Recursion   Lecture 3

Q.

I/p → array

O/p → to tell array is sorted or not.

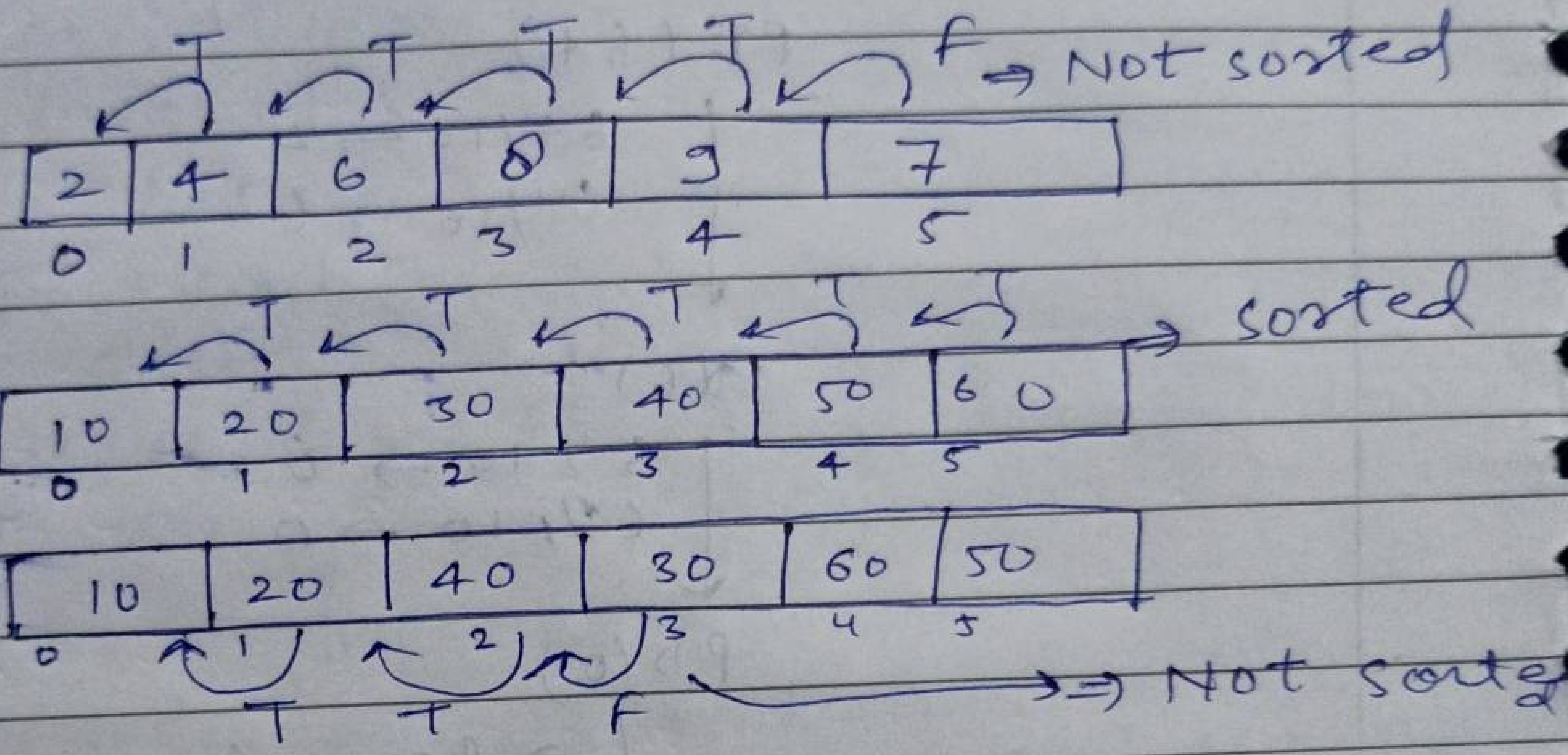
We have to return true or false

It must be sorted in ascending order.

e.g.

