{w | w CONTAINS TWICE AS MANY 0's AS 1's}

Ex: 011000

#011000 — FIND a "1"

#011000 — change 1 TO X; MOVE L

#0X1000 — LOOK FOR LEFT EDGE (HASH MARK)

#0X1000 — SCAN RIGHT FOR A ZERO

#0X1000 — change 0 TO a Y

#YX1000 — SCAN RIGHT FOR SECOND ZERO

#YX1000

#YX1000

#XX1000 — change 0 TO a Y; SCAN LEFT TO X

#YX1Y00

#YX1Y00 — FIND a "1"

#YX1Y00 — chg TO X

#YXXY00 — SCAN FOR A Y

#YXXY00 — SCAN RIGHT FOR A ZERO → FIND 'L' FIRST → REJECT

#YXXYX0 — chg TO X; SCAN R FOR SECOND 0

#YXXYYO — chg TO Y

#YXXYYY — SCAN LEFT FOR X

#YXXXYY — SCAN RIGHT for 1

#YXXYYY — NO 1 FOUND; LOOK FOR REMAINING ZEROS

#YXXYYY — SCAN FOR # AND SEE ONLY X's & Y's
See a 0 → reject
See the # → accept

$(q_0, \#) \rightarrow (q_1, \#, R)$

$(q_1, 0) \rightarrow (q_1, 0, R)$ $\quad (q_1, ⊔) \rightarrow (q_6, ⊔, L)$

$(q_1, 1) \rightarrow (q_2, x, L)$

$(q_2, \#) \rightarrow (q_3, \#, R)$

$(q_2, y) \rightarrow (q_3, y, R)$

$(q_2, 0) \rightarrow (q_2, 0, L)$

$(q_2, 1) \rightarrow (q_2, 1, L)$

$(q_3, x \mid y \mid 1) \rightarrow (q_3, \text{same}, R)$

\# 0 1 ⊔

$(q_3, 0) \rightarrow (q_4, y, R)$

$(q_3, ⊔) \rightarrow$ REJECT $\leftrightarrow (q_r, ⊔, L)$ OMITTED

$(q_4, x \mid y \mid 1) \rightarrow (q_4, \ , R)$

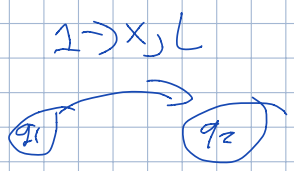
$(q_4, ⊔) \rightarrow$ REJECT

$(q_4, 0) \rightarrow (q_5, y, L)$

$(q_5, 0 \mid 1 \mid y) \rightarrow (q_5, \ , L)$

$(q_5, x) \rightarrow (q_1, x, R)$

M_{2x} { $Q = \{q_0 \ldots q_6\}$,
$\Sigma = \{0, 1, \#\}$,
$\Gamma = \Sigma \cup \{x, y\} \cup \sqcup$
δ = See above
$q_0 = q_0$
$q_{accept} = q \in Q$ is the ACCEPT STATE
$q_{reject} = q \in Q$ is the reject STATE
$q_{accept} \neq q_{reject}$

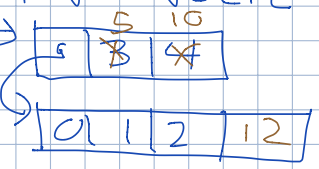


Rust - Collections, Iterators, mutability, and reference counting

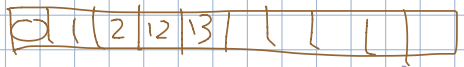
COLLECTIONS

- VECTORS
  - SIZE CAN GROW ON RIGHT
  - COMPOSED OF ANY TYPE OR OBJECT (same type)

```
LET V = VEC![0,1,2];
```




```
V.push(12);
V.push(13);
```




```
LET V2 = VEC!(0;10);
```

- deque
  - FIFO QUEUE
    - Add to one end, pull from other end

- HASHMAP
  - key/value pairs
  - `LET H = HashMap::NEW<STRING, USIZE>();`
  - `h.INSERT("key string", 42);`

ITERATOR:

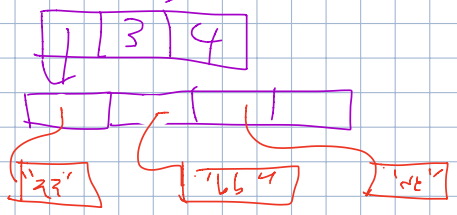
- WALKING THROUGH A COLLECTION

```
LET V = VEC![0,1,2];
FOR i IN V {
}
```

```
LET h = HashMap::new(...)
FOR (k,v) IN h {
}
```

