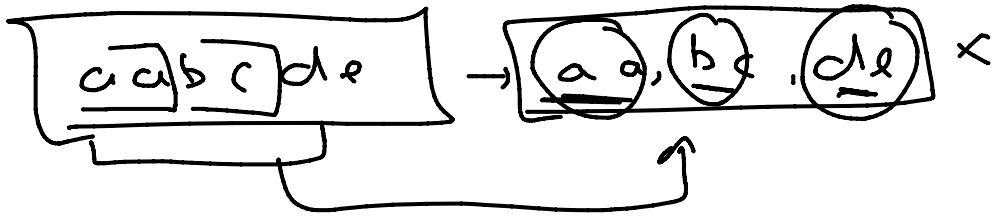
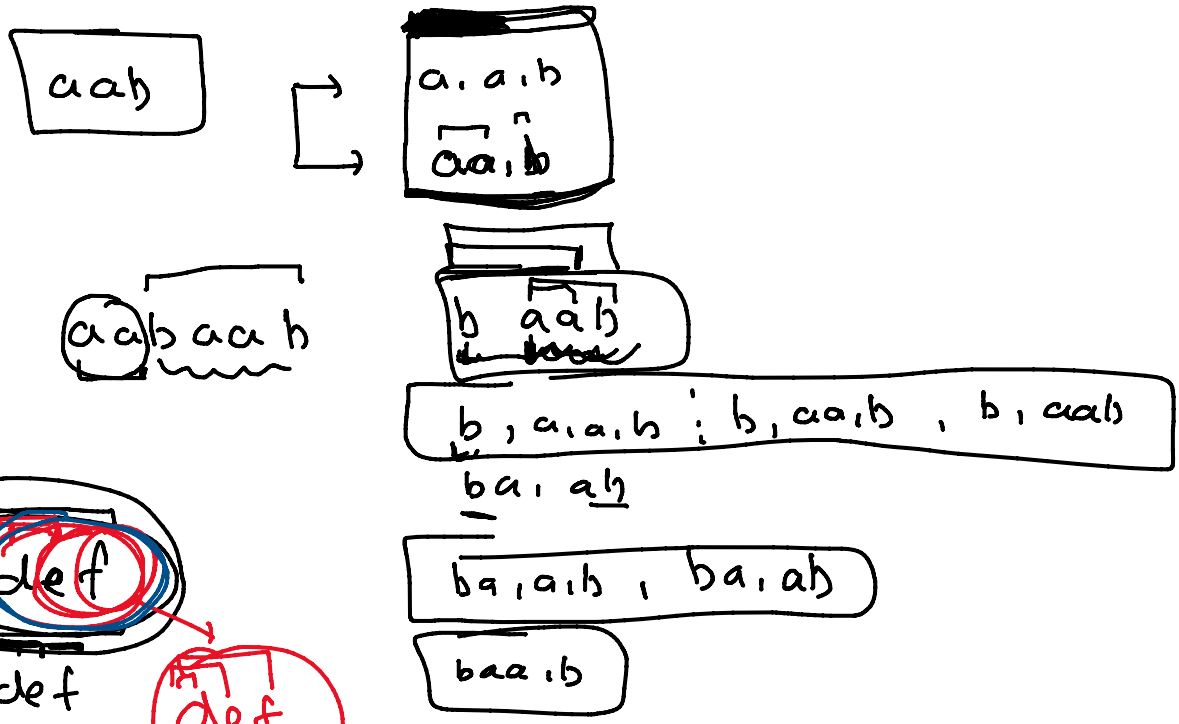
Class 73

