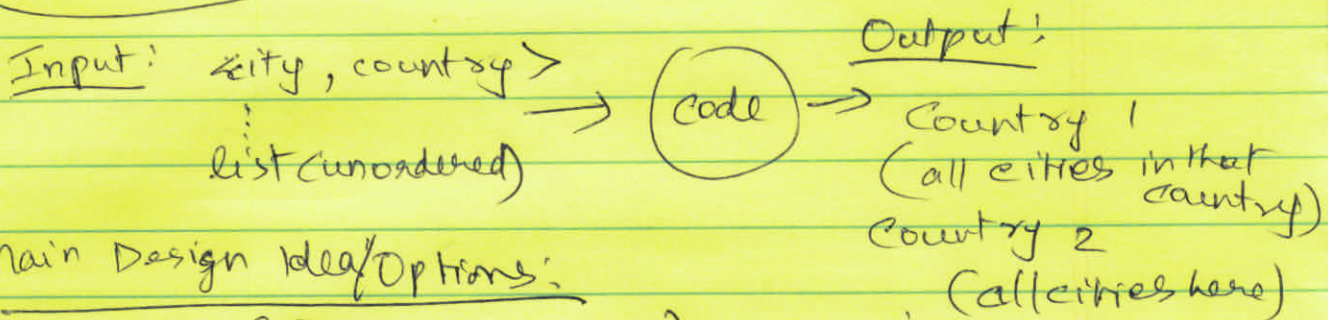


Problem #1 Group Cities By Country



Main Design Idea/Options:

→ Option I: (Best solution)

Sort Input $\langle \text{city, country} \rangle$ list by country field
→ (because this is string sorting they have to write their own comparison function)

→ Any $O(n \lg n)$ sort will be good — merge sort
— quicksort

→ Option II: (Equally good as Option I)

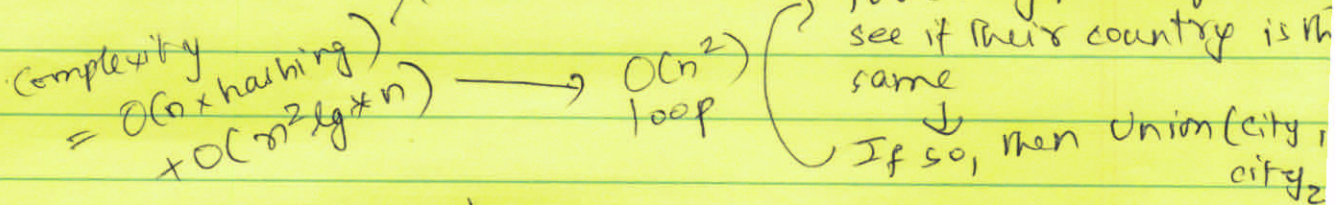
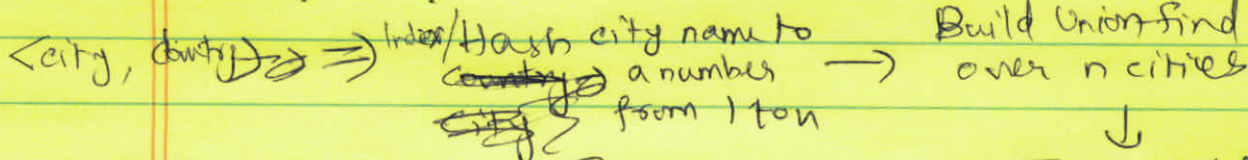
Hash country → key. ~~But then another sort the key values.~~ Then use chaining $O(n + M)$ answer.
~~This is a bad solution because key values can~~

size of hashtable

→ Option III:

Use Union-find to group country names.

~~range over arbitrary range and it is as bad as the Option I, at least~~ ~~sort based algo~~



Bad solution because more complicated than necessary.

Problem #1 Grading rules

Total points } = 50
for prob #1 }

(25)

Design Choice

- Sorting (option I) or ^(Hashing) Optim II → [25]
- Any other design ~~that~~ → [15]
~~is ~~considered~~ ~~as~~ ~~DC~~ ~~design~~~~

(25)

Implementation

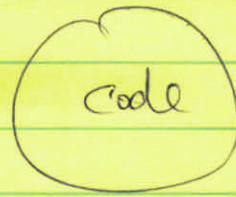
- Code Readability → [5]
- Does the code work (testing)? → [20]

If Report is missing or is totally uninformative, then ~~to~~ reduce 10 points for that

Problem #2 Group cities by Air Routes (50 points)

Input:

$\langle \text{city A, city B} \rangle \rightarrow$
means A & B are
connected by flight



Output:

Eq. Class #1

A, B, C, D

Eq. Class #2

F, X, Y, ...

Goal: If 2 cities have an air route/path between them, then they belong to the same Eq. class.

Design Choices:

\rightarrow Option I: Use Union-find + hashing

eg:

$\langle A, B \rangle$

$\langle C, D \rangle$

$\langle D, A \rangle$

$\langle E, F \rangle$

For # line

in Input

$\langle X, Y \rangle$

do:

$\langle X, Y \rangle$

$\downarrow \downarrow$

hash(X), hash(Y)

$\downarrow \downarrow$

~~check hash(X)~~

Do Union(hash(X), hash(Y))

Report all

Union-find

groups/classes

main problem: Union-find will have hash table sized away - $O(m)$ \rightarrow could become very big

\rightarrow Option II:

Just Use Union-find & then for each city name search a linear ~~array~~ integer array for its label.

Algo:

\rightarrow label each

city by a unique number from 1 to n

Call this Array C

For # line (X, Y)

$\downarrow \downarrow$

Find labels for X & Y from array C

\downarrow

Then Union(C[X], C[Y])

\rightarrow Report all

Union find groups

Good: because of design simplicity & space

Bad because $O(n)$ search for every line input

Problem II Grading Rules (50 points)

25

Design

(25pts) — Use either Option I or Option II

~~(15pts)~~ — Any other option ~~that~~ that is worse than the ^{above} ~~other~~ two options

25

Implementation

(5) — Code readability

+

(20) — Does the code work? (testing)

If report is missing or is totally uninformative,
Reduce 10 points for that