Tug Of War

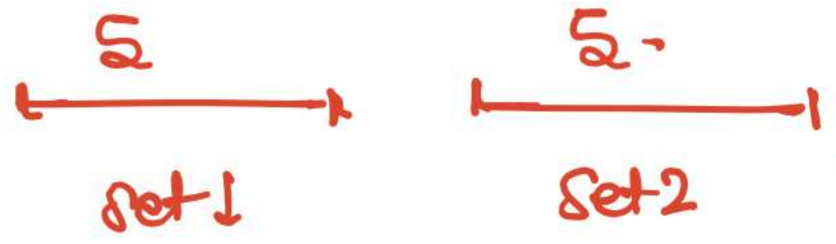
Saturday, 31 July 2021 6:25 PM

① n Elements. present in an array 'arr'

② Divide 'n' Elements in two part Such that both have same no. of Elements or may have one more element in a set

③ print set having min diff. [S1][S2]

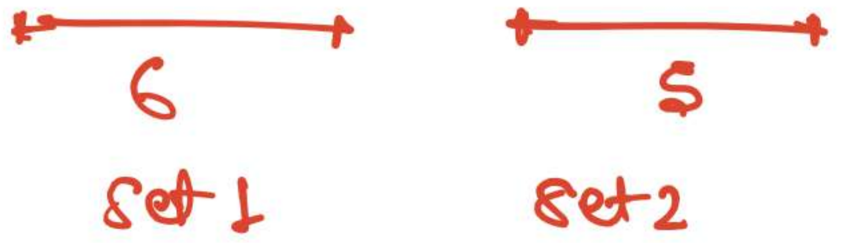
size =10 (Even) →



} → Difference of sum of set minimise.

] Size is Same in Both set

size=11 (odd) →



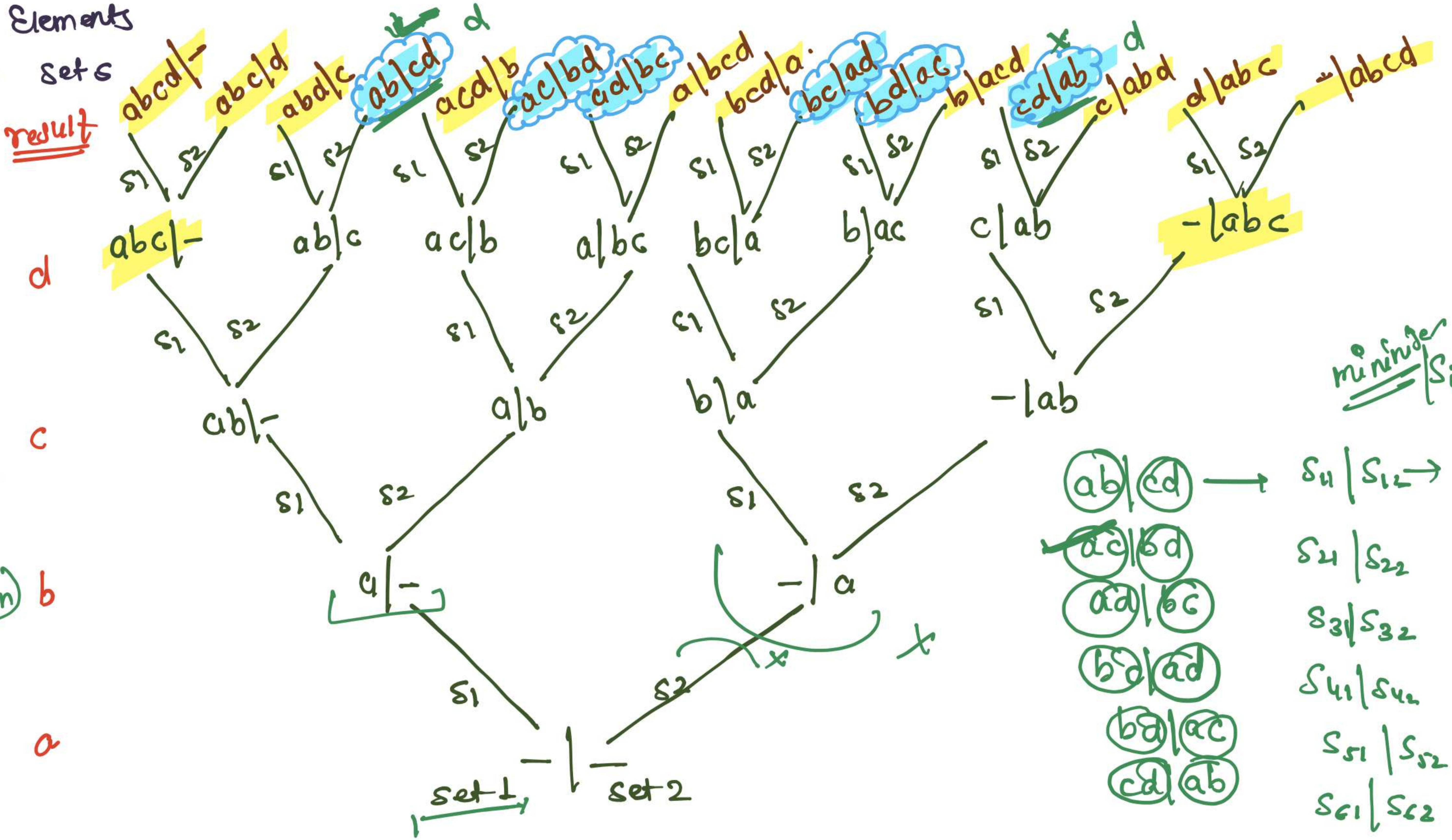
} → Difference of sum of set minimise.