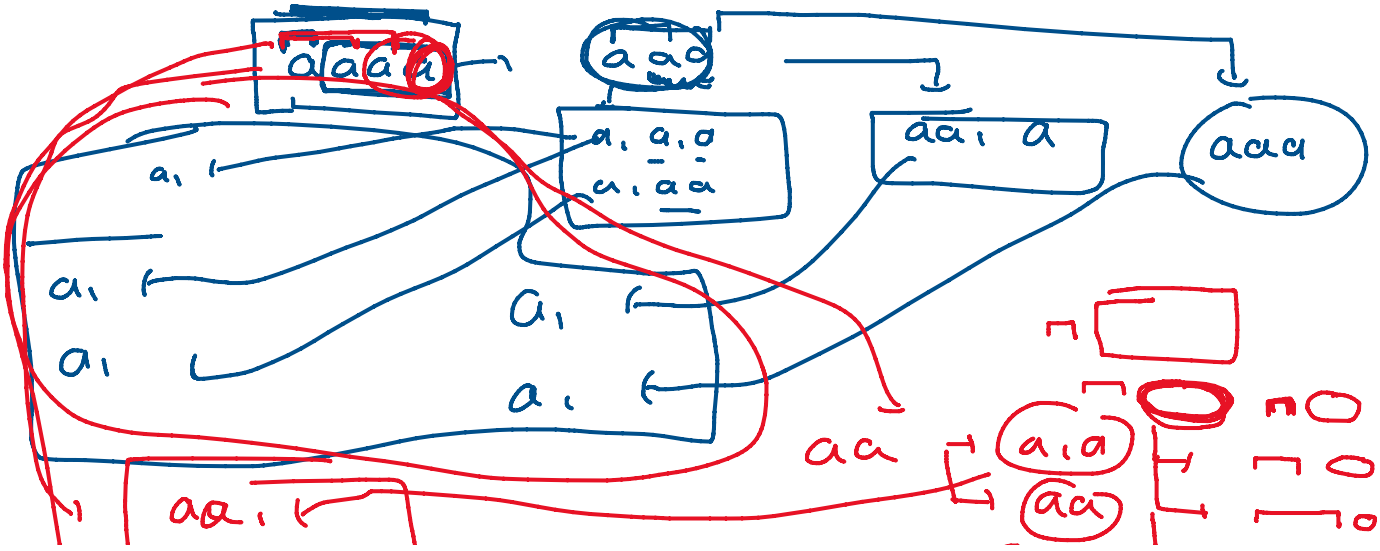
01 May 2023 21:08

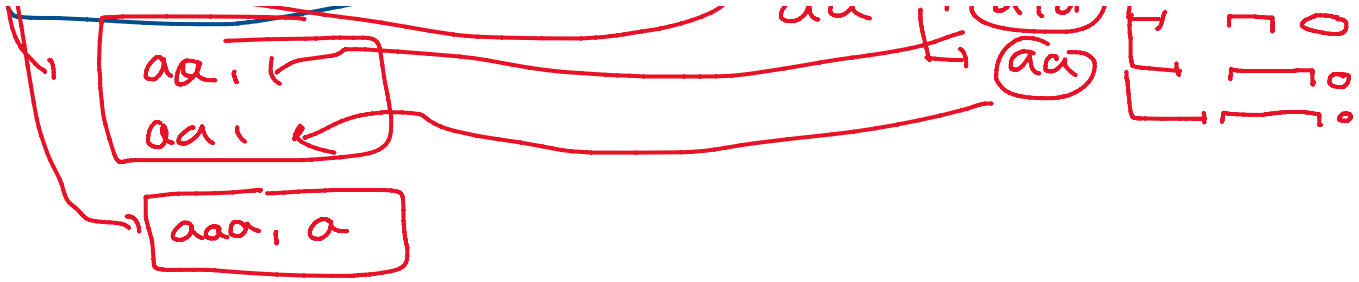
Backtracking

Palindrome Partitioning –










```
vector<vector<string>> partition(string s)
{
    vector<vector<string>> ans;

    int n = s.size();
    vector<string> temp;

    solve(s, ans, temp, 0);

    return ans;
}
```

aahc

```
void solve(string s, vector<vector<string>>& ans, vector<string>& temp, int idx)
{
    int n = s.size();
    if(idx == n)
    {
        ans.push_back(temp);
        return;
    }

    for(int i=idx; i<n; i++)
    {
        string str = s.substr(idx, i-idx+1);

        bool check = checkPal(str);

        if(check)
        {
```

