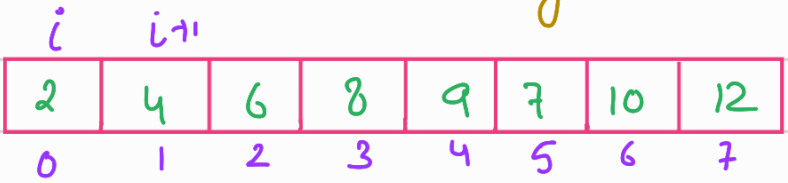
1) Problem → Check if array is sorted or not

I/p → { 2, 3, 4, 5, 6 } O/p → true
→ { 1, 4, 3, 7 } → O/p - False

solution → For 2 elements we will use following conditions to check whether array is sorted



→ if (a[i+1] > a[i])
move Foward
else
return False

base case → i == n-1

```cpp
#include <iostream>
#include <vector>
using namespace std;

bool checkSorted(vector<int> &arr, int&n , int i){
    //base case
    if(i == n-1){
        return true;
    }

    if(arr[i+1] < arr[i]){
        return false;
    }

    return checkSorted(arr, n, i+1);
}

int main()
{
    vector<int> v {10,20,5,50,60};
    int n = v.size();
    int i=0;

    bool isSorted = checkSorted(v , n, i);

    if(isSorted) {
        cout << "Array is sorted "<< endl;
    }else{
        cout << "Array is not sorted "<< endl;
    }

    return 0;
}

```

2) Problem → Binary Search using Recursion

```cpp
#include <iostream>
#include <vector>
using namespace std;

int binarySearch(vector<int> arr, int s, int e, int key){
    //base case
    //key not found
    if(s > e)
        return -1;

    int mid = (s+e)/2;

    //case 2 -> key found
    if(arr[mid] == key)
        return mid;

    // arr[mid] < key - right me search
    if(arr[mid] < key){
        return binarySearch(arr, mid+1, e, key);

    }else{
        //arr[mid] > key -> left me search
        return binarySearch(arr,s, mid-1, key);
    }

}

int main()
{
    vector<int> v {10,20,5,50,60};
    int n = v.size();
    int target = 50;
    int s = 0;
    int e = n-1;
    int ans = binarySearch(v,s,e,target);

    cout << "Answer is " << ans << endl;

    return 0;
}
```

4) Subsequence of a String

i/p → abc

o/p → print all substr → "", a, b, c, ab, bc, ac, abc

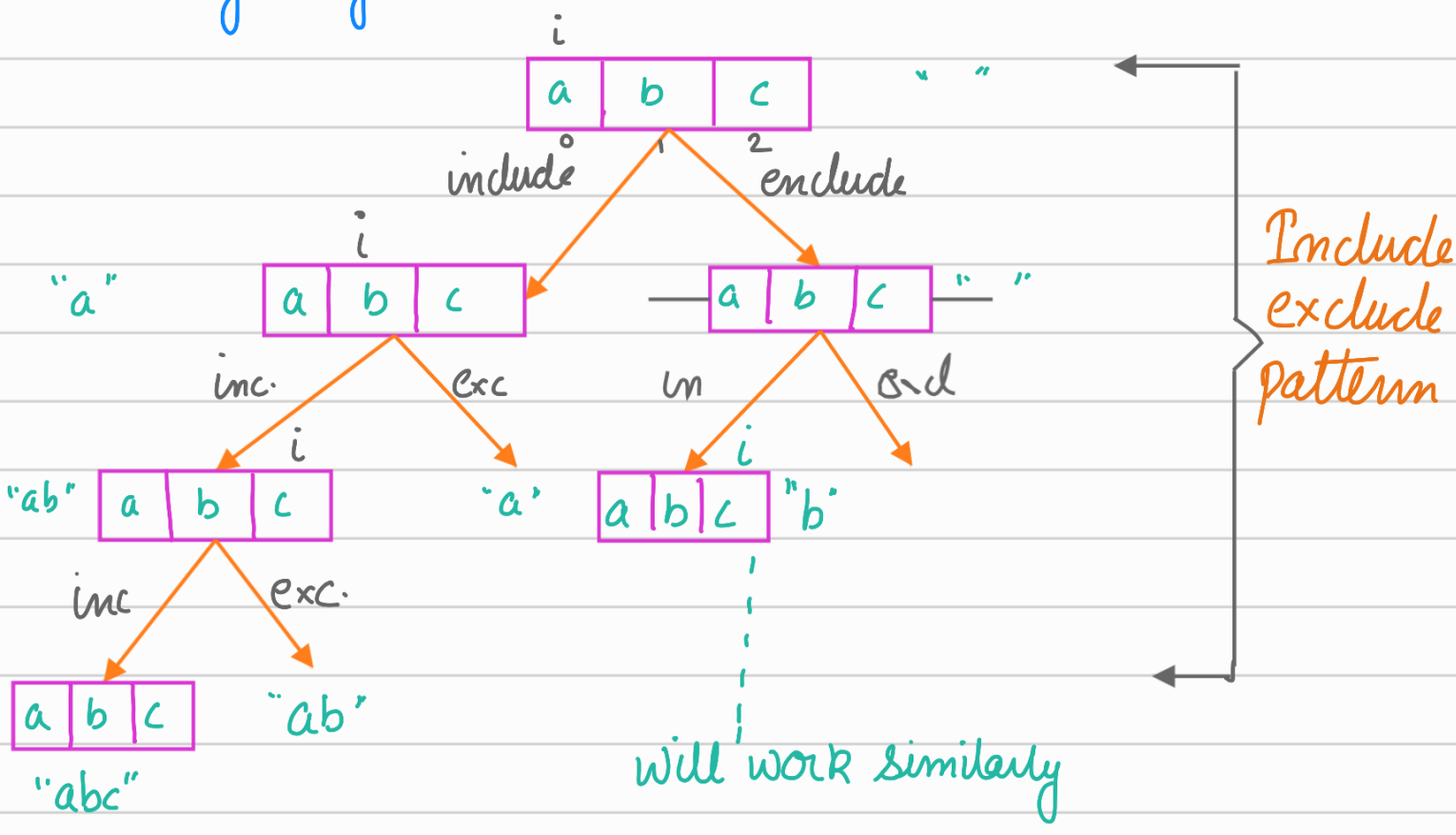
Eg → if i/p → ab

a	b	o/p
x	x	""
✓	x	a
x	✓	b
✓	✓	ab

This pattern is include-exclude pattern

→ For 2 char, we have 4 o/p
so for n=2, o/p → $2^2=4$
for n chars, o/p → 2^n

We will iterate over the string and take an empty string at any stage we will either include it or exclude.




```cpp
#include <iostream>
using namespace std;
void printSubSequence(string str,string output, int i){
    if(i >= str.length()){
        cout << output << endl;
        return;
    }
    //exclude
    printSubSequence(str, output,i+1);
    //include
    output.push_back(str[i]);
    printSubSequence(str, output,i+1);

}
int main()
{
    string str = "abc";
    string output = "";
    int i = 0;
    printSubSequence(str,output,i);
    return 0;
}
```