] one set have one more element from another

Element → a, b, c, d → set①, set② valid-set → |size difference| <= 1

level → Elements

option → sets



global variable

Min → update

set → update

if (diff < min)

minimize $|S_{ij} - S_{ij+1}|$

ab | cd	→	$S_{11} \| S_{12}$ →
~~ac~~ | bd		$S_{21} \| S_{22}$
ad | bc		$S_{31} \| S_{32}$
bc | ad		$S_{41} \| S_{42}$
bd | ac		$S_{51} \| S_{52}$
cd | ab		$S_{61} \| S_{62}$

Pattern Matching

Saturday, 31 July 2021 6:25 PM

String → graphtreesgraph

pattern → pep

print all outputs

result → (i) p → graph e → trees

String → mzaddytzaddy.

patter → a b c b

} Example

output

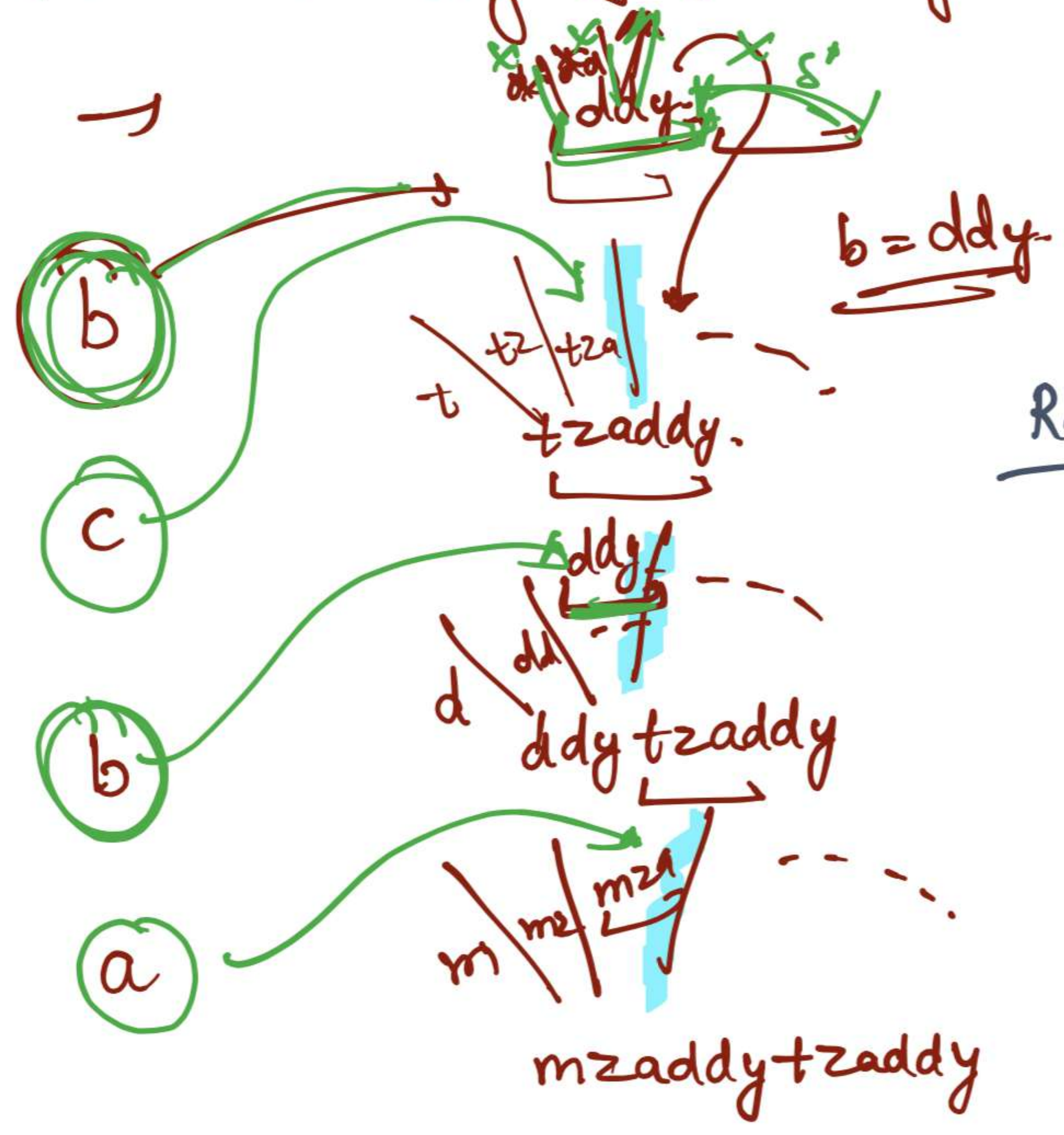
a → m	b → zaddy	, c → t
a → mz	b → addy	, c → tz
a → mza	b → ddy	, c → tza
a → mzad	b → dy	, c → tzad
a → mzadd	b → y	, c → tzadd