a b c d e f

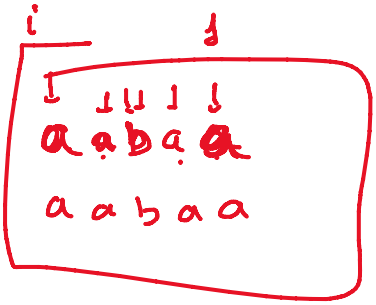
i=1

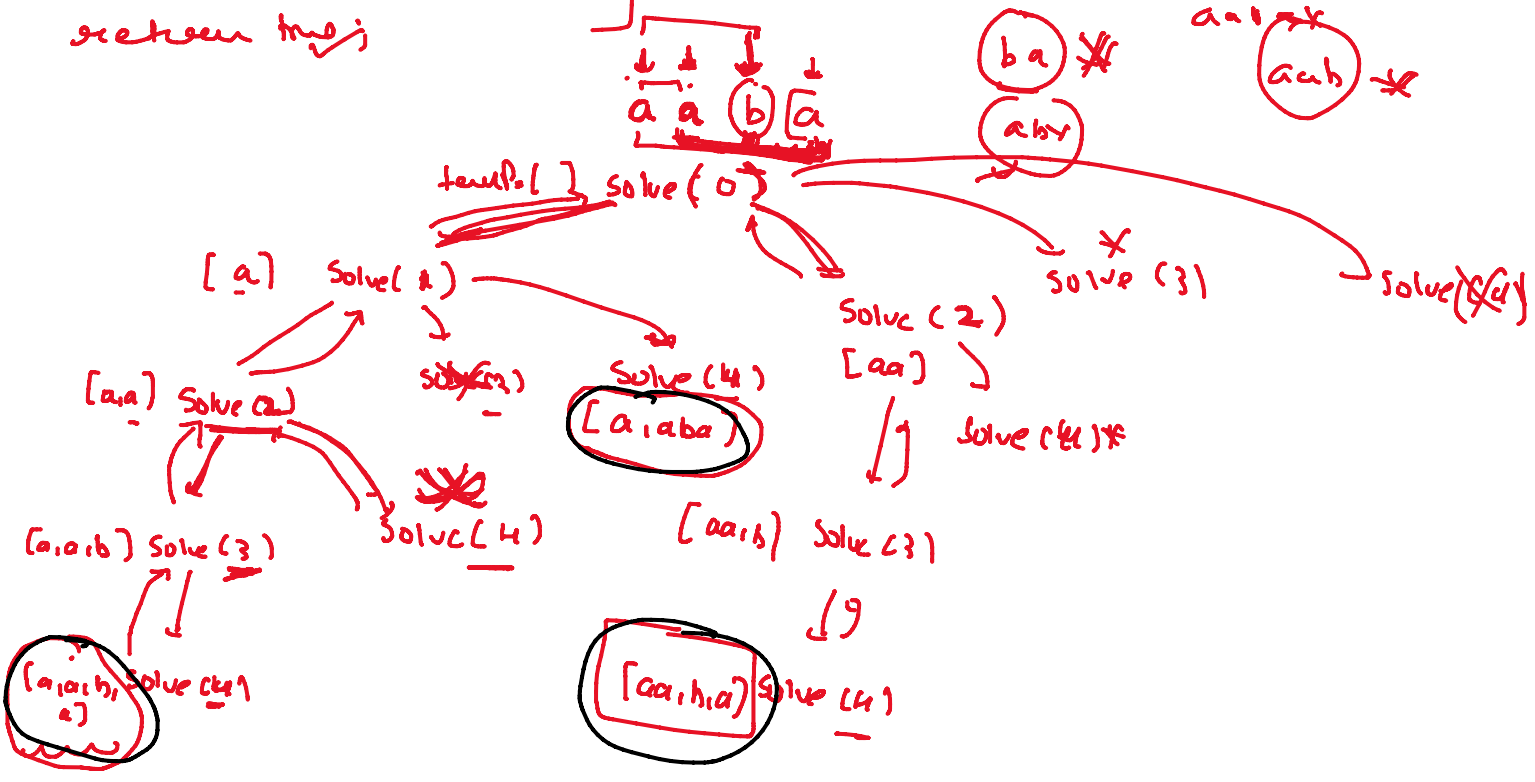
idx

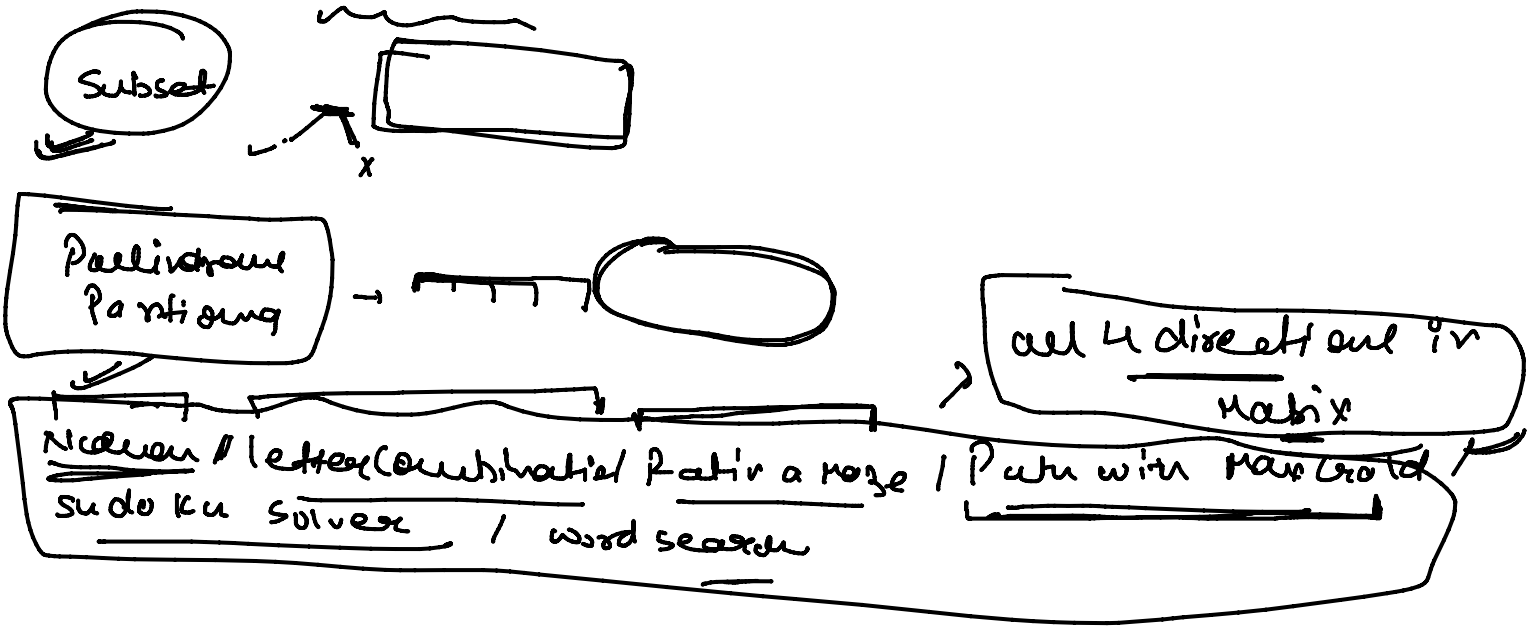
1-1+1 = 1

```
        if (check)
        {  temp.Push_back(str);
           Solve(Save, temp, i+1);
           temp.Pop_back
        }
    }
}
```