arr[i+1] > arr[i]
↳ aage badho

else
return false

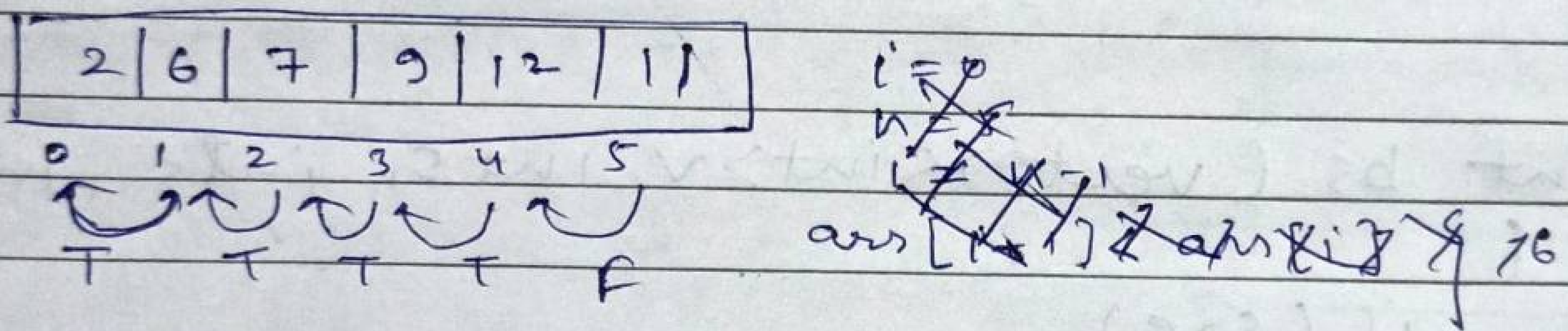


Code

```
int main() {
    vector<int> v{10,20,30,50,60};
   int n = v.size();
  bool sorted = checkSorted(v, n, i);
 if(sorted) {
      cout<< "Array is sorted"<<endl
    }
 else {
    cout<<"Not Sorted" <<endl;
   }
 return0;
}
```

```
bool checksorted (vector<int> & ~~arr~~ int& , n,
                                   int i) {
    if (i==n-1) {
        return true;
    }
   if (arr[i+1] < arr[i])
        return false;

   return checksorted (arr, n, i+1);
}
```

Dry Run on Test cases



└→ Array not sorted

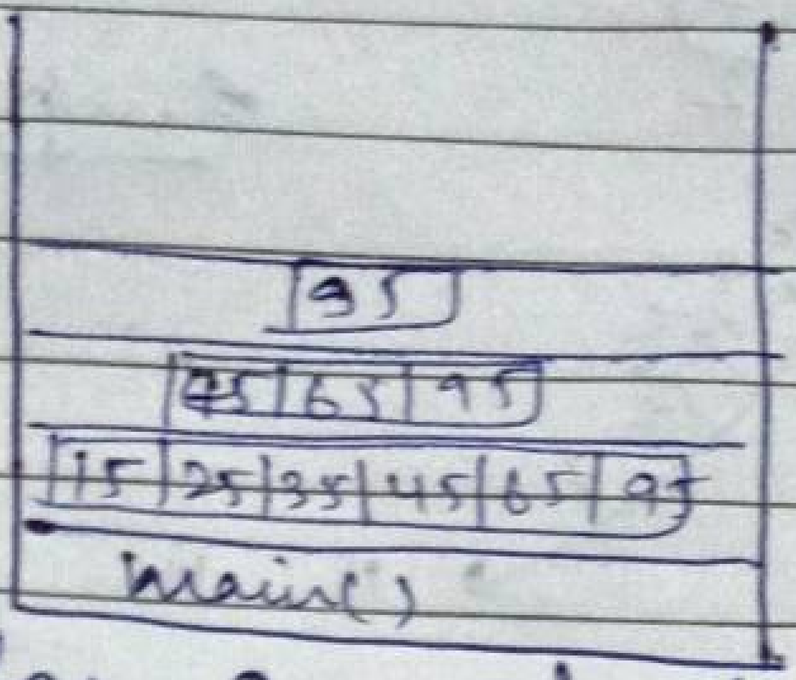
Binary search

```
            int bs (arr, n)
            {
                int s=0;
                int e=n-1;
                int mid = ( s+e / 2 ):

      while (s<=e) {
               if (arr[mid]==key)
                         return mid;
              if (arr [mid] < key)
                        s= mid+1;
              else
                     e = mid-1;
          ? mid= s+e/2
```

Code

```
int main(){
    vector<int> v {10,20,40,60,70,90,99};
 int target = 99;

 int n= v.size();
 int s=0;
 int e=n-1;
 int ans = bs(v, s,e, target);
 cout<<"Ans" << ans << endl;
 return0;
}

int bs (vector<int>v, int s, int e , int key)
 {
    if(s>e)
       return -1; → int mid= (s+e)/2;
   if (arr[mid] == key)
      return mid;

   if(arr[mid]< key){
   int ~~bool~~ ans = bs(v, mid+1, e, key);
       return ans;
     }

    else {
   int ~~bool~~ ans = bs(v, ~~ne~~ s, mid-1, key);
    return ans;
   }
}
```

Call stack



here for a single element 95 we have given 4 times 4 byte due to call by value. so we can ignore doing this by using call by reference.