String → mzaddytzaddy

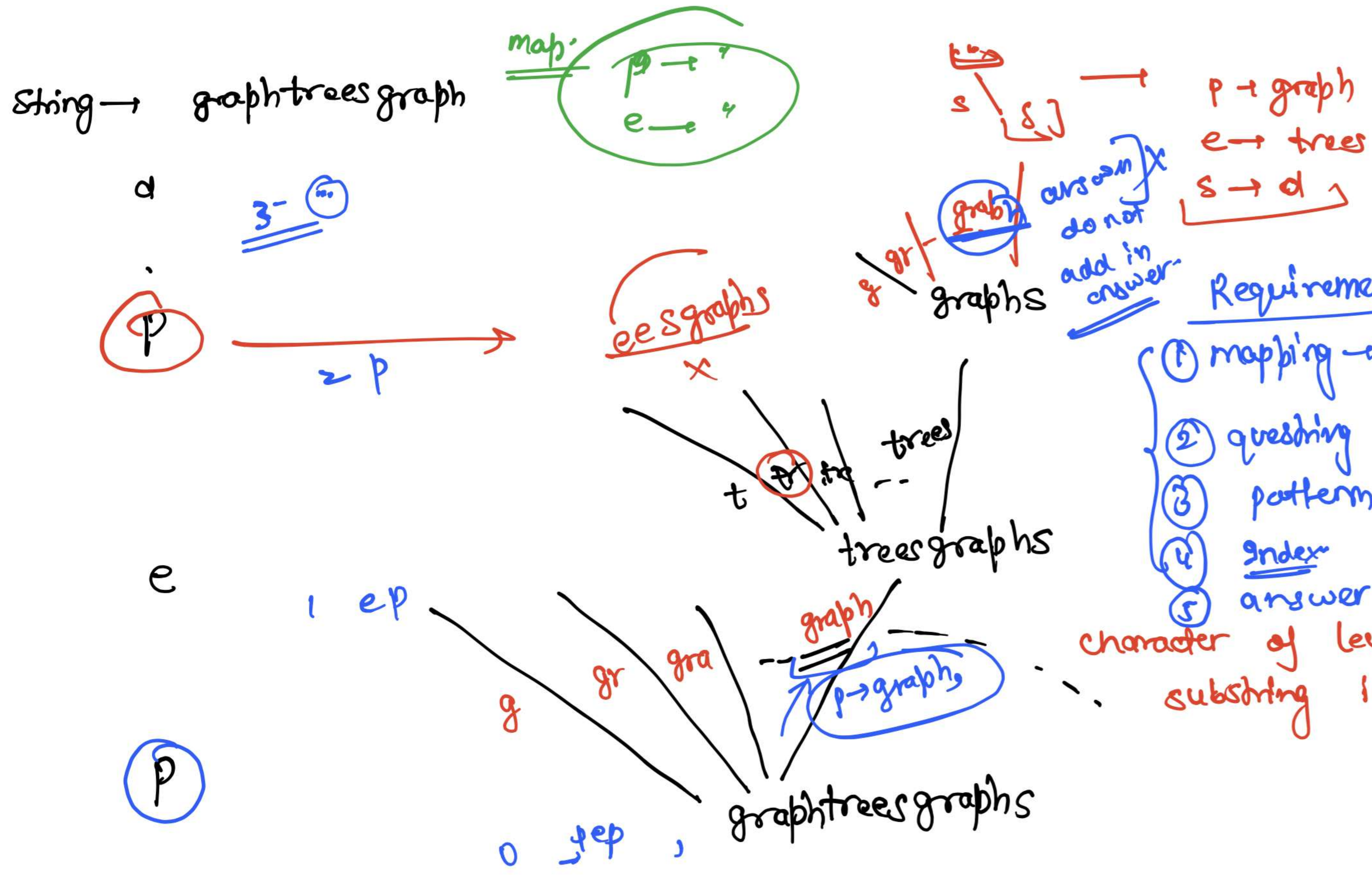
pattern → abcb

level → character of pattern.

