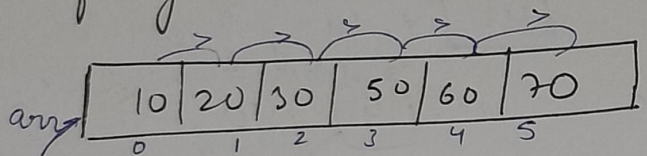
Lecture-21 Recursion - Class 3 Date-13/10/2023

Check if a given no is sorted or not.



f (array, size, index)

if (arr, size, index)

arr[index] > arr[index-1] → true

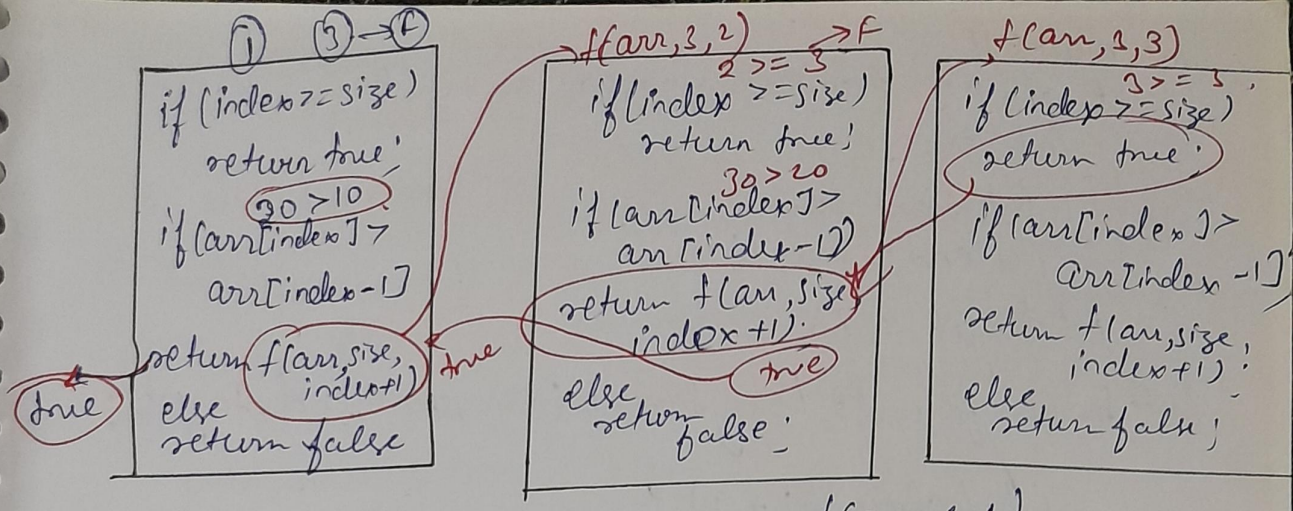
→ false : return false

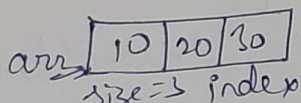
if (arr, size, index+1).

base case.
if (index >= size){
return true;
}

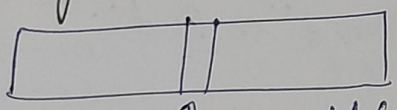
* Code:-

```
bool checkSorted(int arr[], int size, int index){
    //base case
    if(index >= size){
        return true;
    }
    if(arr[index] > arr[index-1]){
        //ab recursion handle karega
        bool aageAns = checkSorted(arr, size, index+1);
    }
    else{
        return false;
    }
}
int main(){
    int arr[] = {10}
    int size = 1;
    int index = 1
    bool isSorted = checkSorted(arr, size, index);
    if(isSorted){
        cout << "Array is Sorted" << endl;
    }
    else{
        cout << "Not";
    }
```





Ques Binary Search with Recursion :-



Basecase if(s > e) return -1;

if (arr[mid] == target) return mid;

Para array traverse ho gya ans nhi mila.

