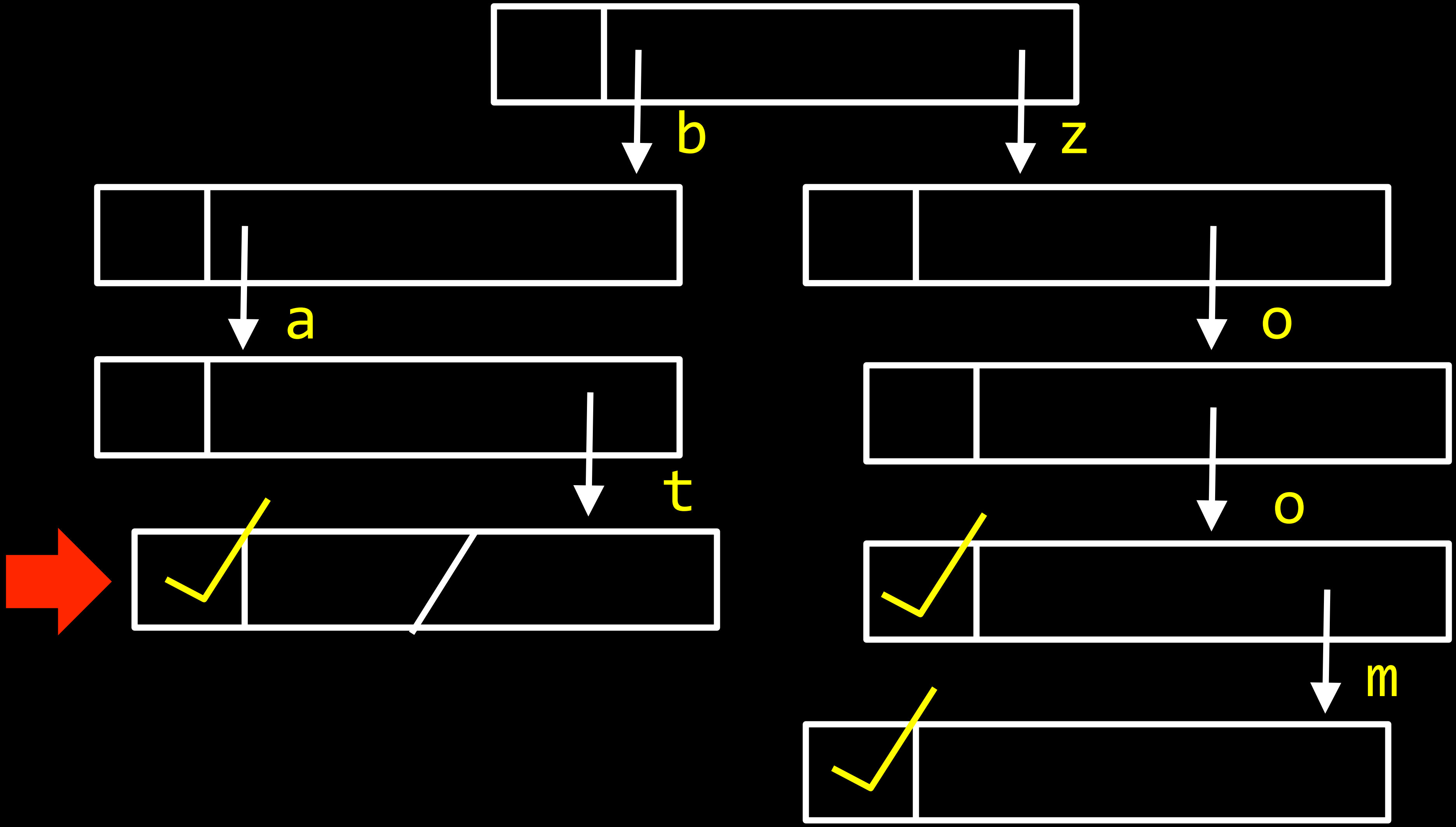

b
z
a
o
t
o
m

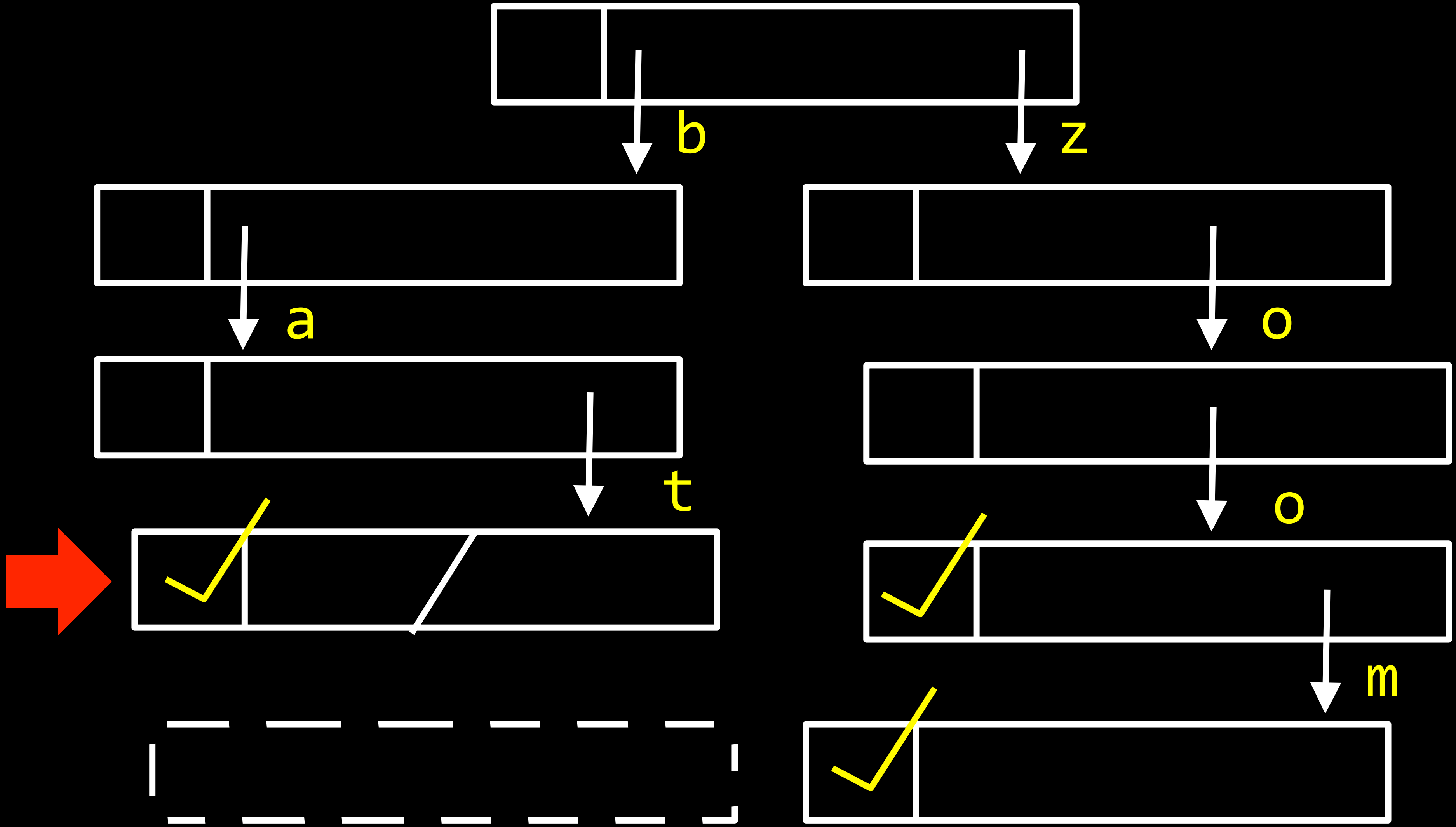

b
z
a
o
t
o
m

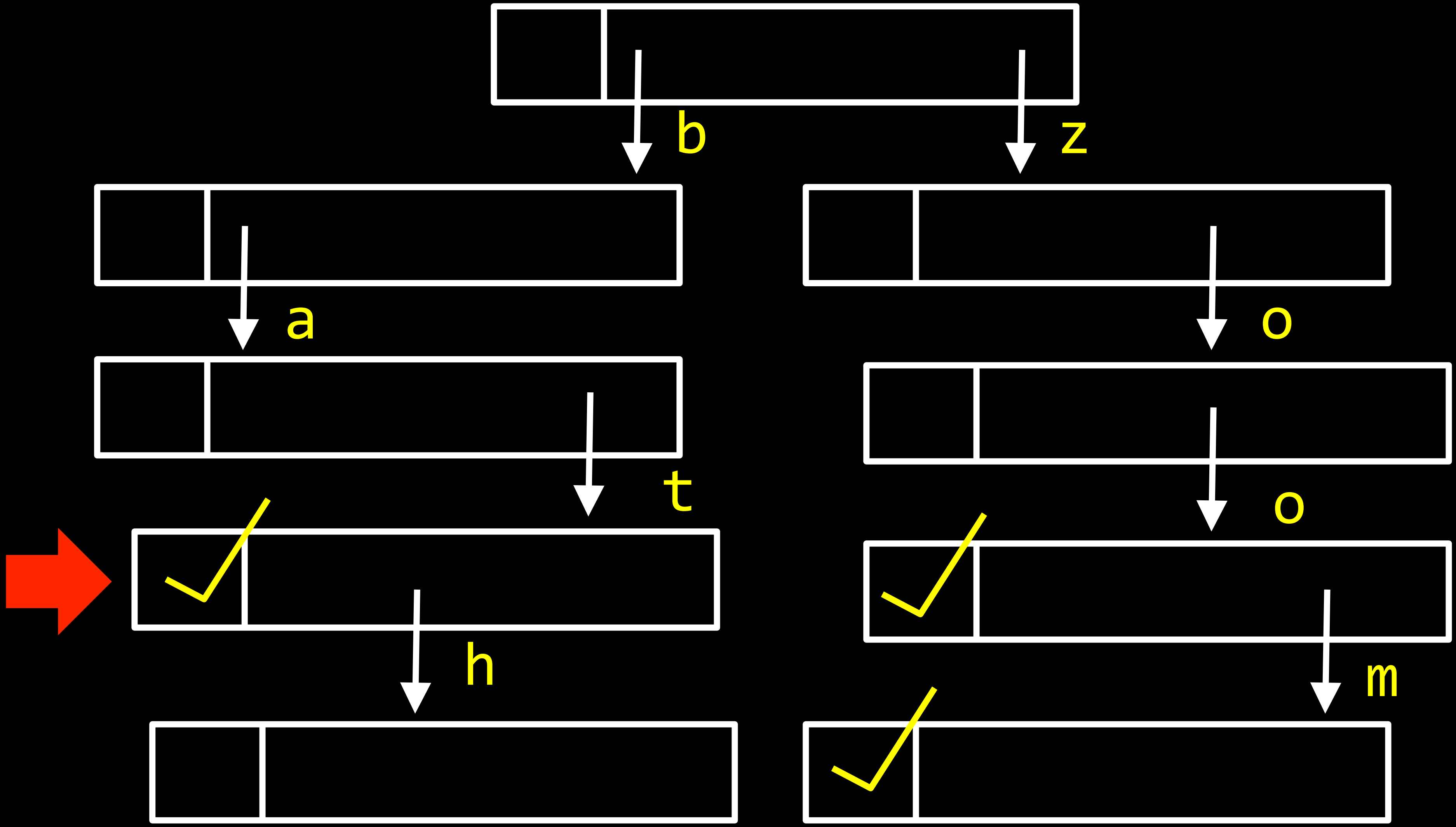

b
z
a
o
t
o
m

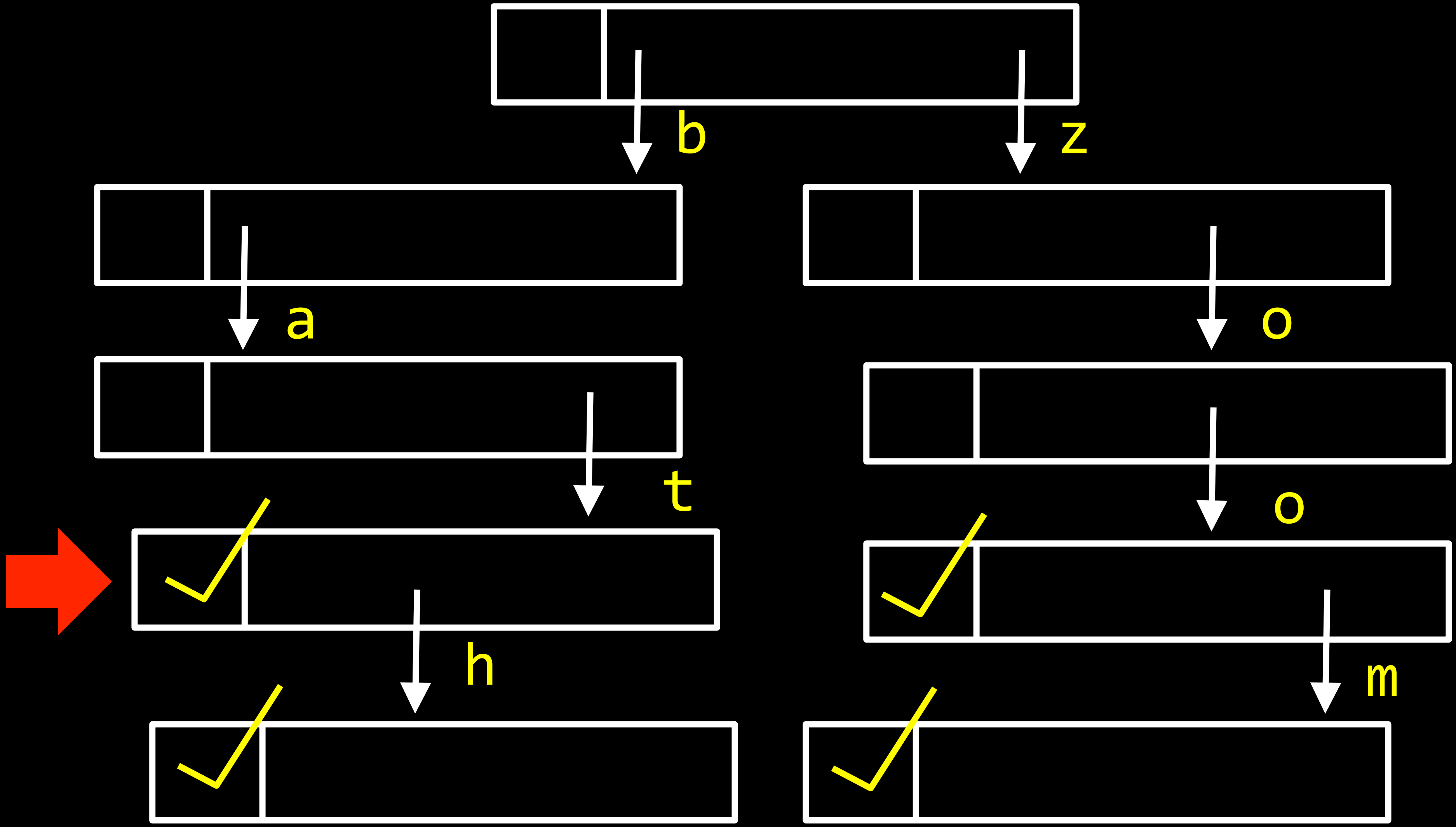

b
z
a
o
t
o
m


b
z
a
o
t
o
m

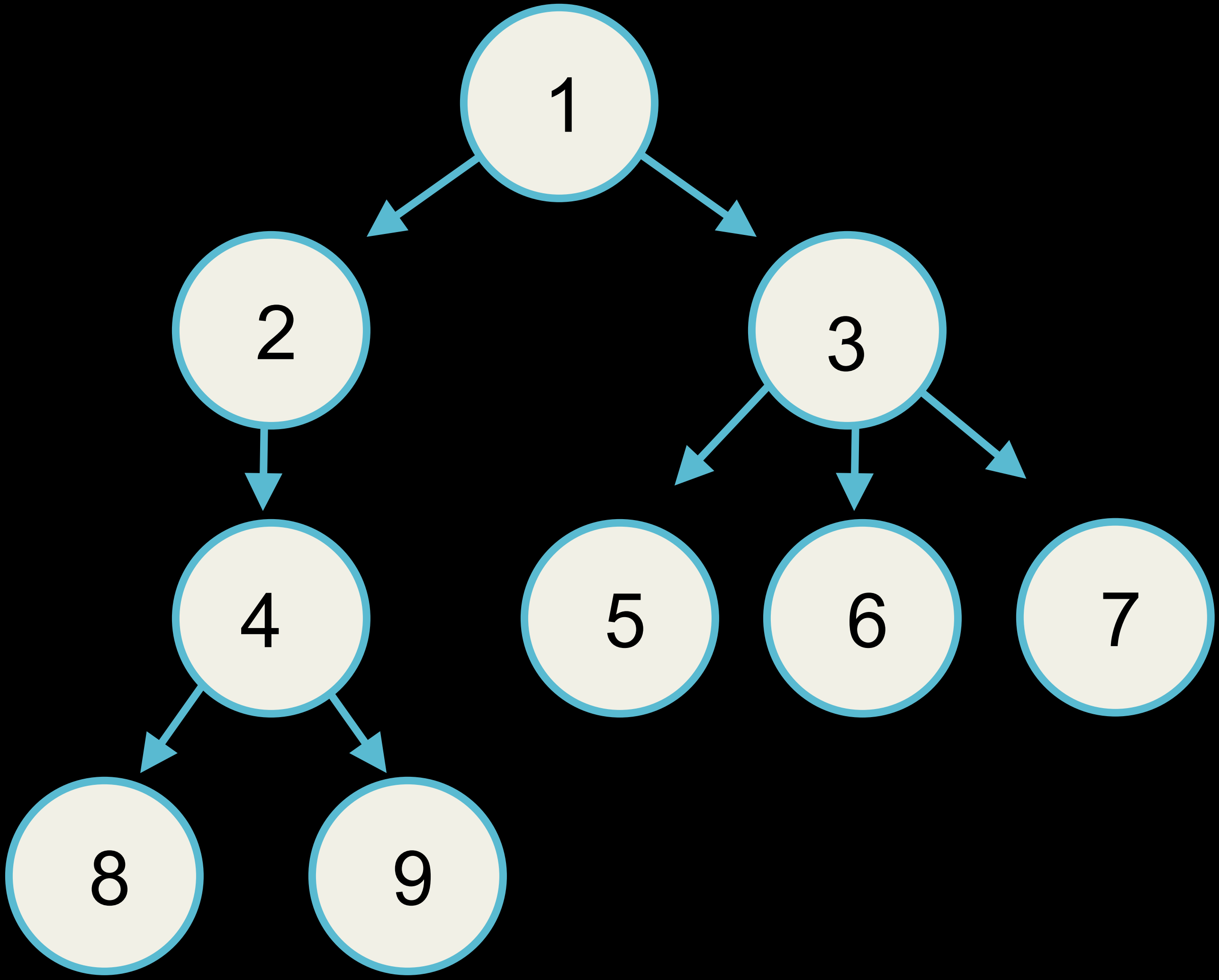

b
z
a
o
t
o
m

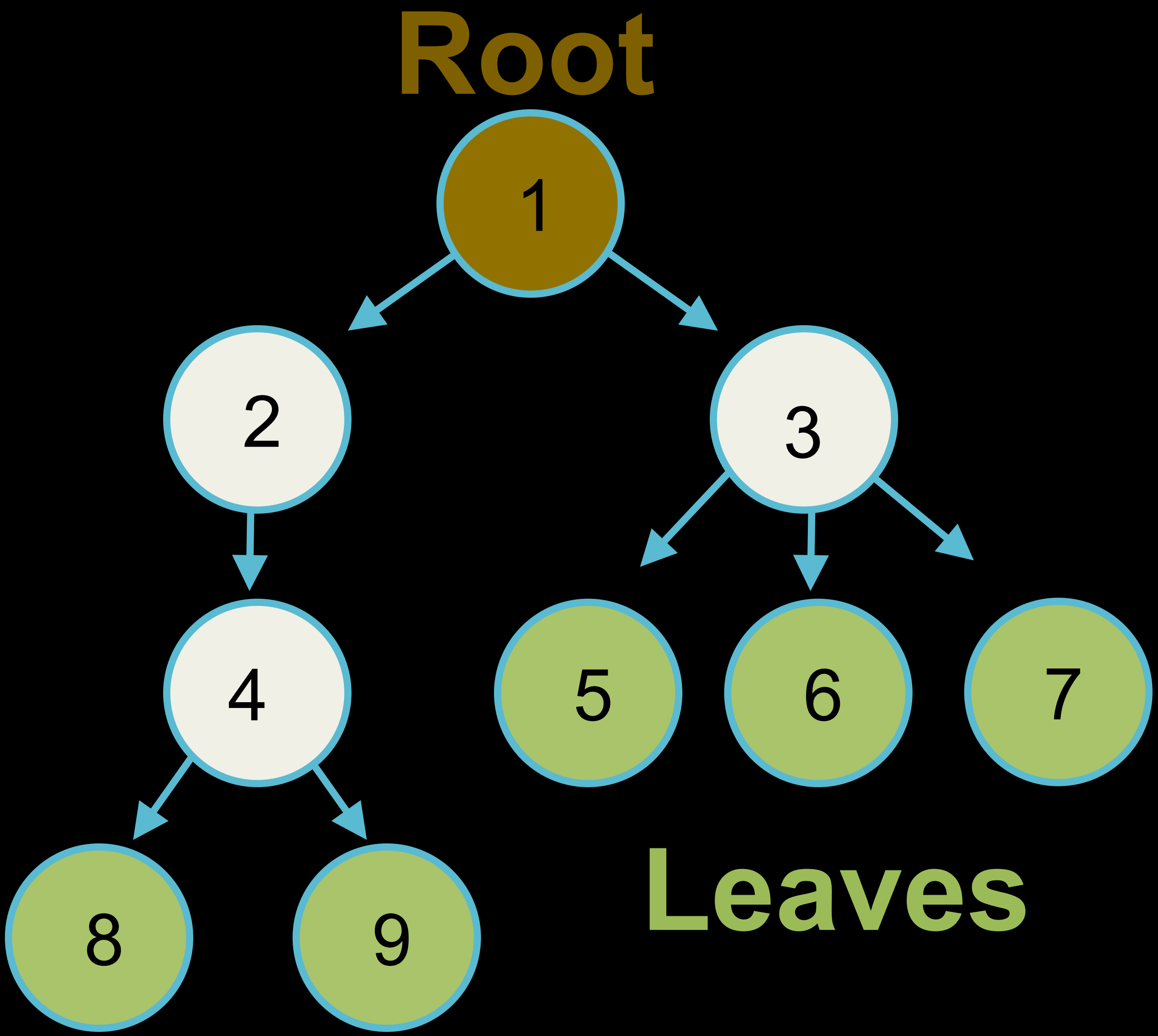

b
z
a
o
t
o
m


b
z
a
o
t
o
h
m