options → substring of given string.



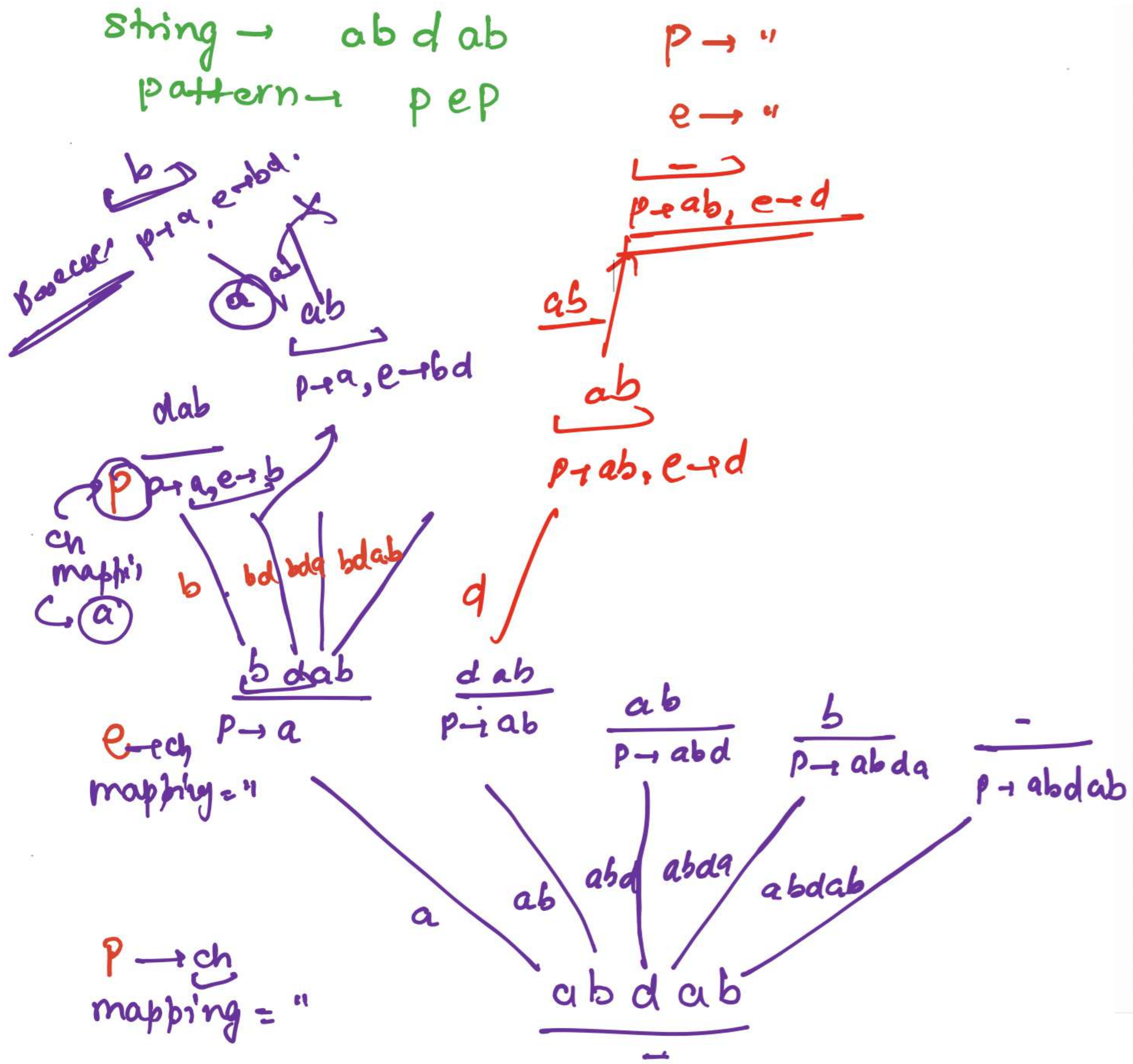
Requirements.

