Page

Date

Online Course udemy →

① front end new projects pending
② front end revise all projects and notes.
③

– from backend complete remaining –

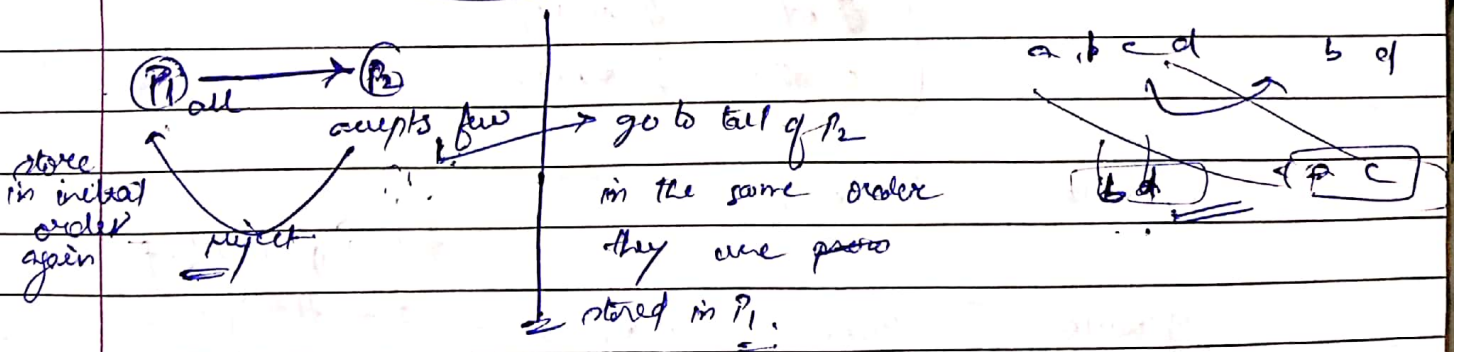
① doubt of validity of email account.

Post Transition Problems | Hackerblocks

A towns_count // no of towns in city. → integer
B every town – some post offices
C every post office – different structure.

Post office → max_weight
→ min_weight

D packages ⇒ Orderly stored. ⇒ used for processing - inventory



E data structures ⇒ structures ⇒ { post office ✓, Town ✓, Package ✓ }

Page
Date

⑥ functionalities ① print_all_packages.

② send_all_acceptable_packages.
↳ source_office_index // source
↳ Target_office_index. // target.

③ town with most_packages.
- if conflict, then print first one from towns collection.

④ find town
- by name
- guaranteed existence

// input

→ 1. towns_count // integer
2. Town blocks [name // string
office_count // integer]

office blocks [totalpkg minweight maxweight]

pkg block. [id // string
weight // integer]

queries. // integer

queries block [type // integer
→ 1 → print_all_packages
→ 2 → transfer
→ 3 → max_pkg_town]

1
town_name // string,

2
source name source_office_index target name target office index,

3
to be follow.

Page

Date

Online geeks for geeks

1. C operator, precedence, associativity — all bookmark pages — revise n execute.

Constraints →

1. 0 ≤ integer ≤ 10.
2. Town_count > 0 | office_count > 0
3. strings < 5 length
4. Town_names — uppercase & unique
5. pkg_ids → lower letter → unique
6. min weight ≤ max weight
7. all queries are valid.

package

- char *id; → string
- int weight;