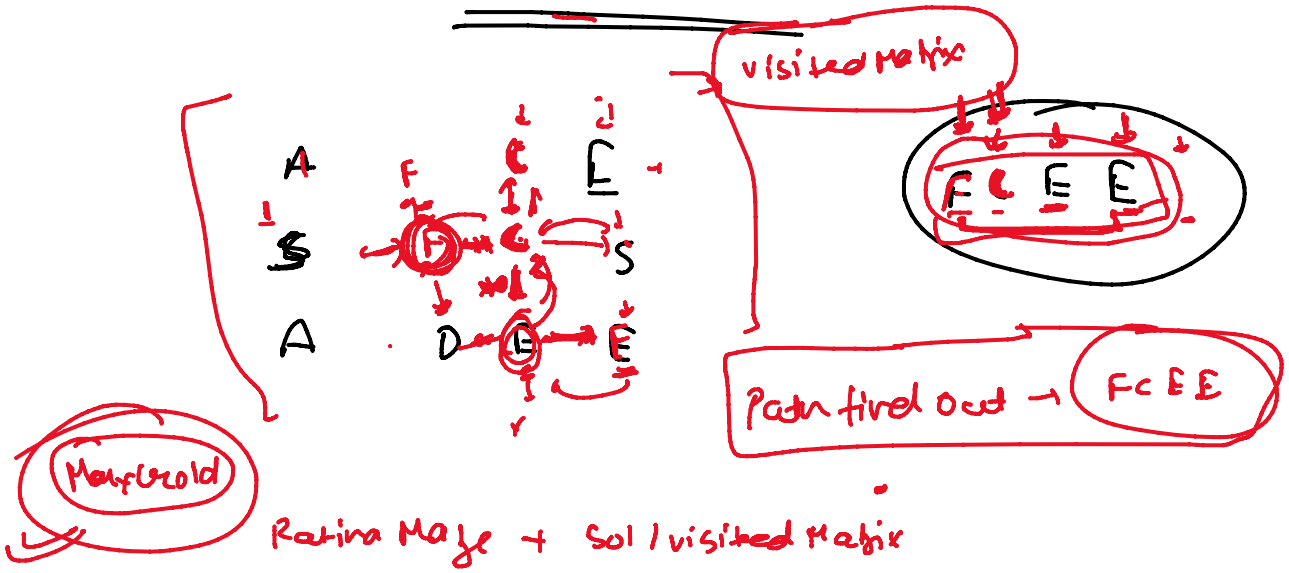
```
bool checkPal(string str)
{  int i=0;
   int j = str.size() - 1;

   while (i <= j)
   {  if (str[i] != str[j])
         return false;

      i++; j--;
   }
   return true;
}
```







WORD search

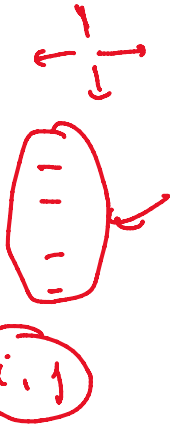


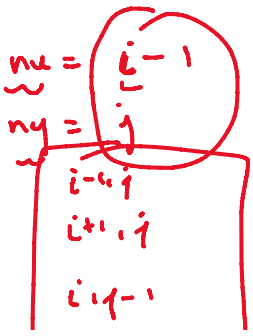

```
bool exist (vector <vector <char> > & board, string word)
{
    int i,j, k, n, m;

    n = word.size();
    int bn = board.size();
    int bm = board[0].size();
```

```
for (i=0; i< bn; i++)
{  for (j=0; j< bm; j++)
   {  if (board[i][j] == word[0])
      {  char cw = board[i][j];
         board[i][j] = '!';
         bool chck = is( board, i, j, word, 1);
         if (chck)
         {  return true;
         }
         board[i][j] = cw;
      }
   }
}

bool is(vector<vector<char>>& board, int i, int j, string word,
        int idx)
{  int n = word.size();
   if(idx == n)
   {  return true;
   }
   int bn = board.size();
   int bm = board[0].size();
   int dx[4] = {-1,1,0,0};
   int dy[4] = {0,0,1,-1};
   for (k=0; k<4; k++)
   {  int nx = dx[k] + i;
```

A → Z

abcd





```
for (int k = 0; k < 8; k++)
{
    int nx = dx[k] + i;
    int ny = dy[k] + j;

    if (nx >= bn or nx < 0 or ny >= bm or ny < 0 or
        board[nx][ny] == '?')
    {
        continue;
    }

    if (board[nx][ny] == word[idx])
    {
        char ch = board[nx][ny];
        board[nx][ny] = '?';
        bool out = is(board, nx, ny, word, idx+1);
        if (out) return true;
        board[nx][ny] = ch;
    }
}

return false;
}
```

i, j-1

i, j+1