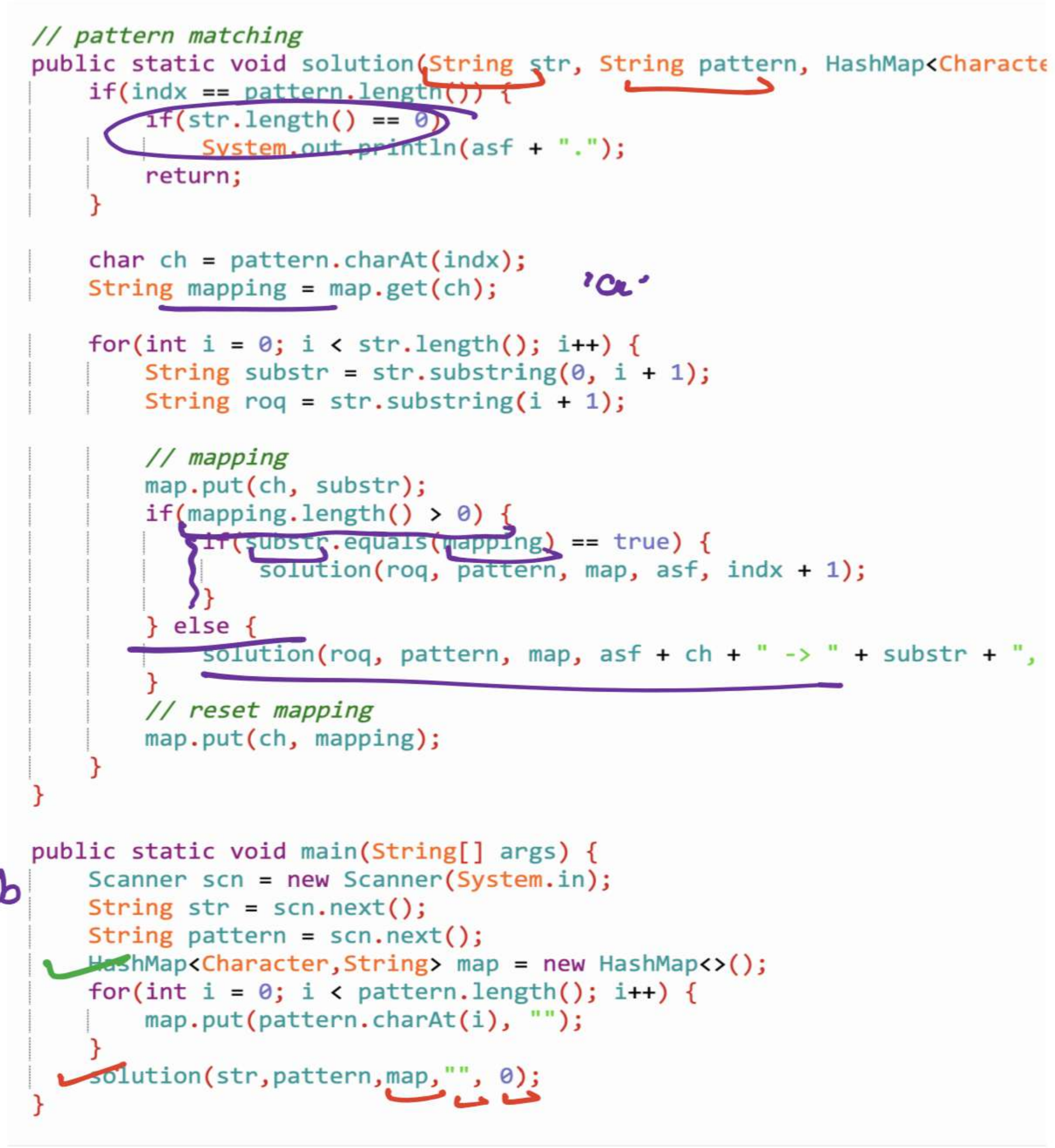


Requirement

1. mapping → HashMap
   ch → string
2. question string
3. pattern
4. index
5. answer so far

