data structure

→ performing some operations

implementation using available features of a prog language.

↳ decide the most important operation(s) → ensure they are efficient.

→ most frequently used operation.

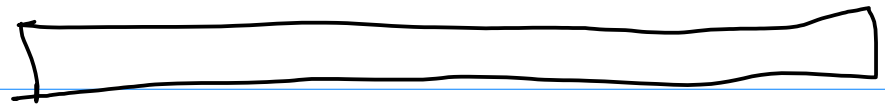
R → register.

This must be executed lots of times, otherwise all other sequences will be small.

R — { R takes as less time as possible!

other → operations. {

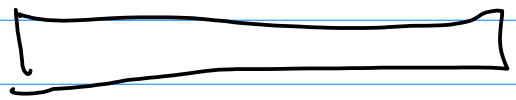
list of courses



sorted vector of courses & their data.

list of students

sorted



map : string → no. (which can be used as index).

Know beforehand all possible strings.

Read all strings → sort once and for all

and to find index use binary search! → lowerbound.

store registration data.

course
↓
list of registered students.

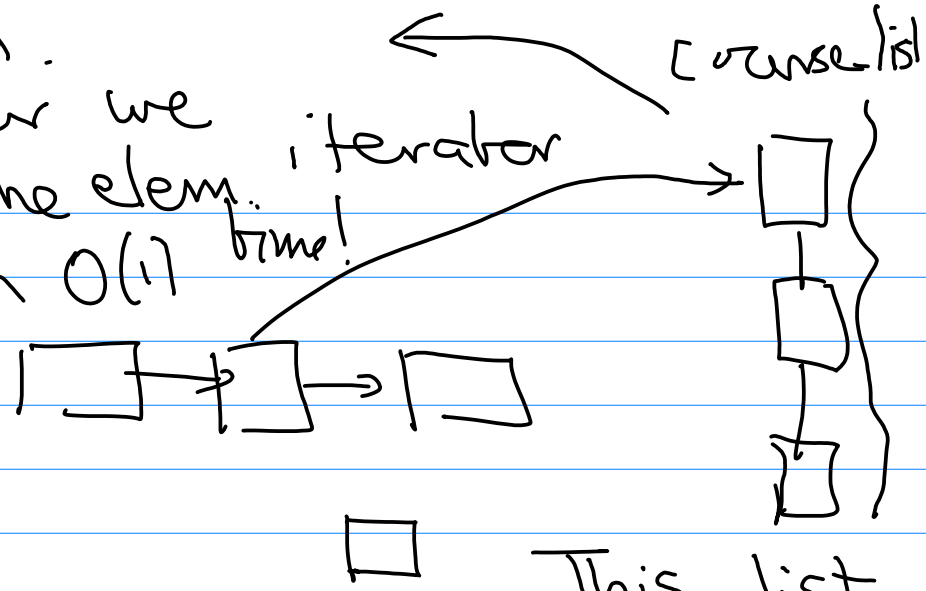
student

list of registered courses!

This list would be small. → ≤ 10

list is needed.

given the iterator we can delete the elem. in O(1) time!

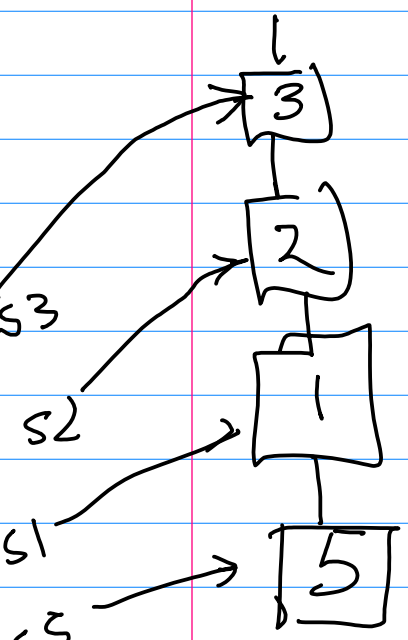


student list in sorted order of courses.

internally all students / courses are members!