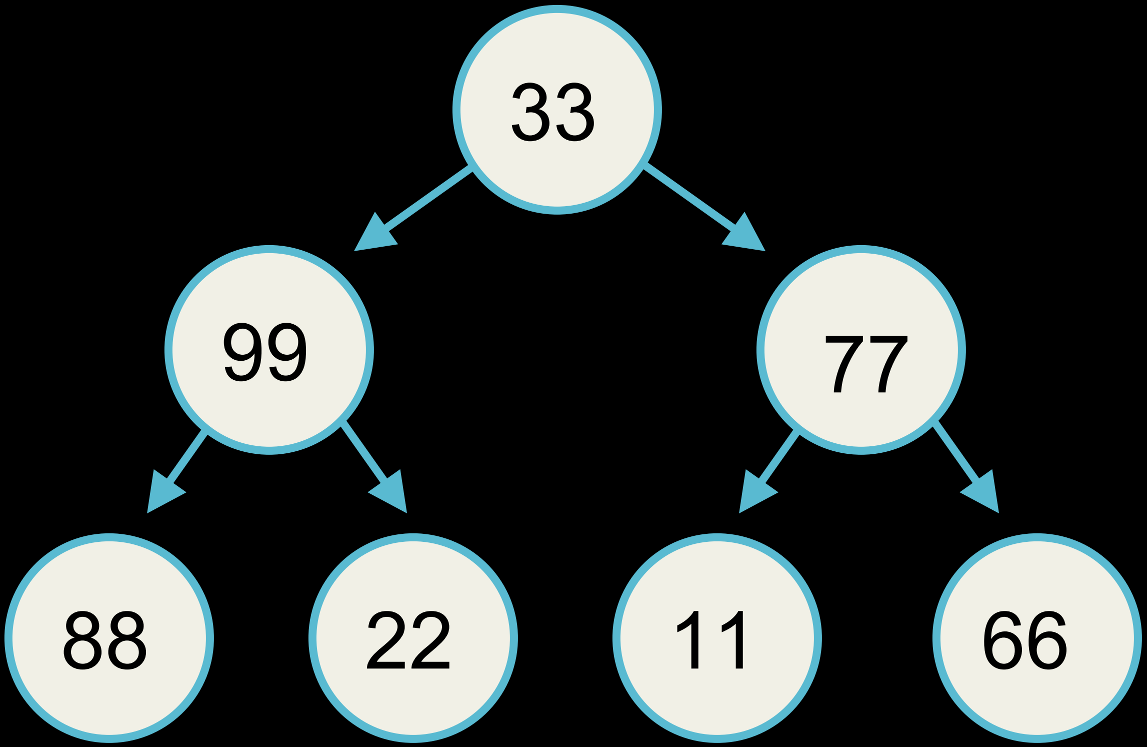

b
z
a
o
t
o
h
m

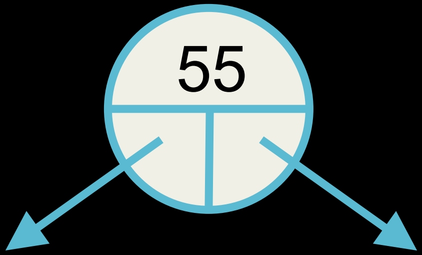
Tree




Root
1
2
3
4
5
6
7
8
9
Leaves

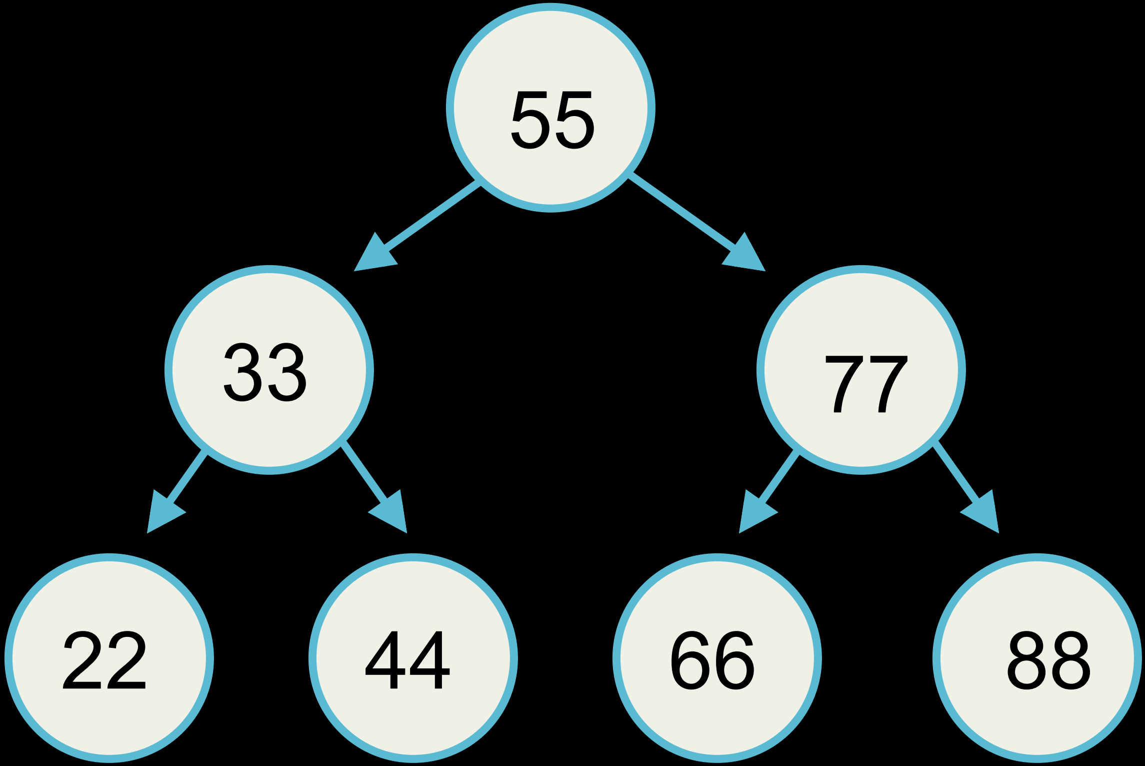
Binary Tree





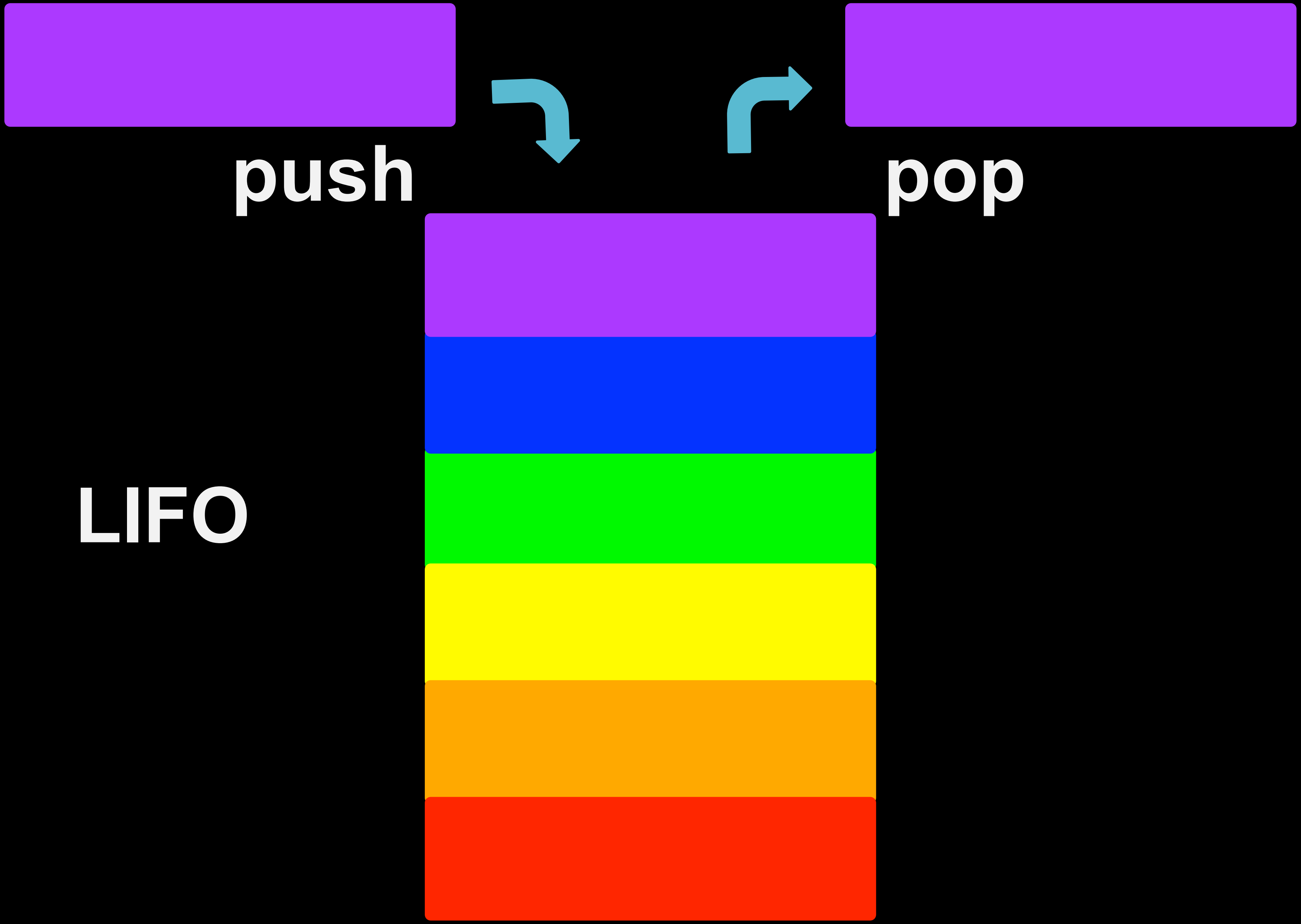
```
typedef struct node
{
      int n;
      struct node* left;
      struct node*
right;
}
node;
```

Binary Search Tree



```