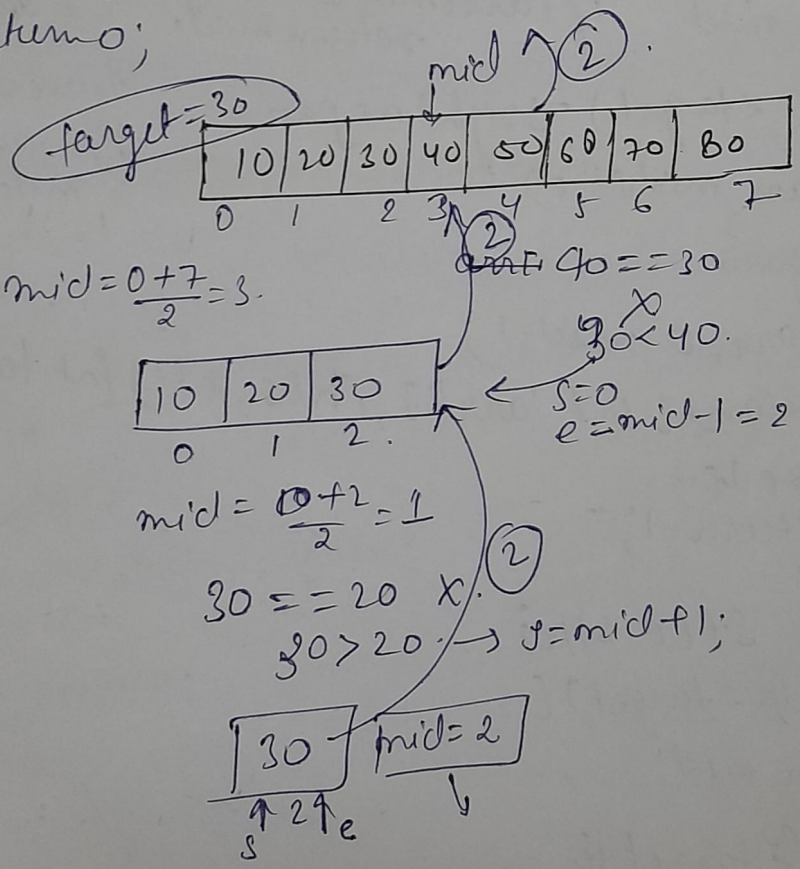
⇒ if (arr[mid] < target) ⇒ right me jao
return f(arr, mid+1, e);

⇒ else left me jaao.
return f(arr, 0, mid-1)

```
int binarySearch(int arr[], int s, int e, int target){
//base case
    if (s > e){
        return -1;
    }
    int mid = s+(e-s)/2;
    if (arr[mid] == target){
        return mid;
    }
//recursive relation
    if (arr[mid] < target){
        return binarySearch(arr, mid+1, e, target);
    }
    else{
        return binarySearch(arr, s, mid-1, target);
    }
```

```
int main(){
    int arr[] = {10,20,30,40,50,60,70,80};
        int size = 8;
        int start = 0;
        int end = size-1;
    int target = 80;
int bool found = binarySearch(arr,start,end,target);

    if(found!=-1){
        cout<<"target found"<<found<<endl;
    }
    else{
        cout<<"target not found"<<endl;
    }
    return 0;
}
```



TC → O(log n).

Very Important Pattern

Include or exclude Pattern

⇒ Subsequences of string :-

A subsequence of a string is a sequence that can be derived from the given string by deleting zero or more elements without changing the order of the remaining elt.

abc

✓ x x → a
x ✓ x → b
x x ✓ → c
✓ ✓ x → ab
x ✓ ✓ → bc
✓ x ✓ → ac
✓ ✓ ✓ → abc
x x x → ""