character of level is mapped substring in options





```java
// pattern matching
public static void solution(String str, String pattern, HashMap<Characte
    if(indx == pattern.length()) {
        if(str.length() == 0)
            System.out.println(asf + ".");
        return;
    }

    char ch = pattern.charAt(indx);
    String mapping = map.get(ch);

    for(int i = 0; i < str.length(); i++) {
        String substr = str.substring(0, i + 1);
        String roq = str.substring(i + 1);

        // mapping
        map.put(ch, substr);
        if(mapping.length() > 0) {
            if(substr.equals(mapping) == true) {
                solution(roq, pattern, map, asf, indx + 1);
            }
        } else {
            solution(roq, pattern, map, asf + ch + " -> " + substr + ",
        }
        // reset mapping
        map.put(ch, mapping);
    }
}

public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.next();
    String pattern = scn.next();
    HashMap<Character,String> map = new HashMap<>();
    for(int i = 0; i < pattern.length(); i++) {
        map.put(pattern.charAt(i), "");
    }
    solution(str,pattern,map,"", 0);
}
```

Word Break - I

Saturday, 31 July 2021 6:25 PM

11 $\longrightarrow$ n

per

i like pep coding pepper eating mango man go in pepcoding $\longrightarrow$ Dictionary of 'n' words

ilikepeppereatingmangoinpepcoding $\longrightarrow$ string

break words

Ex

i like man go mango

i like mango

i like man go

i lik mango

pep

S1 → i _ like _ pepper eating man go in pep coding

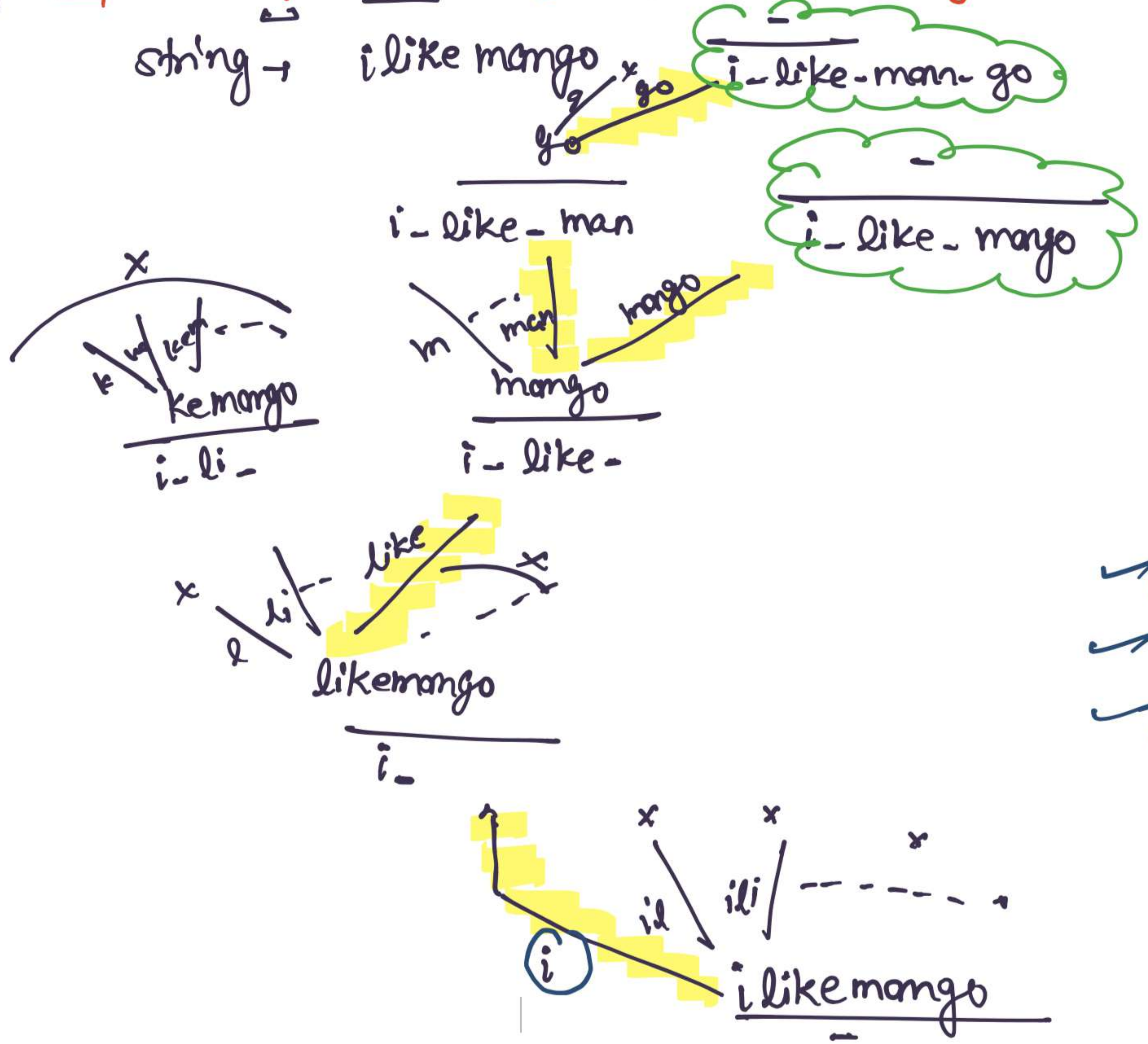
S2 → i like pepper eating man go in pepcoding