bool search(node* root, int val)
{
    if root is NULL
        return false.

    if root->n is val
        return true.

    if val is less than root->n
        search left child

    if val is greater than root->n
        search right child
}
```

Stacks



LIFO

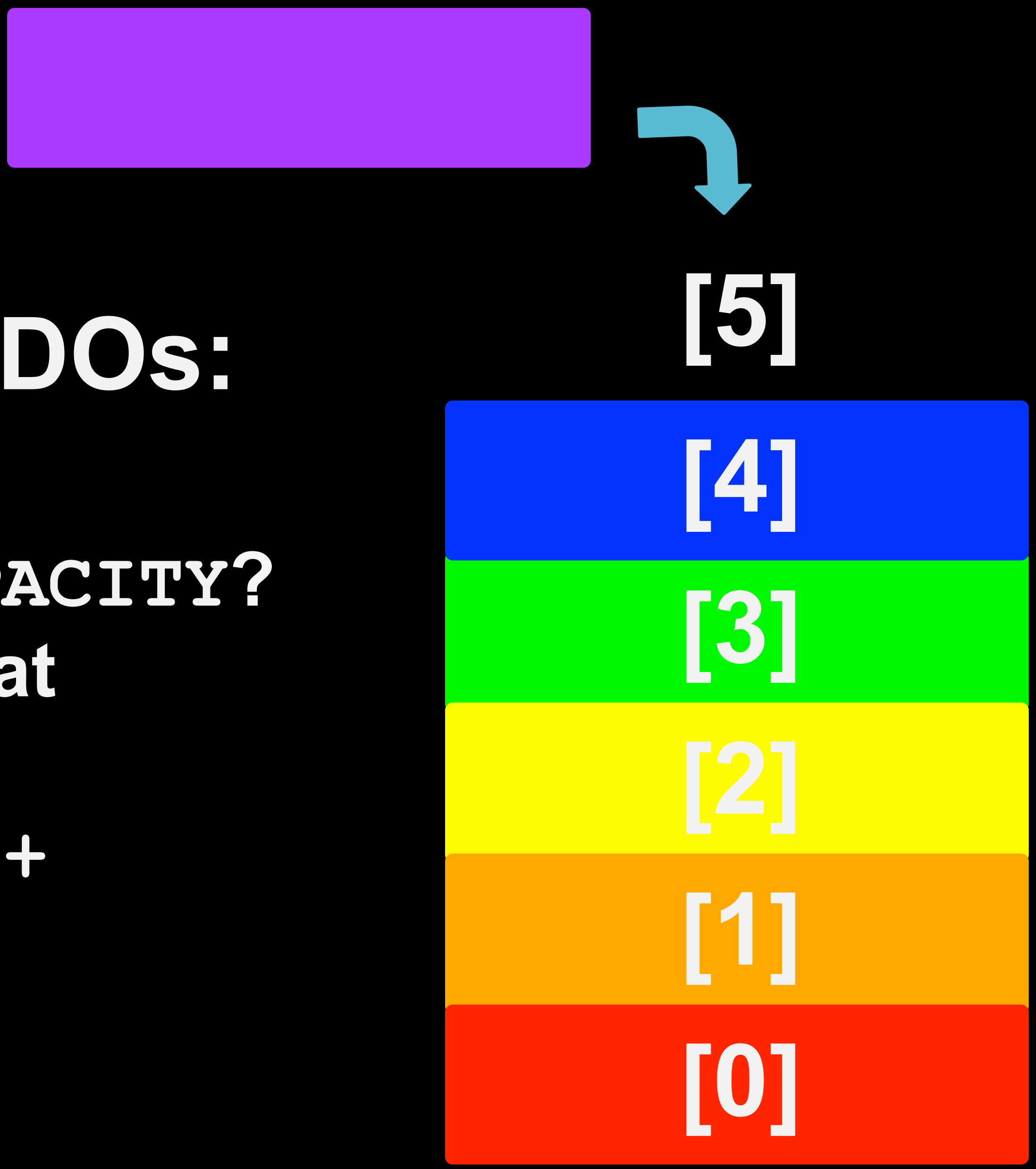
```
typedef struct
{
   char* strings[CAPACITY];
   int size;
}
stack;
```