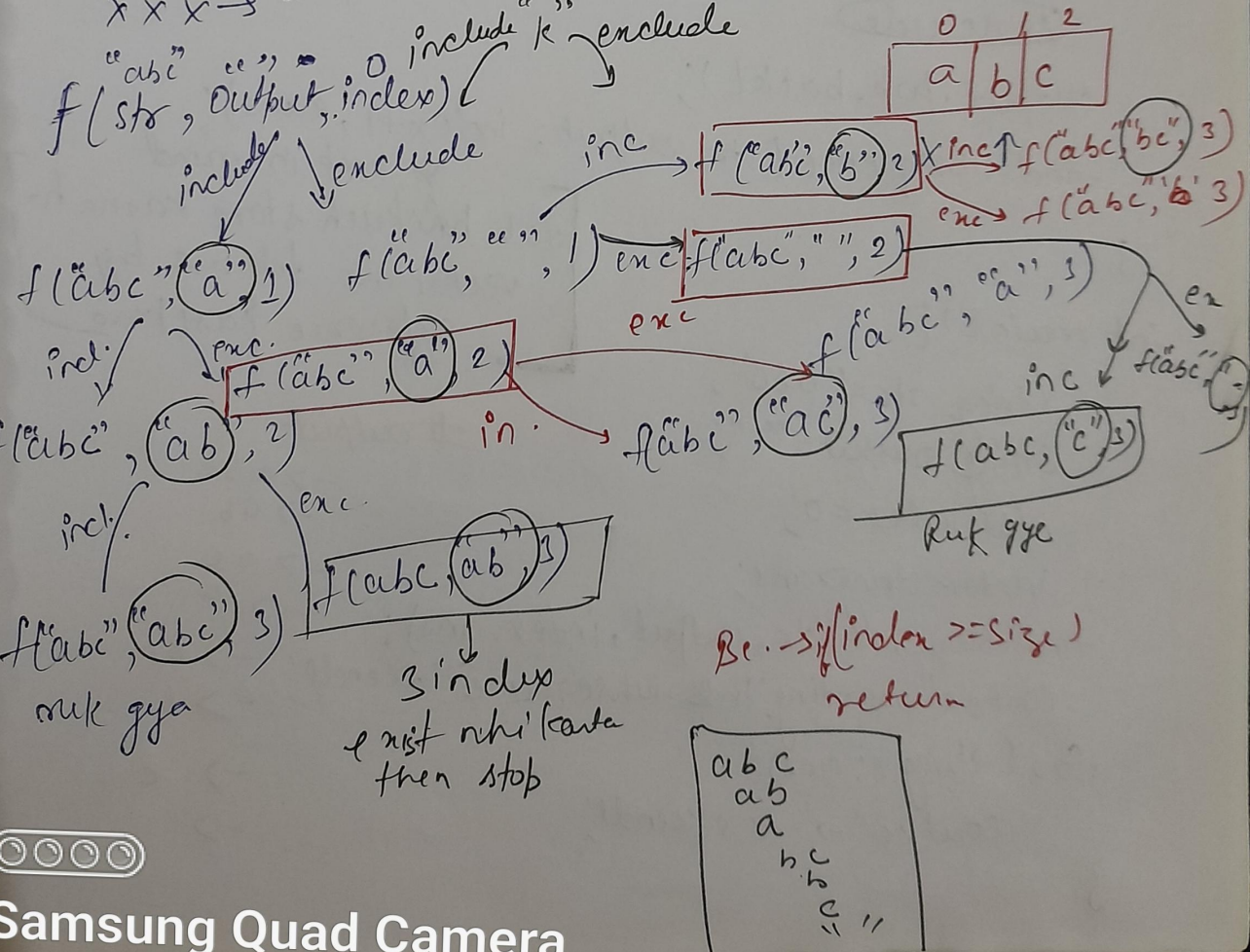
xy

✓ x → x
x ✓ → y
✓ ✓ → xy
x x → ""

n length
total subsequence = 2^n.

traverse karne ke liye index

f("abc", "", 0) include "k" exclude



```
void findSubsequence(string str, string output, int index,
                     vector<string> &ans) {
    if (index >= str.length()) {
        // ans jo h vo o/p string me build ho chuka hai
        // print krdo
        // cout << "->" << output << endl;
        ans.push_back(output);
        return;
    }
    char ch = str[index];
    // include
    output.push_back(ch);
    findSubsequence(str, output, index+1, ans);
    // exclude
    output.pop_back();
    findSubsequence(str, output, index+1, ans);
}
int main() {
    string str = "abc";
    string output = "";
    int index = 0;
    vector<string> ans;
    findSubsequence(str, output, index, ans);
    cout << "printing the subsequence" << endl;
    for (string s : ans) {
        cout << "->" << s << endl;
    }
}
```

keep in mind
Jab bhi kuch store karna ho vector me toh use by reference pass karo.