S3 → i like pepper eating mango in pep coding

S4 → i like pepper eating mango in pepcoding

Dictionary → i li like man go mango cat dream

string → ilikemango



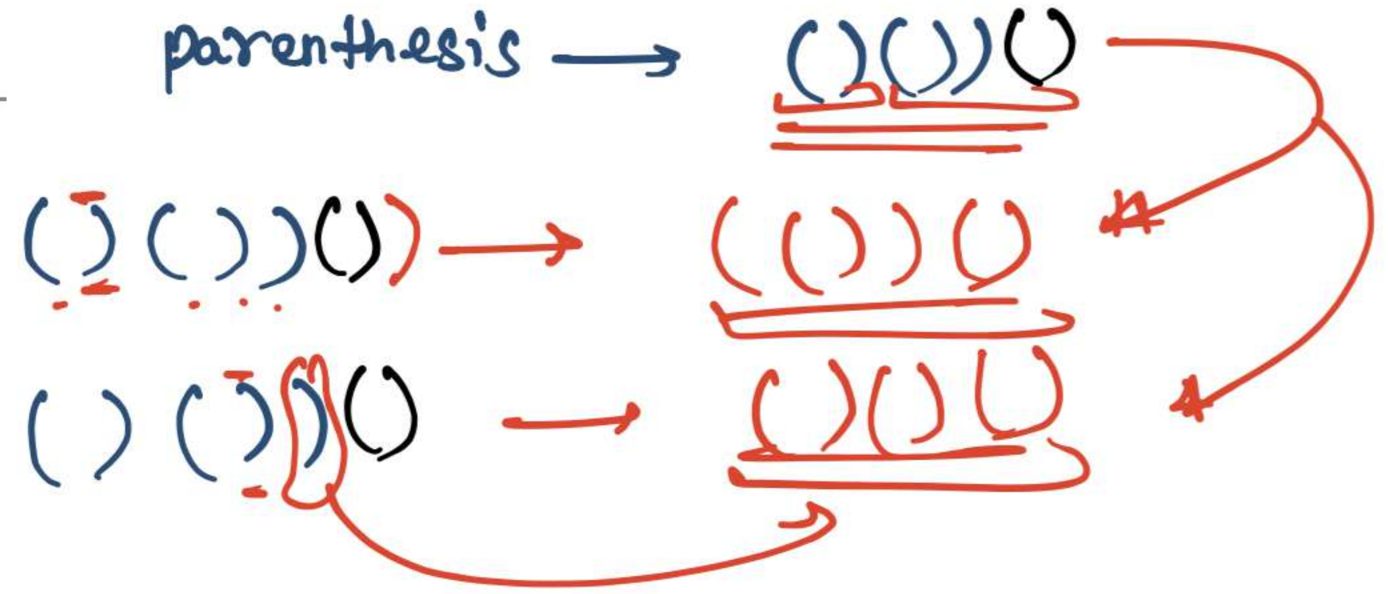
i_like_man_go

i_like_mango

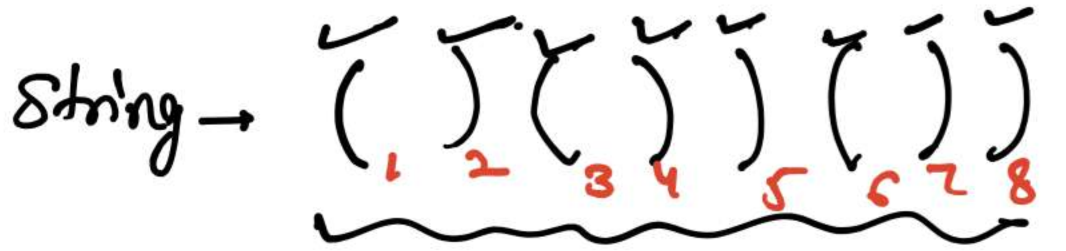
- Hashset<string> → diction,
- question string
- answer so far