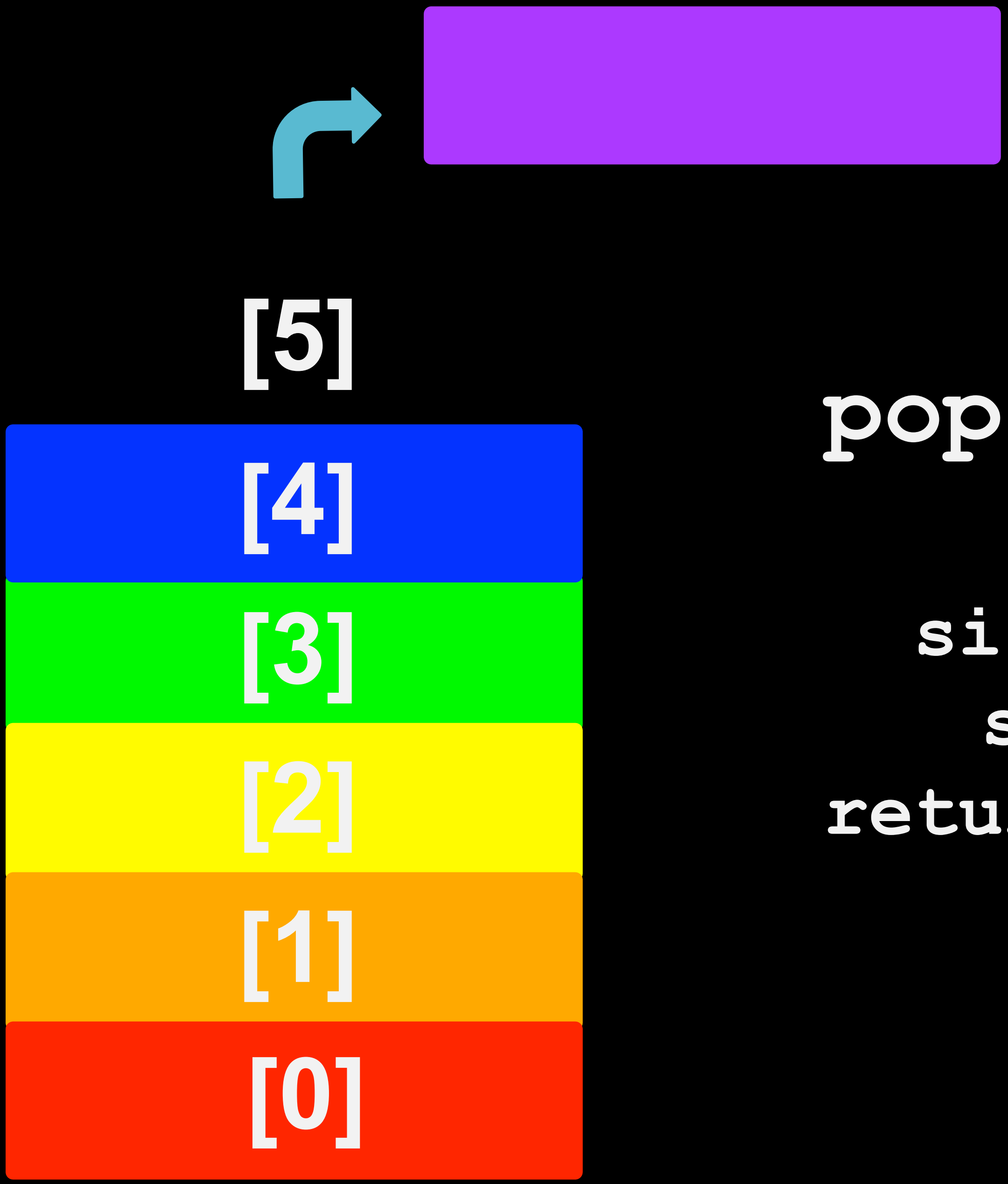
`push` TODOs:

`size < CAPACITY?`

store element at `[size]`

`size++`



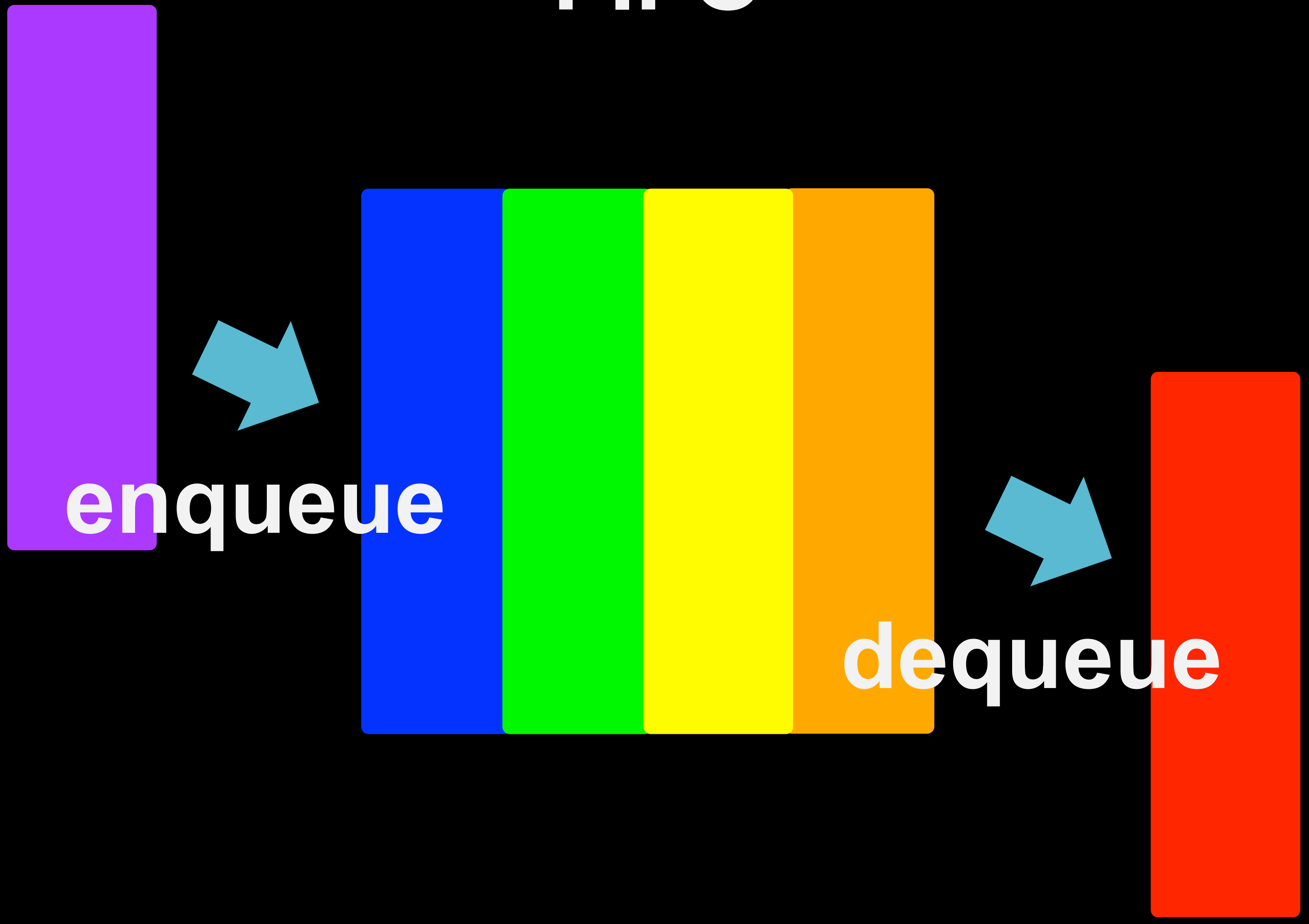


pop TODOs:

```
size > 0?
size--
return [size]
```

Queues

FIFO



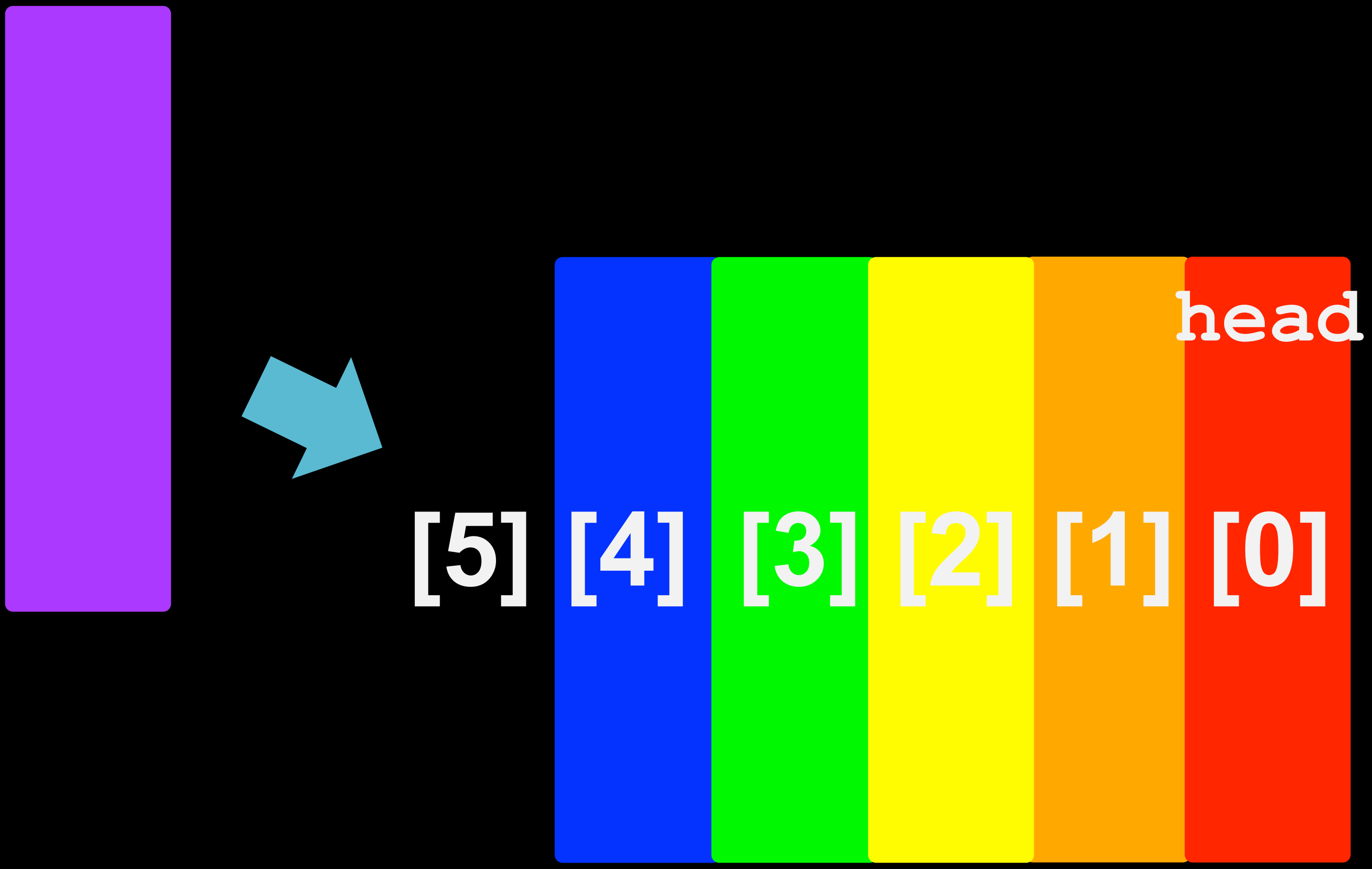
```
typedef struct
{
    int head;
    char* strings[CAPACITY];
    int size;
}
queue;
```