This list was not sorted!

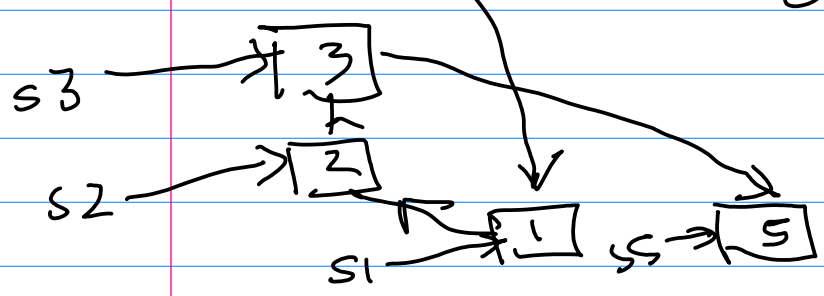
registration / drop only need to traverse student list! which is small!



L.sort() course-list.

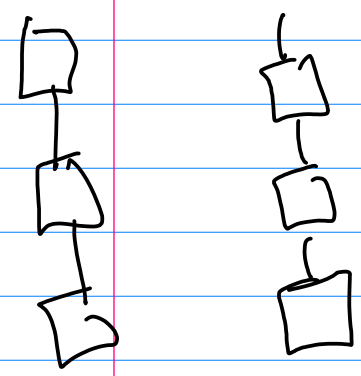
→ sort only when print is required!

L.sort() → merge-sort & only no copying changes of values pointer.



common students.

c1 c2 → sort both and find common elements!



→ student list is small.

sort only the smaller of the two lists.

c1 c2

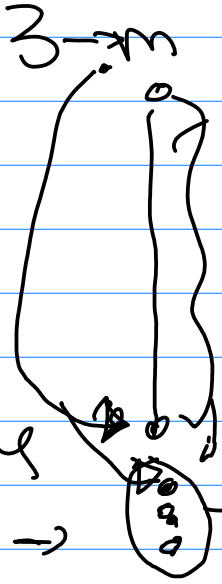
traverse the student list and check if c2 occurs!

first

10^7 → half to be just R.

There are improvements possible! 1 min. → 30 sec! 3 or 4 times faster on faster machines

3 → m



c list known to be sorted.

sort when printing.

sort this portion only & merge! → possibly unsorted. did not really help!

Assigment 2.

substring.

is w a substring of $f^n(a)$.

→ if w is a substring then it is substring of $f^n(a)$ for some n such that length of $f^n(a)$ is not "much larger" than that of w.

Simple algorithm → find $f^n(a)$ such that if w is a substring it is a substring of $f^n(a)$ → use KMP → pattern matching to check!

$|w| \leq 10^4$!

fibonacci word. $|w| = 10^4$

find the smallest Fibonacci number $\geq 10^4$ say it is f_i

$|f^i(a)| \geq |w|$.

$f^i(a)$

$$f^{i+1}(a) = f^i(f(a)) = f^i(ab)$$

$$= \underbrace{f^i(a)\, f^i(b)}_{w}. \qquad |w| \le |f^i(a)|$$

$$f^{i+2}(a) = f^i(aba) = f^i(a)\, f^i(b)\, f^i(a)$$

already have occurred in $f^{i+1}(a)$

w

$$f^{i+3}(a) = f^i(a)\, f^i(b)\, f^i(a)\, f^i(a)\, f^i(b)$$

w $\qquad |w| \le |f^i(a)|$

$$f^{i+4}(a) = f^i(abaababa)$$

No point checking this! use KMP pattern matching.

all such strings would have appeared earlier!

$O(|f(n)|) = (2^n)$ times.

general. generate only the portion of $f^n(a)$ that may contain w.

$f(abaab)$ $\quad f^i(a) . f^i(a)$

aba $\quad$ ba $\rightarrow$ small no. of patterns for which you need to check!

ab

a.