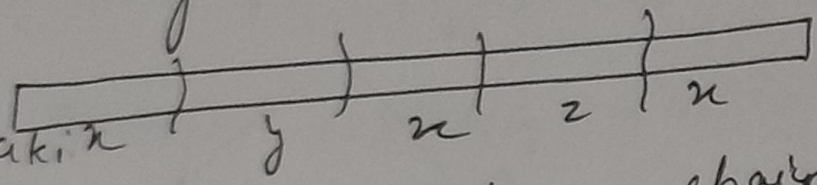
#output

```
->abc
->ab
->ac
->a
->bc
->b
->c
->
```

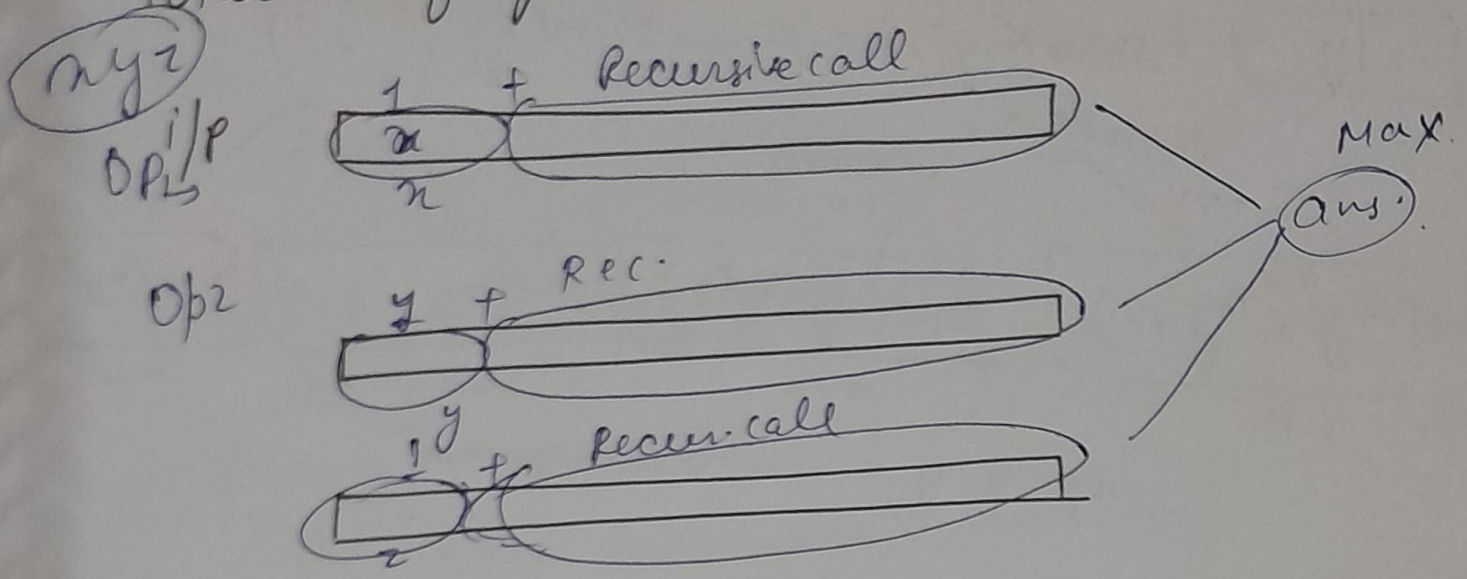
Ques Cut into Segments

Most Imp. Pattern Bhut Ques me use hoga

Is tarike se todna hai taki x y z

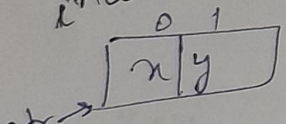


Total No of segment Max^m banane chaiye



Dry run:-

str = "xy"
output = ""
index = 0



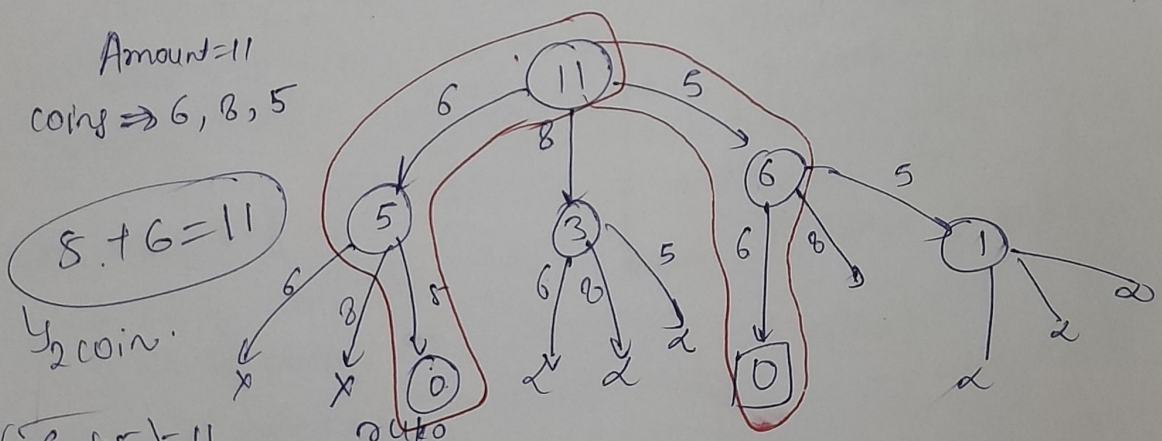
xy, x, y, "" → subsequence

Ques Coin change

I/P → coins = [1,2,5], amount = 11

O/P → 3

Minm no of coins me amount banana hai



f(11) → 1 coin 6 → f(5) → recu.

f(n) → f(11) → 1 coin → f(3) → recurs

f(n) → f(11) → 1 coin → f(6) → recurs

1 + f(n-6)

1 + f(3)

1 + f(6)

BS → if(n==0){
return 0;
}

coin value > amount

↓
no recursion call

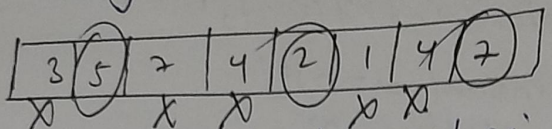
```
int solve(vector<int>& coins, int amount){
    if(amount == 0){
        return 0;
    }
    int mini = INT_MAX;
    int ans = INT_MAX;
    for(int i=0; i<coins.size(); i++){
        int coin = coins[i];
        if(coin <= amount){