Enqueue TODOs:

`size < CAPACITY?`

store at tail

`size++`



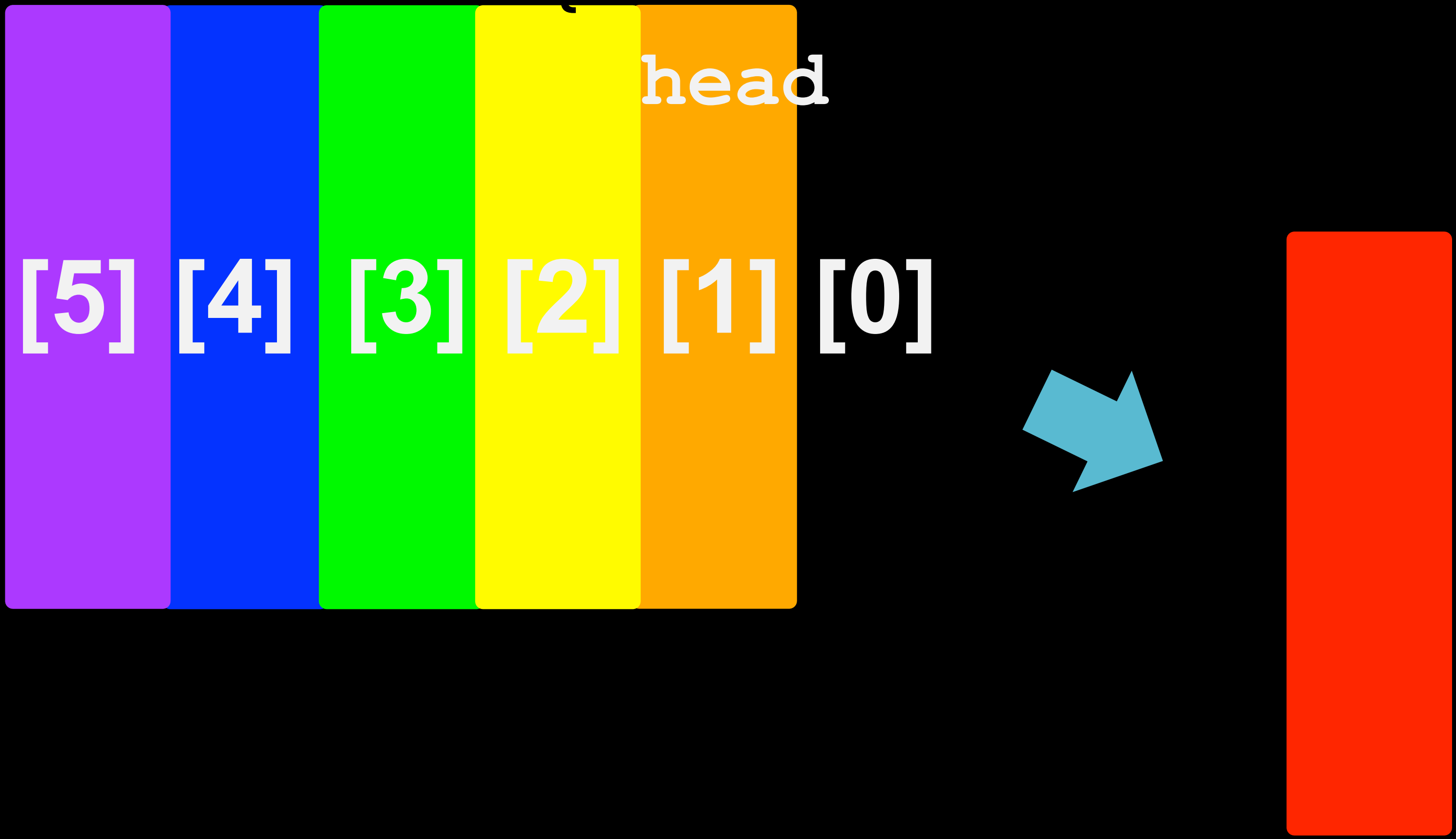
Dequeue TODOs:

`size > 0?`

move `head`

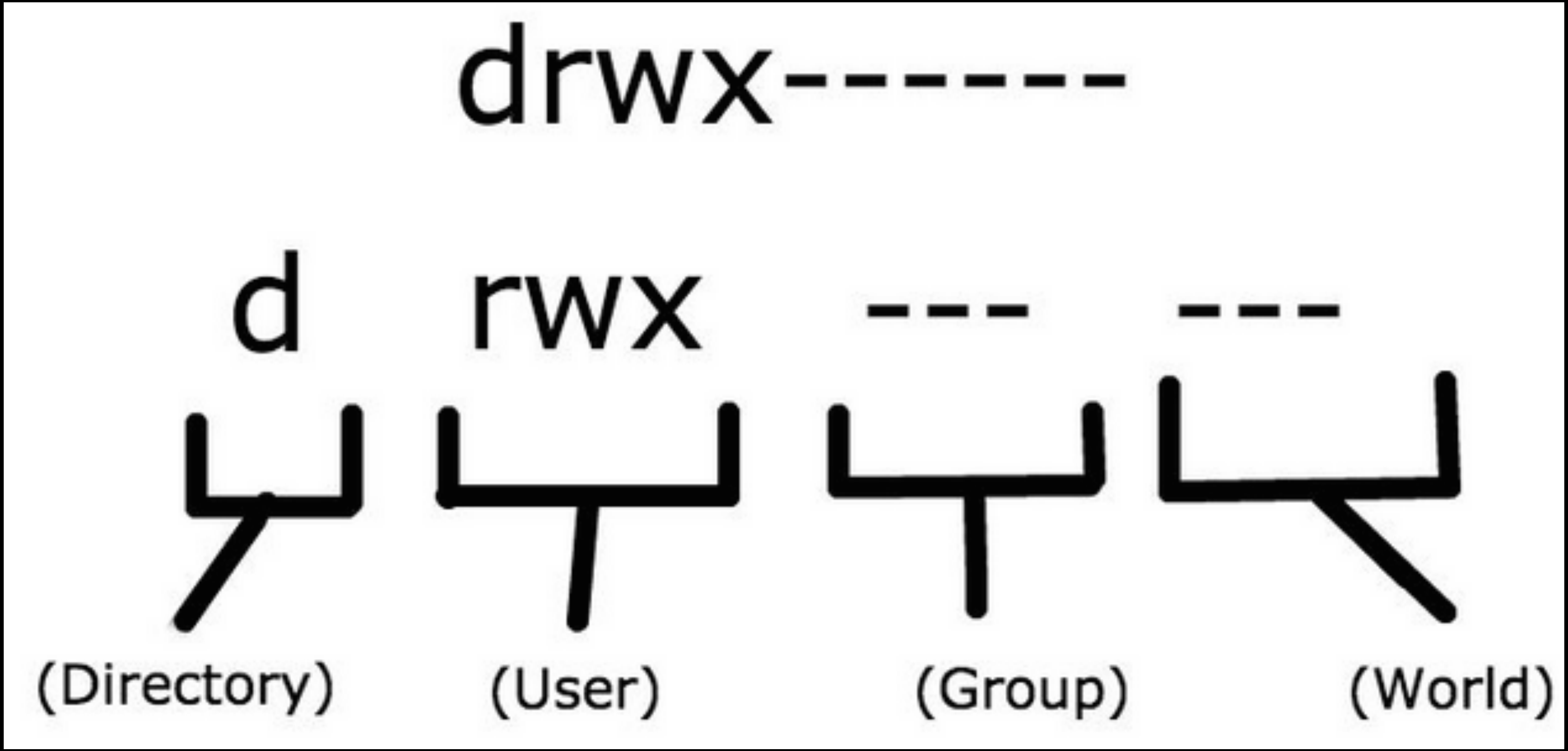
`size--`

return element



chmod

- Unix system call to change file permissions
- `ls -l` : see file permissions
- chmod - - - (each – is from 0-7)
- `r` : readable : 4
- `w` : writeable : 2
- `x` : executable : 1


drwx------
d
rwx

(Directory)
(User)
(Group)
(World)

- chmod group+permissions
    - + adds permissions
    - - takes away permissions
- u: user or owner
- g: group
- o: others

Example

- rwx --- --- can also be represented as 700
- chmod 444 file would give what permissions?
    - What’s another way we could write this?

Example

- `rwx --- ---` can also be represented as 700
- `chmod 444 file` would give what permissions?
  - Readable to everyone!
  - Could also do `chmod a+r file`

Translations

- chmod 555
- chmod u+x
- chmod 640

Translations

- chmod 555
  - Chmod a+rx
  - Gives everyone read and execute access
- chmod u+x
  - Chmod 100
  - Gives the owner execute access
- chmod 640
  - Chmod u+rw, chmod g+r
  - Gives owner read and write permissions
  - Gives group read permission

Common cases

- `chmod 711 directory`: Use for any directory
- `chmod  644 file.txt`: Use for any non-PHP file you create
- `chmod 600 file.php: Use for PHP files`

TCP/IP

- Transmission Control Protocol/Internet Protocol
- Gives a set of standards that govern how data should be packetized, transmitted, routed and received
    - Increases chances the data will get where you want it to!

Ports

- Need to tell our end destination what type of data is in the packet; packets might be routed in various ways/paths
  - 21: FTP: File transfer protocol
  - 25: SMTP: Email
  - 53: DNS: Domain Name System
    - What is the IP address of a domain name?
  - 80: HTTP: Webpage
  - 443: HTTPS: Secure webpage

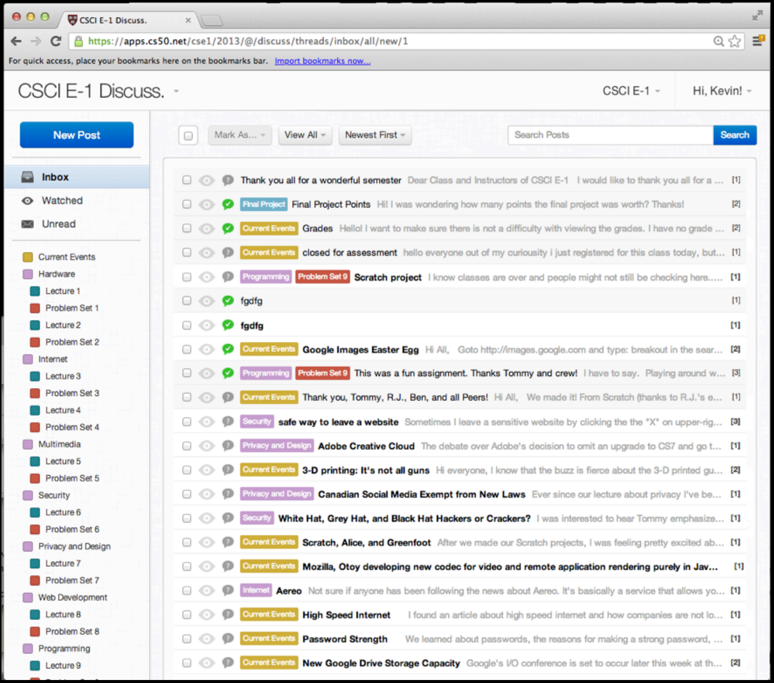
HTML & CSS

- HyperText Markup Language
  - Practice and experiment!
- Best practices:
  - Close all your tags!
  - Validate your page with W3 Validator
  - Separate markup (HTML) and style (CSS)
    - MVC paradigm to come!

