Lecture - 29

Recursion, palindrome, permutation

1) Given a string s and the task is to print all the substring present inside the string s.

s = abc

↳ Total substring

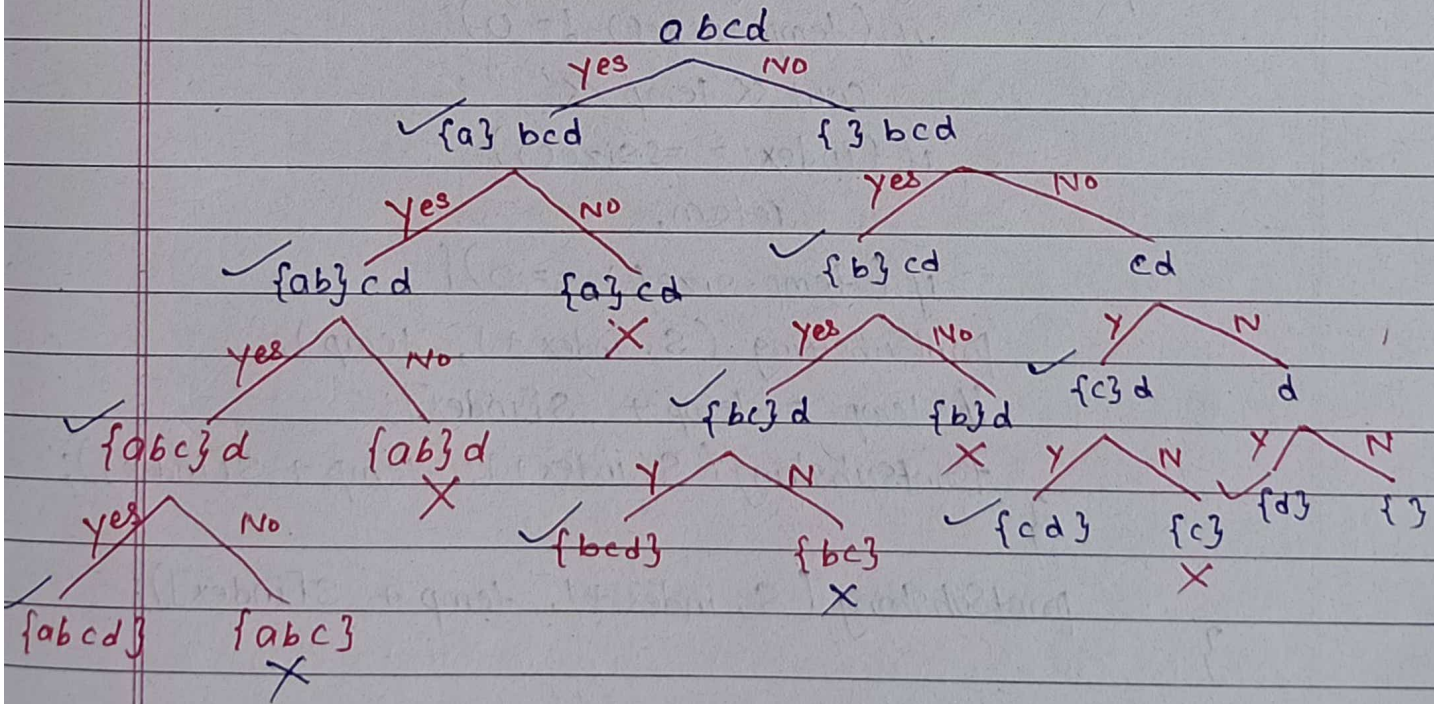
a, b, c, ab, bc, abc

s = abcd

a, b, c, d, ab, bc, cd, abc, bcd, abcd

↳ Total substring

By recursion tree



Total = 10

Teacher's Signature.....