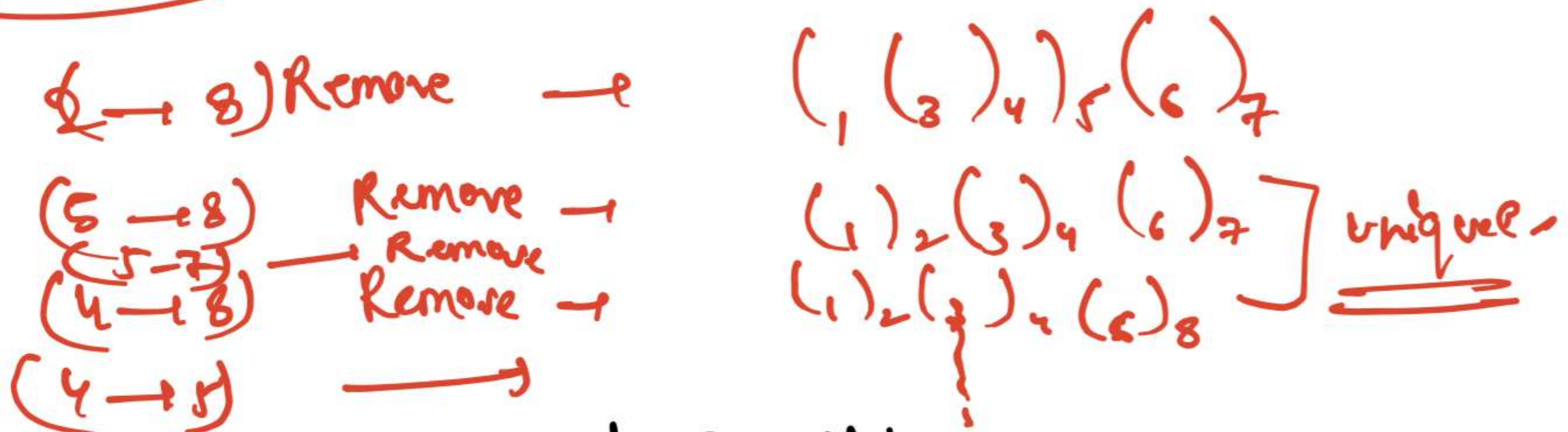
Remove Invalid Parenthesis

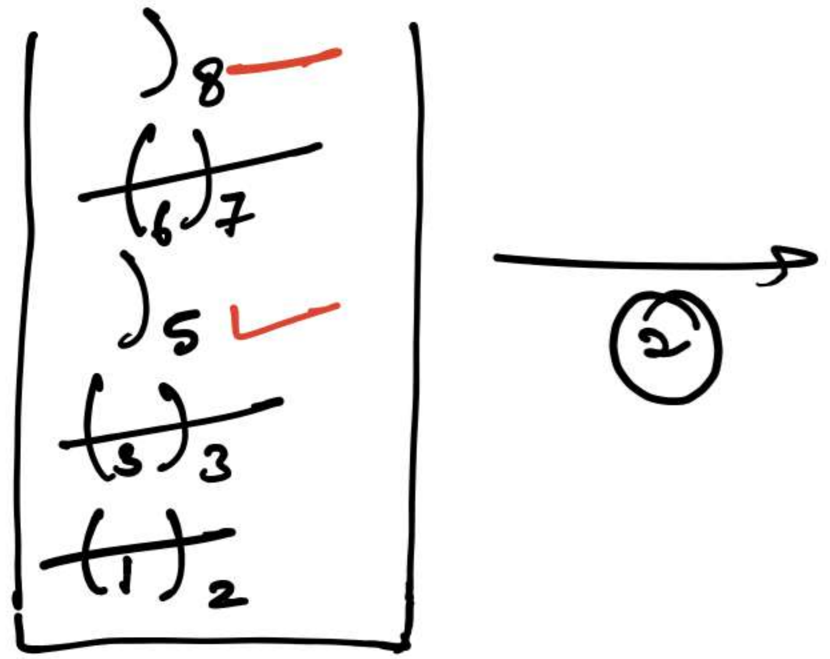
Saturday, 31 July 2021 6:27 PM



Min Removal of parenthesis







stack.size → No. of Invalid parenthesis

$\underbrace{\qquad\qquad}$

Min no. of parenthesis required to removes

Min Removal = 2.

5 → 7

5 → 8

String → ( ) ( )) ( ))

Valid parenthesis (print)

min Removal