CSS

- Instead of tags, CSS uses selectors
    - Match tags with attributes
- Selectors can be
    - id : unique
        - #id in a CSS file
    - class: can refer to multiple blocks
        - .class in a CSS file

HTTP



CSCI E-1 Discuss.

https://apps.cs50.net/cse1/2013/@/discuss/threads/inbox/all/new/1

For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

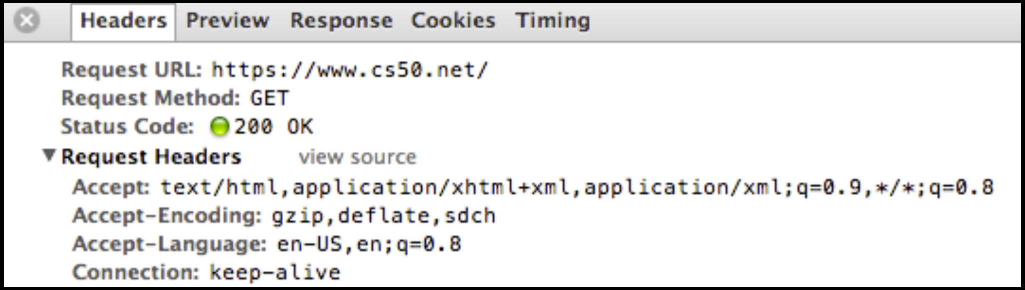
CSCI E-1 Discuss. CSCI E-1 Hi, Kevin!

New Post | Mark As... | View All | Newest First | Search Posts | Search

- Inbox
- Watched
- Unread

- Current Events
- Hardware
  - Lecture 1
  - Problem Set 1
  - Lecture 2
  - Problem Set 2
- Internet
  - Lecture 3
  - Problem Set 3
  - Lecture 4
  - Problem Set 4
- Multimedia
  - Lecture 5
  - Problem Set 5
- Security
  - Lecture 6
  - Problem Set 6
- Privacy and Design
  - Lecture 7
  - Problem Set 7
- Web Development
  - Lecture 8
  - Problem Set 8
- Programming
  - Lecture 9

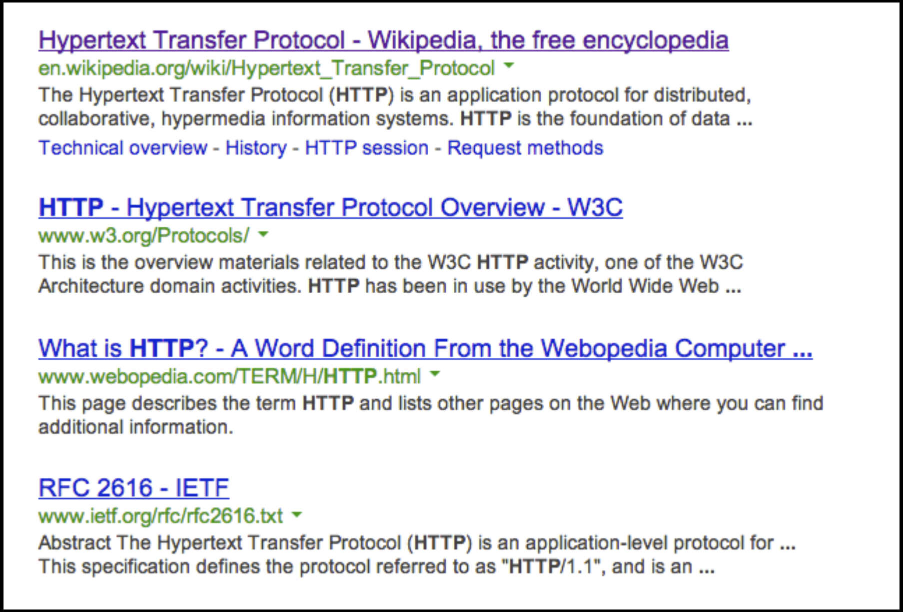
	Thank you all for a wonderful semester	Dear Class and Instructors of CSCI E-1 I would like to thank you all for a ...	[1]
Final Project	Final Project Points	Hi! I was wondering how many points the final project was worth? Thanks!	[2]
Current Events	Grades	Hello! I want to make sure there is not a difficulty with viewing the grades. I have no grade ...	[2]
Current Events	closed for assessment	hello everyone out of my curiousity i just registered for this class today, but...	[1]
Programming, Problem Set 9	Scratch project	I know classes are over and people might not still be checking here.....	[1]
	fgdfg		[1]
	fgdfg		[1]
Current Events	Google Images Easter Egg	Hi All, Goto http://images.google.com and type: breakout in the sear...	[2]
Programming, Problem Set 9	This was a fun assignment. Thanks Tommy and crew!	I have to say. Playing around w...	[3]
Current Events	Thank you, Tommy, R.J., Ben, and all Peers!	Hi All, We made it! From Scratch (thanks to R.J.'s e...	[1]
Security	safe way to leave a website	Sometimes I leave a sensitive website by clicking the the "X" on upper-rig...	[3]
Privacy and Design	Adobe Creative Cloud	The debate over Adobe's decision to omit an upgrade to CS7 and go t...	[1]
Current Events	3-D printing: It's not all guns	Hi everyone, I know that the buzz is fierce about the 3-D printed gu...	[2]
Privacy and Design	Canadian Social Media Exempt from New Laws	Ever since our lecture about privacy I've be...	[1]
Security	White Hat, Grey Hat, and Black Hat Hackers or Crackers?	I was interested to hear Tommy emphasize...	[1]
Current Events	Scratch, Alice, and Greenfoot	After we made our Scratch projects, I was feeling pretty excited ab...	[1]
Current Events	Mozilla, Otoy developing new codec for video and remote application rendering purely in Jav...		[1]
Internet	Aereo	Not sure if anyone has been following the news about Aereo. It's basically a service that allows yo...	[1]
Current Events	High Speed Internet	I found an article about high speed internet and how companies are not lo...	[1]
Current Events	Password Strength	We learned about passwords, the reasons for making a strong password, ...	[1]
Current Events	New Google Drive Storage Capacity	Google's I/O conference is set to occur later this week at th...	[2]



```
Request URL: https://www.cs50.net/
Request Method: GET
Status Code: ● 200 OK
▼Request Headers      view source
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
  Accept-Encoding: gzip,deflate,sdch
  Accept-Language: en-US,en;q=0.8
  Connection: keep-alive
```

HyperText Transfer Protocol

Hyper**T**ext

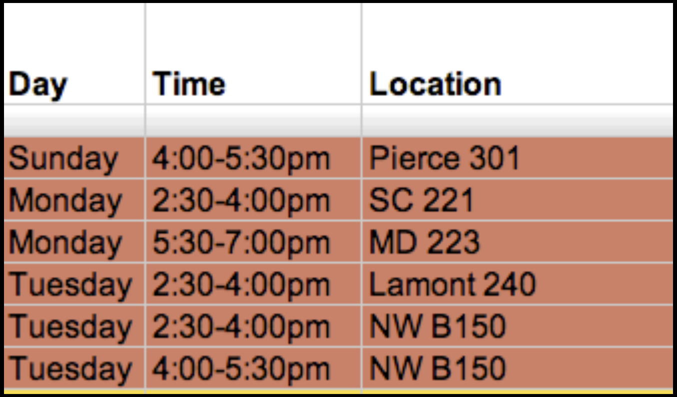


Hypertext Transfer Protocol - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
The Hypertext Transfer Protocol (**HTTP**) is an application protocol for distributed, collaborative, hypermedia information systems. **HTTP** is the foundation of data ...
Technical overview - History - HTTP session - Request methods

HTTP - Hypertext Transfer Protocol Overview - W3C
www.w3.org/Protocols/
This is the overview materials related to the W3C **HTTP** activity, one of the W3C Architecture domain activities. **HTTP** has been in use by the World Wide Web ...

What is **HTTP**? - A Word Definition From the Webopedia Computer ...
www.webopedia.com/TERM/H/**HTTP**.html
This page describes the term **HTTP** and lists other pages on the Web where you can find additional information.

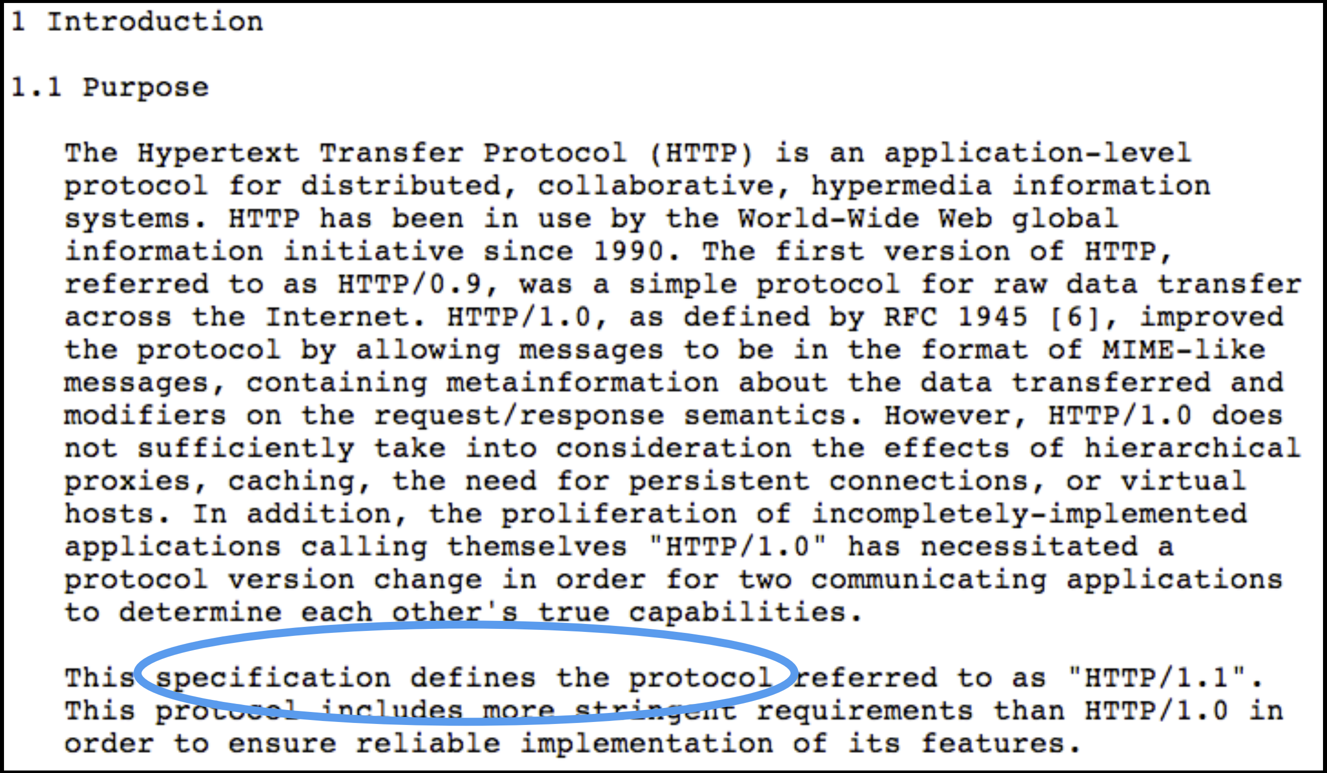
RFC 2616 - IETF
www.ietf.org/rfc/rfc2616.txt
Abstract The Hypertext Transfer Protocol (**HTTP**) is an application-level protocol for ... This specification defines the protocol referred to as "**HTTP**/1.1", and is an ...