Page No.:
Date: / /

1st Approach

print all substring using for loop.

```
for (int i=0;  i< s.size(); i++) {
    for(int j=0 ; j < s.size(); j++) {
        print substring ( i to j);
    }
}
```

2nd Approach

By recursion:

→ Element found : print it
→ After that which ever element found print it.

```
void printSubstring ( String S, int index, String temp){
        if (temp.size() != 0)
            cout << temp << " ";
        if (index == s.size())
            return;
        if ( temp.size() == 0){
    printSubstring (S, index+1, temp);
    // temp = temp + S[index];
    PrintSubstring ( S, index+1, temp + S[index]);
        }
    printSubstring (S, index+1, temp + S[index]);
}
```

Teacher's Signature.....

```
int main() {
    String s;
    cin >> s;
    printSubstring (s, 0, " ");
    return 0;
}
```

2) Combination sum

You have given an array of n elements and a target. By adding elements from array find the target and return the array of element.

Note: one element can be repeated any No. of time.

Ex: Candidate = [2, 3, 6, 7] target = 7

output: [2, 2, 3]
[7]
→ It is like a 2D-Array

Size is not fixed, so we used arraylist or vector

vector < vector << int >> ans; → 2D vector
vector <int> vec; → 1D vector

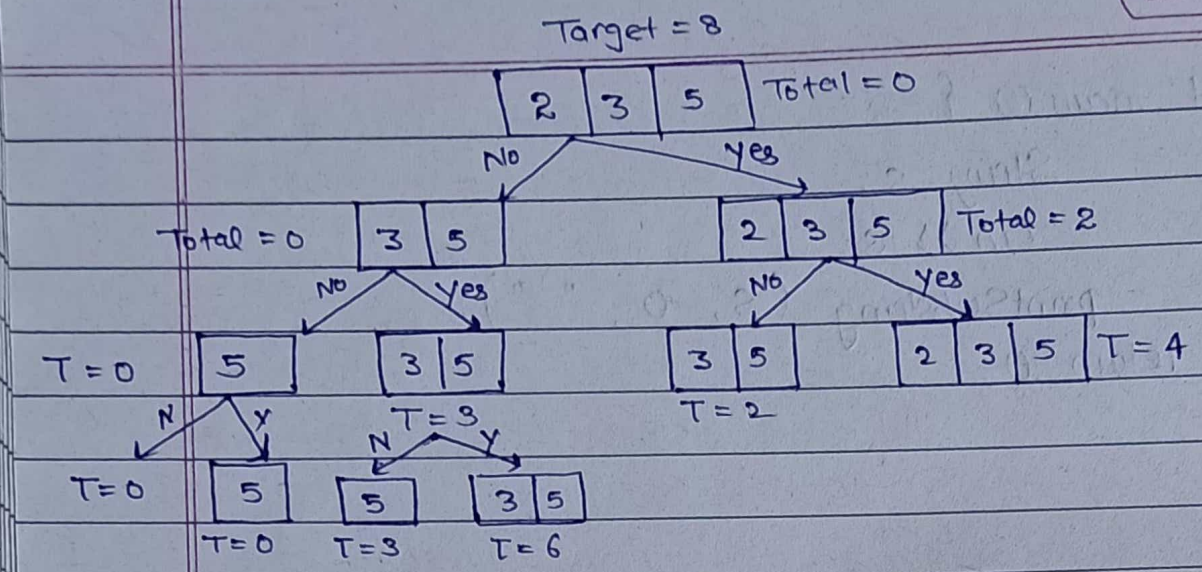
on 2D vector, we can directly push 1D vector by

ans.push_back(vec);

or

ans[0].push_back(5);
↳ on this we put element by element in the array.

Teacher's Signature.....



Breaking point :

① if (sum > target) {

}

② if (sum == target) {

}

③ if (index == arr.size) {

}

Here we need a vector or Arraylist.

* Code in C++ :

```
Vector<vector<int>> CombinationSum (vector<int>&
        candidates, int target){
    vector<vector<int>> ans;
        vector<int> temp;
```

```
        int sum = 0;
    find (candidates, ans, temp, sum, target, 0);
     return ans;
}

void find (vector<int> & candidates, vector<vector<int>> & ans,
     vector<int> temp, int sum, int target, int index){

        if (index == candidate.size()) {
              if (sum == target)
                ans.push_back (temp);
             return;
        }

        if (sum > target)
          return;

    find (candidates, ans, temp, sum, target, index+1);
      sum += candidates [index];
    temp.push_back (candidates [index]);
    find (candidates, ans, temp, sum, target, index);

}
```

Teacher's Signature.....

3) Permutation of a given string

You have given a string.
You have to find all the permutation of given string.