MUTABILITY

```
LET V = VEC!["a", "b", "c"]
LET MUT VR = &V;
```



INTERIOR MUTABICITY

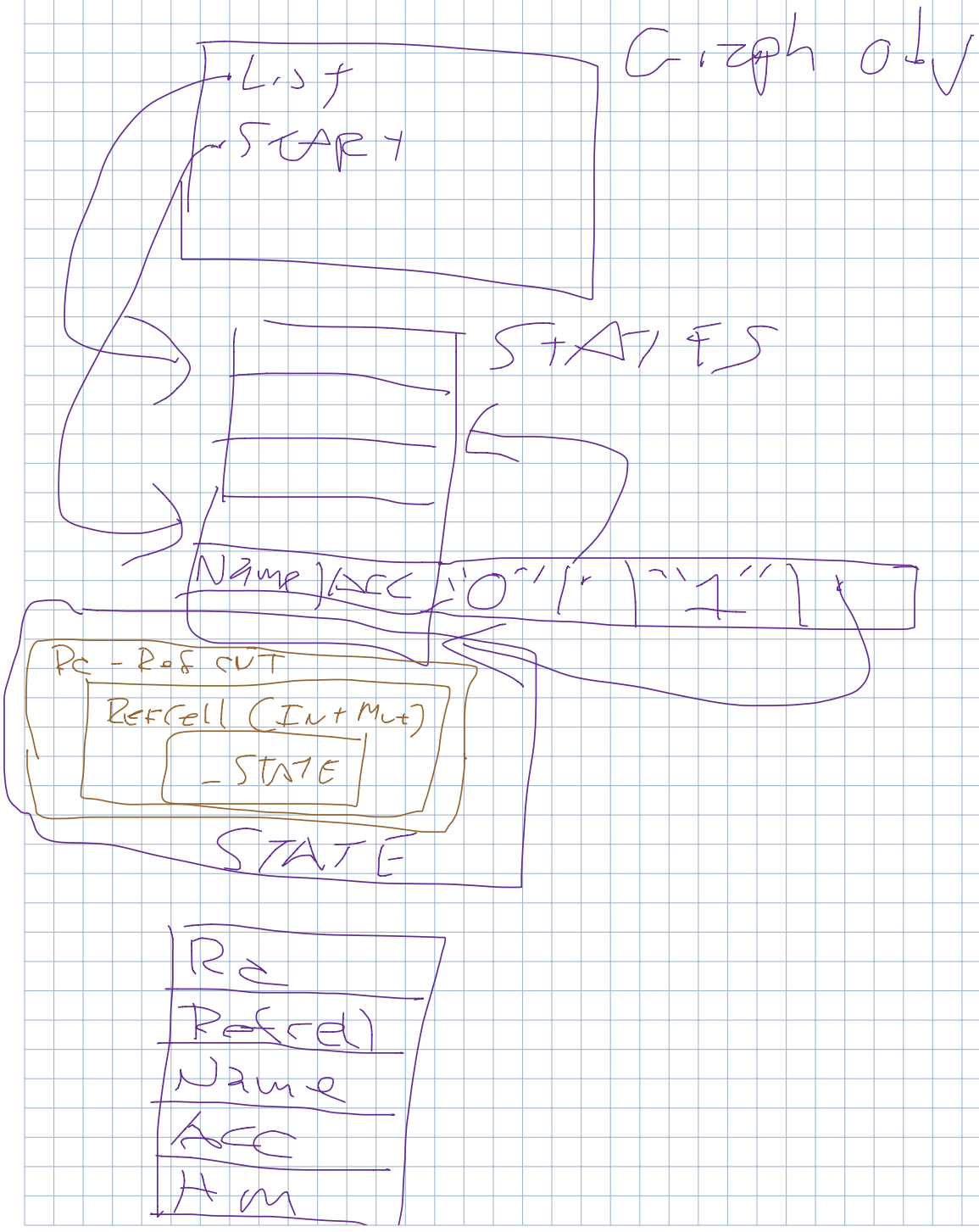
- MUTABLE REFERENCE TO SOMETHING INSIDE AN IMMUTABLE OBJECT

REFERENCE COUNTED REFENCES

- MULTIPLE MUT/IMUT REFERENCES TO SOMETHING

LIST OF STATES

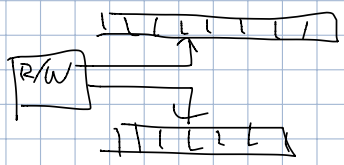
- START STATE → ONE STATE
- MARK STATES AS ACCEPT STATES
- TRANSITION STORED AS REFERENCES



VARIANTS OF T.M's

- MULTIPLE TAPES

$\delta(q_{ph}, \sqcup_i) \Rightarrow (q_{ph+1}, \sqcup_{i+1}, L/R)$



$\delta(q_0, \sqcup^{T_1}, \sqcup^{T_2} \ldots) \Rightarrow$

$(q_{n-1}, \sqcup^{T_0}, L/R/S, \sqcup^{T_2}, L/R/S)$

- Provable that any MULTI-TAPE T.M. CAN BE expressed as a SINGLE TAPE T.M.

- ENUMERATOR
  - 2 TAPES (CR model)
    - ONE READ/WRITE
    - WRITE ONLY TAPE

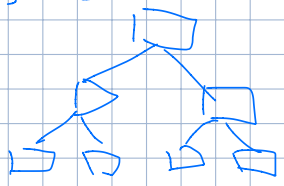
$L = \{a^n b^n c^n \mid n \geq 0\}$

abc
aabbcc
aaabbbccc

Non-Deterministic T.M.

$\delta(q_n, \Sigma_n) \Rightarrow (q_{n+1})$

NFA: $\delta(q_n, \Sigma_n) \rightarrow \{(q_{n+1}), (q_{n+2})\}$



$\delta(q_n, \Gamma_n) \rightarrow \{(q_{n+1}, \Gamma_m, L/R/S), (q_{n+2}, \Gamma_{m+1}, L/R/S)\}$

- Tree
  - At least one branch accepts $\Rightarrow$ accept
- Turing Decidable
  - ALL possible branches halt

$L = \{0^m \mid m \text{ is a compound number}\}$

Compound number has at least 2 factors not including 1
- m is not prime

NTM — is m a compound number
- Select 2 #'s p & q
- Deterministic T.M to see if p & q are factors of
- If so ACCEPT
- Otherwise REJECT