Day	Time	Location
Sunday	4:00-5:30pm	Pierce 301
Monday	2:30-4:00pm	SC 221
Monday	5:30-7:00pm	MD 223
Tuesday	2:30-4:00pm	Lamont 240
Tuesday	2:30-4:00pm	NW B150
Tuesday	4:00-5:30pm	NW B150

```
Check out
<a href="https://www.cs50.net">this</a>
really cool website!
```

Transfer Protocol



```
1 Introduction

1.1 Purpose

   The Hypertext Transfer Protocol (HTTP) is an application-level
   protocol for distributed, collaborative, hypermedia information
   systems. HTTP has been in use by the World-Wide Web global
   information initiative since 1990. The first version of HTTP,
   referred to as HTTP/0.9, was a simple protocol for raw data transfer
   across the Internet. HTTP/1.0, as defined by RFC 1945 [6], improved
   the protocol by allowing messages to be in the format of MIME-like
   messages, containing metainformation about the data transferred and
   modifiers on the request/response semantics. However, HTTP/1.0 does
   not sufficiently take into consideration the effects of hierarchical
   proxies, caching, the need for persistent connections, or virtual
   hosts. In addition, the proliferation of incompletely-implemented
   applications calling themselves "HTTP/1.0" has necessitated a
   protocol version change in order for two communicating applications
   to determine each other's true capabilities.

   This specification defines the protocol referred to as "HTTP/1.1".
   This protocol includes more stringent requirements than HTTP/1.0 in
   order to ensure reliable implementation of its features.
```

Introduction to the HTTP specification, from http://www.ietf.org/rfc/rfc2616.txt.

An Example Request

```
GET / HTTP/1.1
User-Agent: curl/7.24.0
Host: www.apple.com
<name>: <value>
```

Key:

```
MethodRequest URI
Protocol Version
field name
field value
```

An Example Response

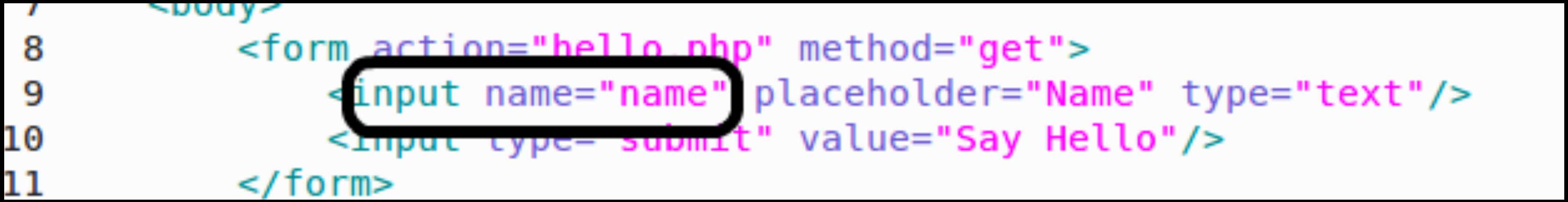
```
HTTP/1.1 200 OK
Server: Apache
Content-Type: text/html; charset=UTF-8
Server: Apache
Content-Length: 16286
Connection: keep-alive
```

Key:

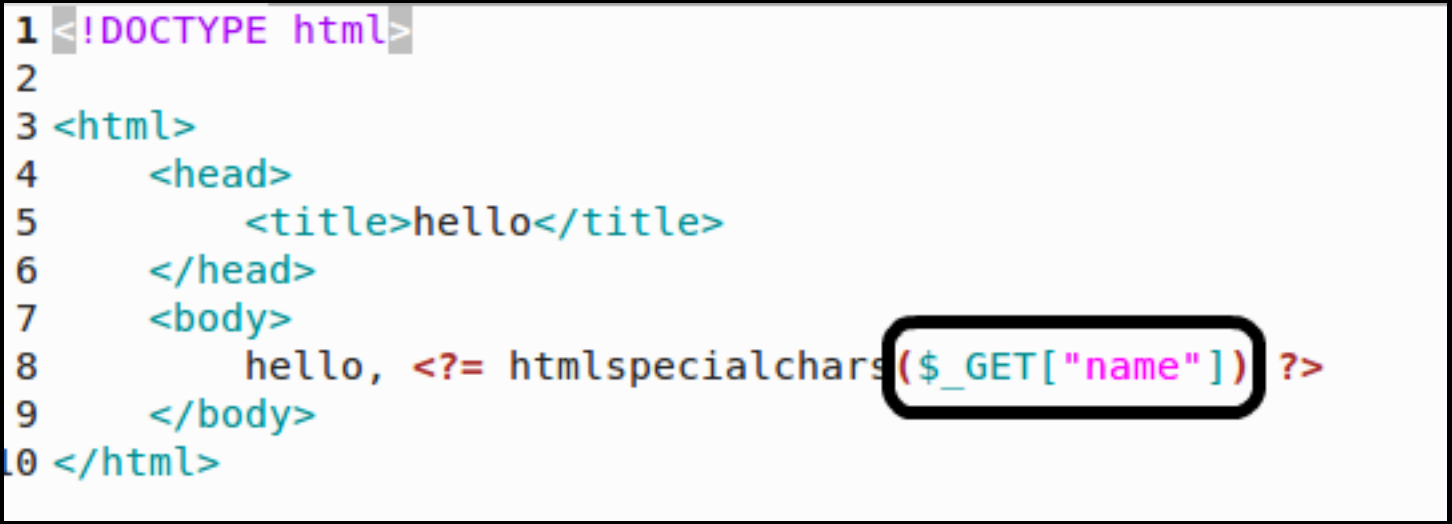
Status Code
Protocol Version
field name
field value

PHP

- Think back to pset6 and hello.html!



```
    <body>
        <form action="hello.php" method="get">
            <input name="name" placeholder="Name" type="text"/>
            <input type="submit" value="Say Hello"/>
        </form>
```



```
<!DOCTYPE html>

<html>
    <head>
        <title>hello</title>
    </head>
    <body>
        hello, <?= htmlspecialchars($_GET["name"]) ?>
    </body>
</html>
```

So what is PHP?

- PHP Hypertext Preprocessor
- Server side scripting language
  - Programming language because it has logic (loops, condition, etc)
- Can combine with HTML
- Allows us to create dynamic webpages!
  - Can incorporate HTML and PHP in the same file or pass information from separate files

Crash Course!

- Declaring variables
  - No need to specify type!
  - Loosely or dynamically typed -> determined at runtime
    - `$var = 3;`
- Arrays can be associative!
  - `$array = [key1 => value1, key2 => value2]`
  - However, keys are optional!
    - `$array = [10, 20, 30]`
    - `Index would be just like in C!`
    - `$array[1] = 20;`

Crash Course!

- == versus ===
  - == checks for equality after type juggling
  - === checks if they are equal and of the same type
- foreach loop
  - Way to iterate over arrays
  - If there are no keys

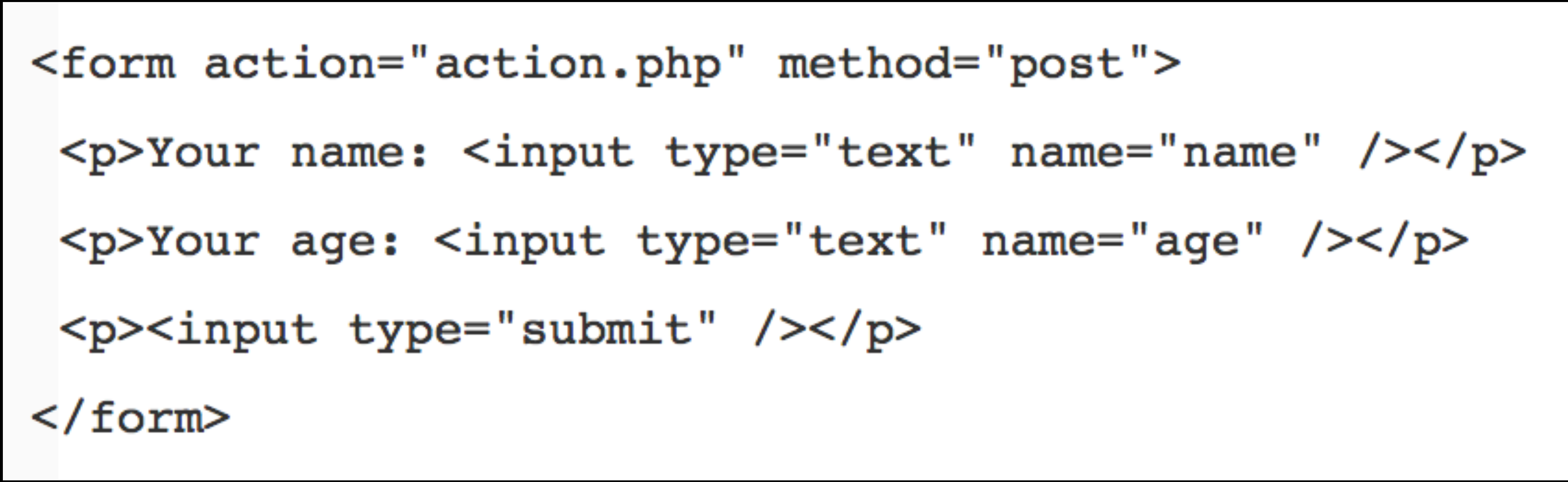
```
foreach ($array as $value) {
    //do this
}
```

  - If there are keys

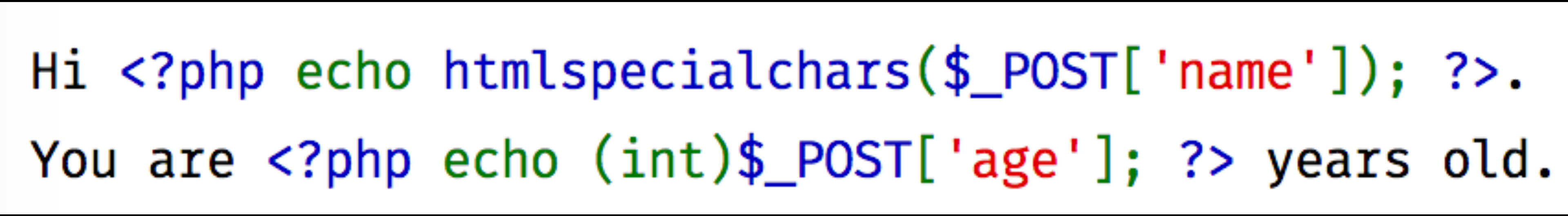
```
foreach ($array as $key => $value) {
    //do this
}
```

Two ways to pass info

- GET
  - Information passed via URL
- POST
  - Passes data in the HTTP message body
  - Consider the data to be “hidden” compared to GET requests



```
<form action="action.php" method="post">
 <p>Your name: <input type="text" name="name" /></p>
 <p>Your age: <input type="text" name="age" /></p>
 <p><input type="submit" /></p>
</form>
```



```
Hi <?php echo htmlspecialchars($_POST['name']); ?>.
You are <?php echo (int)$_POST['age']; ?> years old.
```

http://php.net/manual/en/tutorial.forms.php

SQL

- UPDATE
  - Update data in a database table
- INSERT INTO
  - Insert certain values into a table
- SELECT
  - Select values to view
- DELETE
  - Delete from table

JavaScript

"JavaScript is the best programming language currently in existence. Other people will try to tell you otherwise. They are wrong."