Ex: ABC

→ ABC ACB
BAC BCA
CAB CBA

What we do here

take 2 possibility

A _ _
A B C A C B

Similarly,

B A/C _
B A C B C A

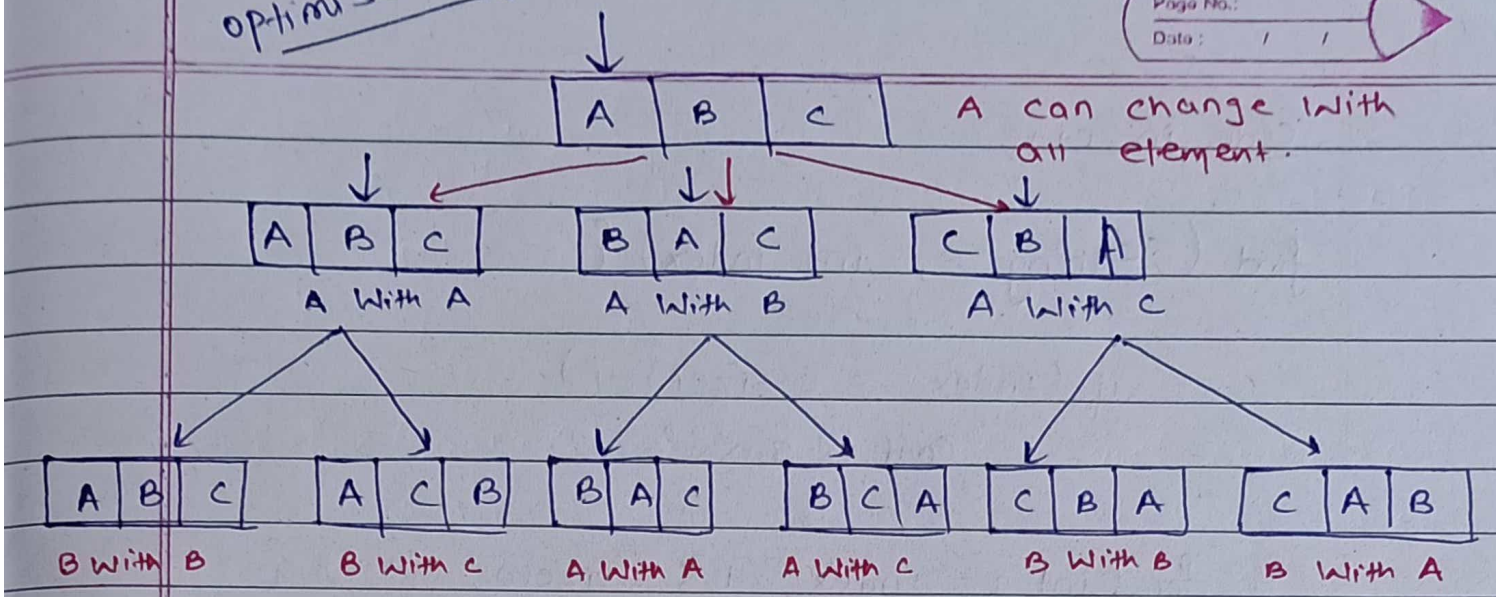
C A/B _
C A B C B A

No, we can understand it by recursion tree:

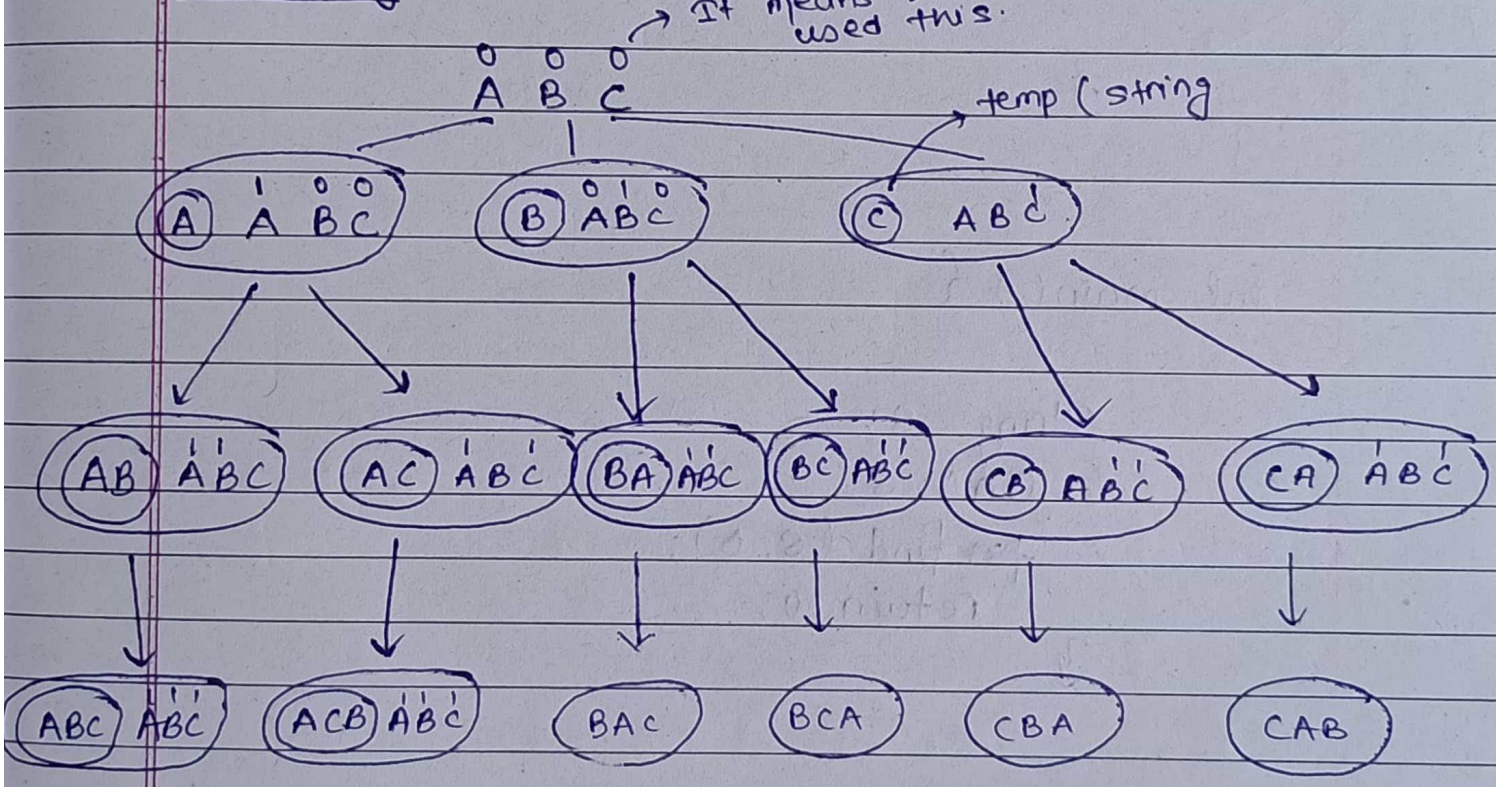
How it works:

Teacher's Signature.....

optimise logic



2nd logic



Teacher's Signature.....

Code in c++;

```
find ( String s, int index) {

    if (index = s.size()-1)
        Cout << s << " ";

  for (int i= index; i < s.size(); i++) {
          Swap( s[i], s[index]);
          find ( s, index+1);
          Swap ( s[i], s[index]);

    }
}

int main() {

    String s;
    cin >> s;
    find (s, 0);
    return 0;
}
```

Teacher's Signature.....

4) Letter Combination of a phone number :

→ In keypad phone
↳ If we want to write alphabet then we use number button from 2 to 9

→ you have get a number and find all the possible string which is generated by phone.