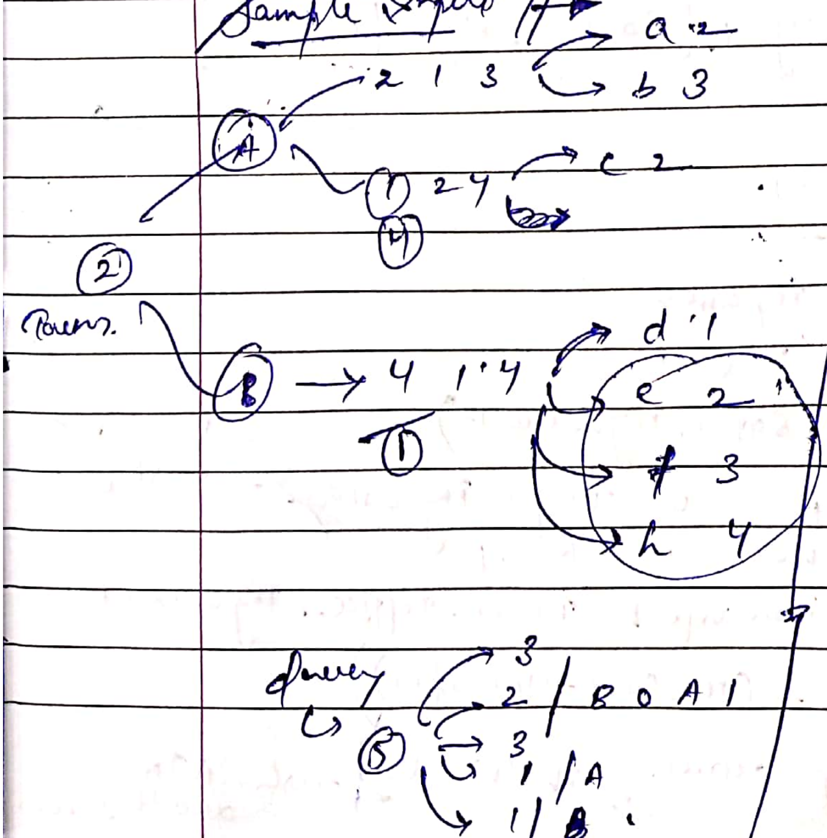
post_office

- int min_weight;
- int max_weight;
- package *packages; → array
- int packages_count;

town

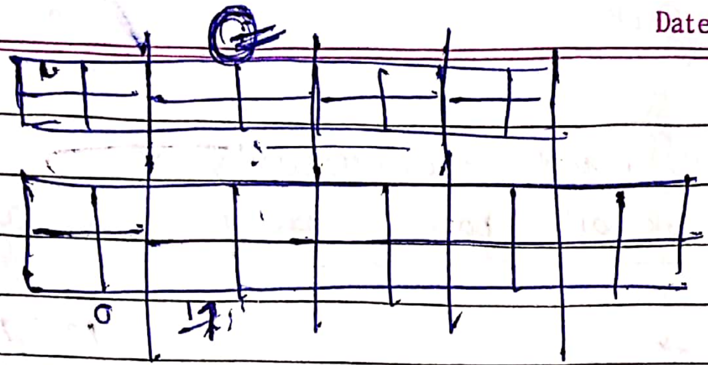
- char *name; → string
- post_office *offices; → array
- int offices_count

Sample Input →

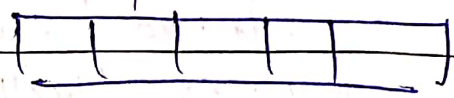


source po1

Target po2



① get the unit of target
② go to source → po → ~~all packa~~ packages



loop for Towns [source] .po [poindex] .pkg count

{ find all the relevant packages.
~~② count them~~
~~④~~ ② → adjust them → ~~decrement count of source~~
③ → • shift all further packages backwards.
① → • add removed package to target list
② → • increment count of ~~source~~ target
④ → • decrement count of source

Towns [target] . po [index] . pkg count
Towns [~~target~~] . po [index] . pkges [pkg count] .

implementation for (i2

① [min weight | max weight] target.

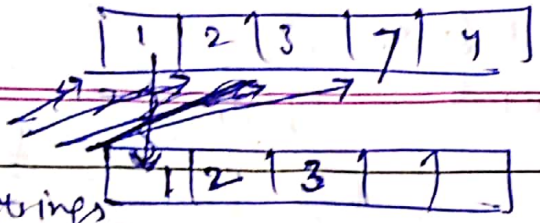
② Source pkg array (dyna) | Target pkg array (dyna)

③ loop (i=0 to ~~tor~~ STown. Soffice. Pkg_Count)
→ if (valid) → push in Target Pkg array → push, realloc
count Total Transfers { → mark weight = -1.
→ increment PTown. Toffice. Pkg_Count

④ malloc (stown. soff ? pkg — (Pkg_Count — Transfers))
~~Free~~ (old wala).
copy data & loop again. → if (invalid) push into another....

Page
Date

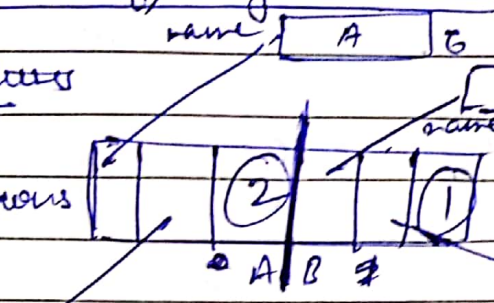
Backtracking



C →

① Sorting of strings

② permutation of strings...



data structures

④ towers