Thomas MacWilliam
Head Teaching Fellow, 2012

Hello World

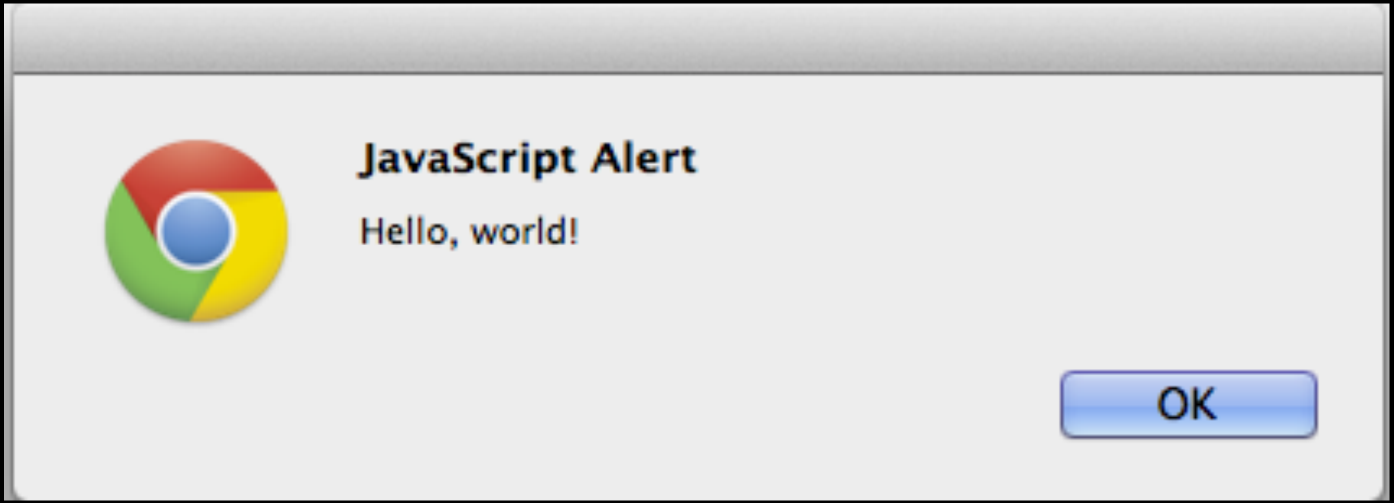
index.html:

```
<!DOCTYPE html>
        <html>
                <head>
                        <script type="text/javascript"
src="hello.js">
            </script>
                        <title>Hello, world!</title>
                </head>
                <body>
                        Body HTML here
                </body>
    </html>
```

hello.js:

```
alert("Hello, world!");
```

Hello World



JavaScript Alert

Hello, world!

Variable Declarations

```
var s = "CS50";
var n = 3.14;
var b = true;

...

alert("Type of b: " + typeof(b));
b = "make b a string";
alert("Type of b: " + typeof(b));
```

Loops

```
for(/* init */; /* condition*/; /* update */)
{
   /* code */
}


while(/* condition */)
{
     /* code */
}

do {
   /* code */
} while(/* condition */);
```

Function Declarations

```
function sum(x, y)
{
      return x + y;
}
/* or */

var sum = function(x, y)
{
      return x + y;
}

var sum = sum(3, 5);
alert("3 + 5 = " + sum);
```

Arrays in JavaScript

```
var arr = [];
var arr2 = ["Arrays", "in", "JS"];
var thirdElement = arr2[2];

var arr2len = arr2.length;

var arr3 = [2.3, true, 5];
arr3[2] = "not a number";
arr3[100] = "legit";
```

Objects in JavaScript (1)

```
var emptyObject = {};

emptyObject["newProperty"] = "newValue";
emptyObject.otherNewProperty =
"otherNewValue";

alert(emptyObject.otherNewProperty);
alert(emptyObject["otherNewProperty"]);
```

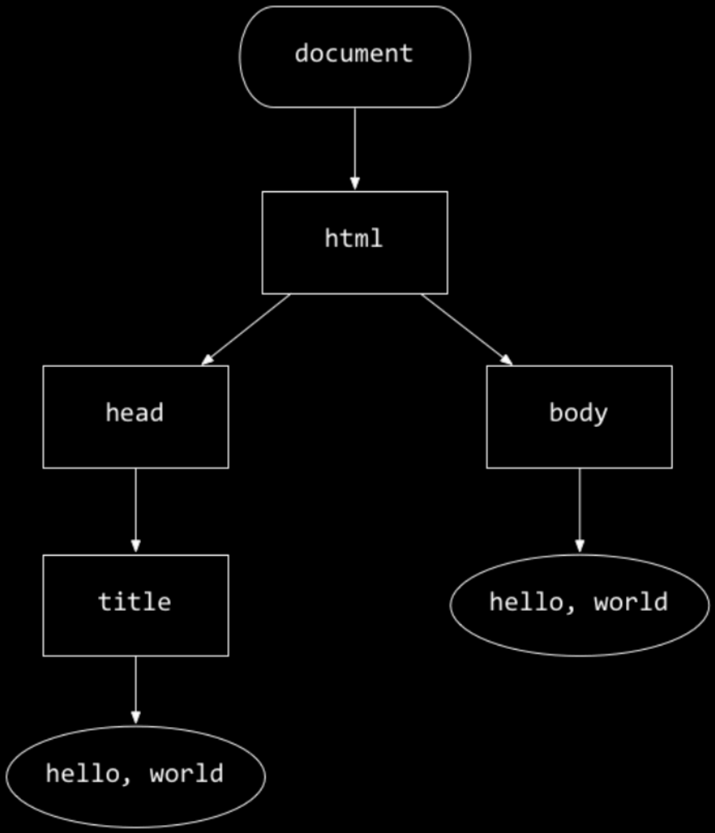
Objects in JavaScript (2)

```
var CS50 = {
          "course": "CS50",
     "instructor": "David J. Malan '99",
     "tfs": ["R.J.", "Ben", "Pat", "Chris"],
     "psets": 8,
     "taped": true
};
```

Objects in JavaScript (3)

```
var cottage = [
  {name: "James", house: "Winthrop"},
  {name: "Molly", house: "Cabot"},
  {name: "Carl", house: "Kirkland"}
];

for(var i = 0; i < cottage.length; i++)
{
    alert(cottage[i].name);
}
```

DOM: Document-Object Model



```
<!DOCTYPE html>

<html>
      <head>
            <title>hello, world</
title>
      </head>
      <body>
            hello, world
      </body>
</html>
```

DOM: Document-Object Model (2)

Examples

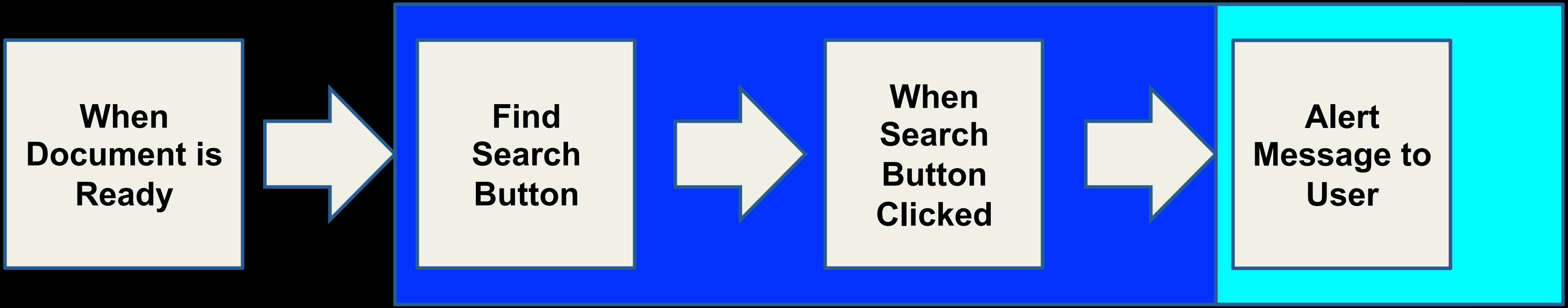
- `document.title`
- `document.body`
- `document.body.innerHTML`

Useful Functions

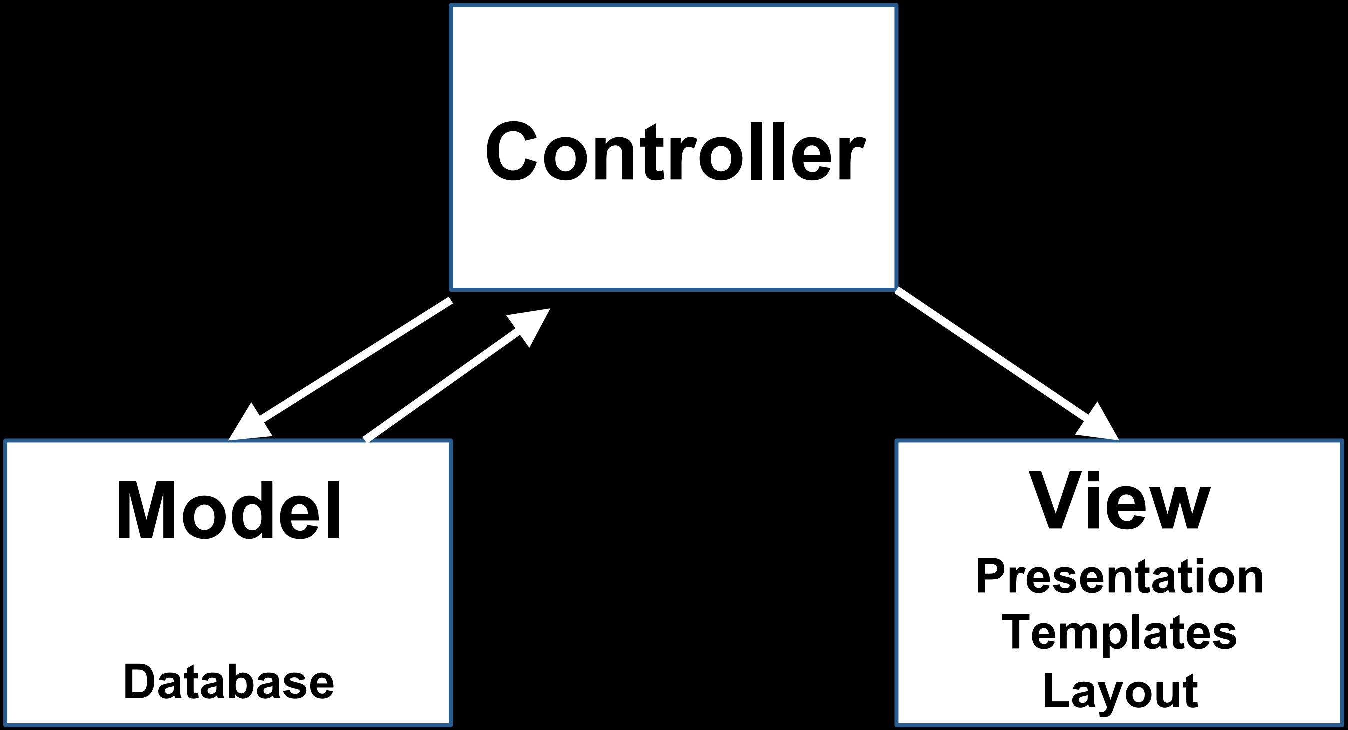
- `document.getElementById(`*`string`*`)`
- `document.getElementsByClassName(`*`string`*`)`
- `document.getElementsByTagName(`*`string`*`)`

JavaScript Events

```
window.onload = function() {
   var searchButton =
      document.getElementById("search_button");

   searchButton.onclick = function() {
            alert("You clicked the search
button");
   }
}
```



MVC



Model-View-Controller

MVC

COMPONENT	FUNCTION	EXAMPLE
Model	- Persistent storage of information - Managing and organizing data	- MySQL database - Data files
View	- Presentation of information to user - User interface	- HTML - Minimal PHP (e.g., for iterating over data to print it out)
Controller	- Handles user requests, gets information from the model	- PHP

C$50 Finance: Model

C$50 Finance: Controller

```
$result = query("SELECT * FROM some_table");

if ($result !== false)
{
        foreach ($result as &$row)
        {
                /* process row of result */
        }   render("some_template", array("data" => $result));
}
else
{
        apologize("Error communicating with database.");
}
```

C$50 Finance: View

some_template.php:

```
<ul>
    <?php foreach($data as $row): ?>
        <li><?= $row["name"] ?></li>
    <?php endforeach ?>
</ul>
```

Tips

- Take practice tests under time constraints
- Get sleep and rest!
- Relax
- Remember to breathe!
- Put time into your cheat sheet
- You got this!

This was section.