2 abc 3 def

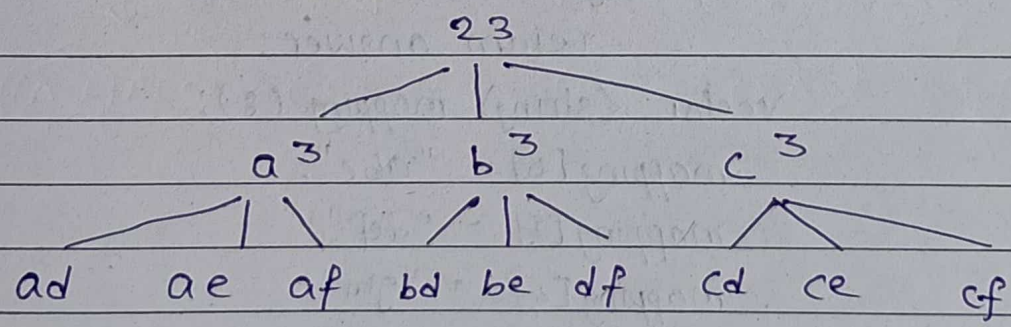
4 ghi 5 jkl 6 mno

7 pqrs 8 tuv 9 wxyz

If number = 5
possible string : j, k, l

let num = 23

2 3
a b c d e f

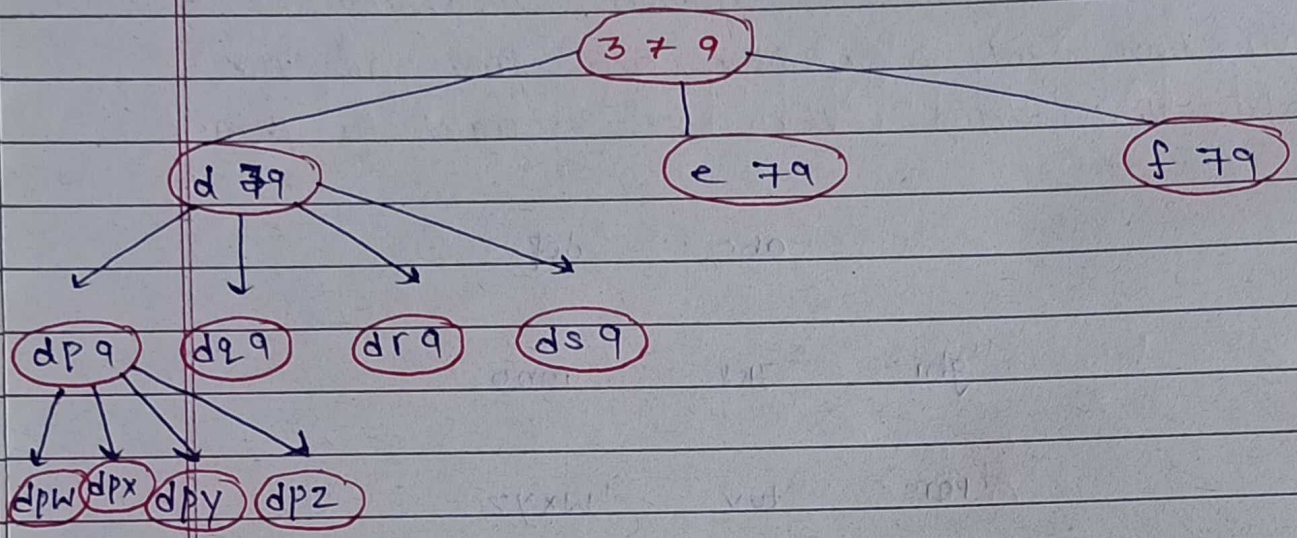


These are the possibilities which can be make by 23.

Teacher's Signature.....

Num = 379

3	7	9
def	pqrs	wxyz



By this we can calculate other strings.

Code:

```
Vector <string> letterCombination (string digits){
        Vector <string> answer;
        if (digits.size() == 0)
            return answer;
        vector <string> mapping (8);
        mapping[0] = "abc";
        mapping[1] = "def";
        mapping[2] = "ghi";
        mapping[3] = "jkl";
        mapping[4] = "mno";
        mapping[5] = "pqrs";
        mapping[6] = "tuv";
```

Teacher's Signature.....

```
        mapping[7] = "wxyz";
    find(digits, answer, mapping, "", 0);
    return answer;
}

Void find(String &digits, Vector<string> &answer,
          Vector<string>&mapping, String temp, int index){

        if(index == digit.size()){
            answer.push_back(temp);
            return;
        }

        int pos = digits[index] - '2';
    for(int i=0; i<mapping[pos].size(); i++)
find(digits, answer, mapping, temp+mapping[pos][i],
                                index+1);

}
```