            int recAns = solve(coins, amoun-coin);
            if(recAns != INT_MAX)
                ans = 1+recAns;
        }
        mini = min(mini, ans);
    }
    return mini;
}
int coinChange(vector<int>& coins, int amount){
    int ans = solve(coins, amount);
    if(ans == INT_MAX){
        return -1;
    }
    else{
        return ans;
    }
}
```

LC-198

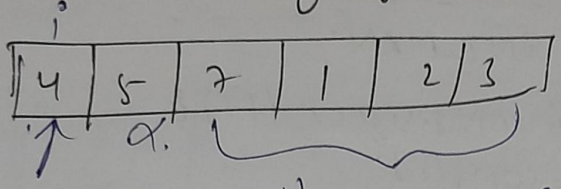
Ques House Robber

Bitmasking & subsequence

Sum of non-adjacent element



Chori is tarah karni hai taki Max amount of money you bna paaye



index > size ruk jaao

Chori: 4 + f(i+2, n-1) > max → final Ans.

Chori → 0 + f(i+1, n-1);

```
int solve(vector<int>& nums, int size, int index) {
    if (index >= size) {
        return 0;
    }
    int Op1 = nums[index] + solve(nums, size, index+2);
    int Op2 = 0 + solve(nums, size, index+1);
    int finalAns = max(Op1, Op2);
    return finalAns;
}
int rob(vector<int>& nums) {
    int size = nums.size();
    int index = 0;
    int ans = solve(nums, size, index);
    return 0;
}
```