Q.3) Subsequences of a string

i/p ⇒ "a b c"

o/p ⇒ print all subsequences.

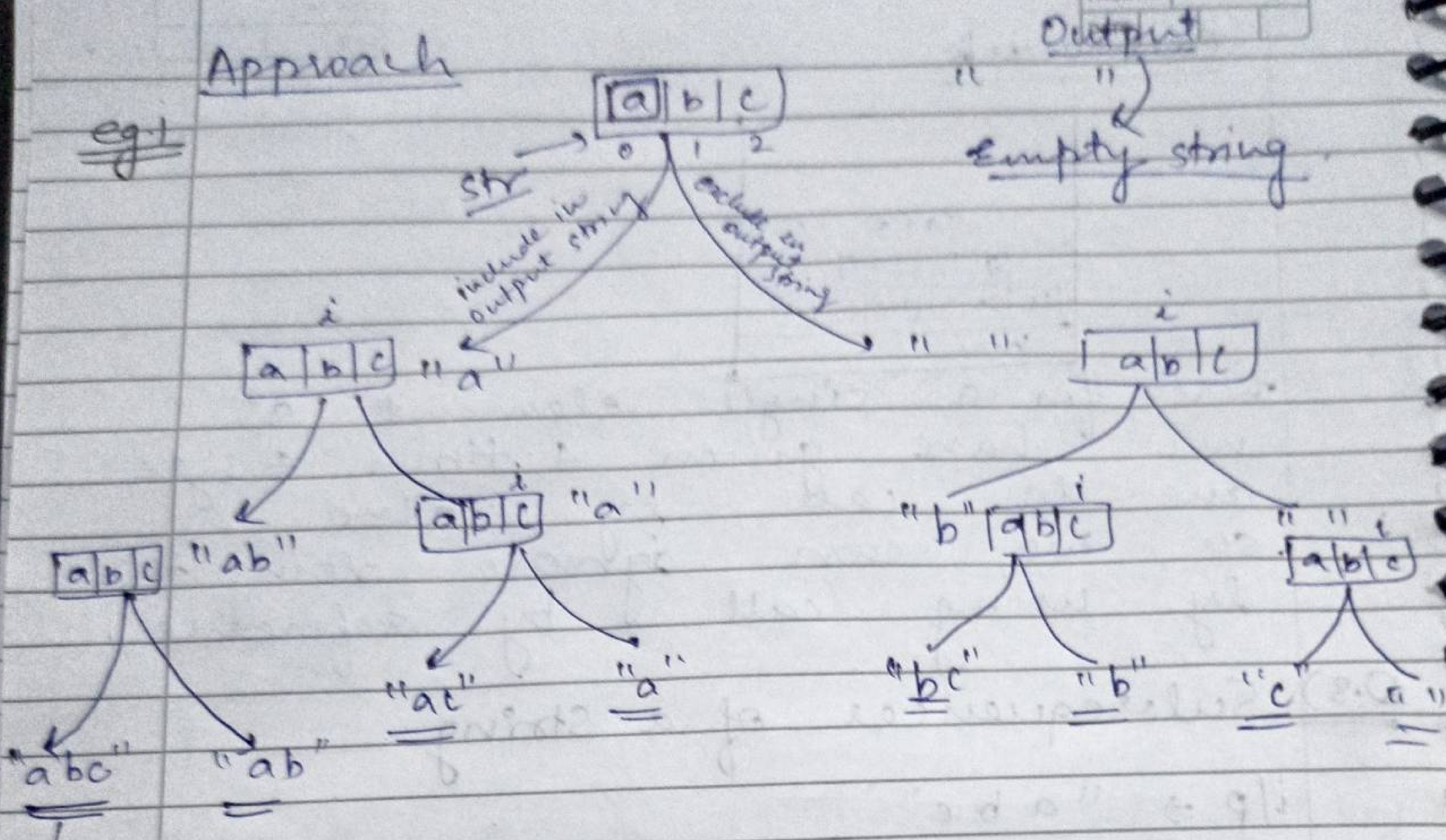
a	b	c	
✓	✓	✓	abc
✓	✓	x	ab
✓	x	x	b
x	x	x	—
x	✓	x	b
x	x	✓	c
x	✓	✓	bc
✓	x	✓	ac

{}, {abc} {ab} {b} {c}
{c} {bc} {ac}

include - exclude problem

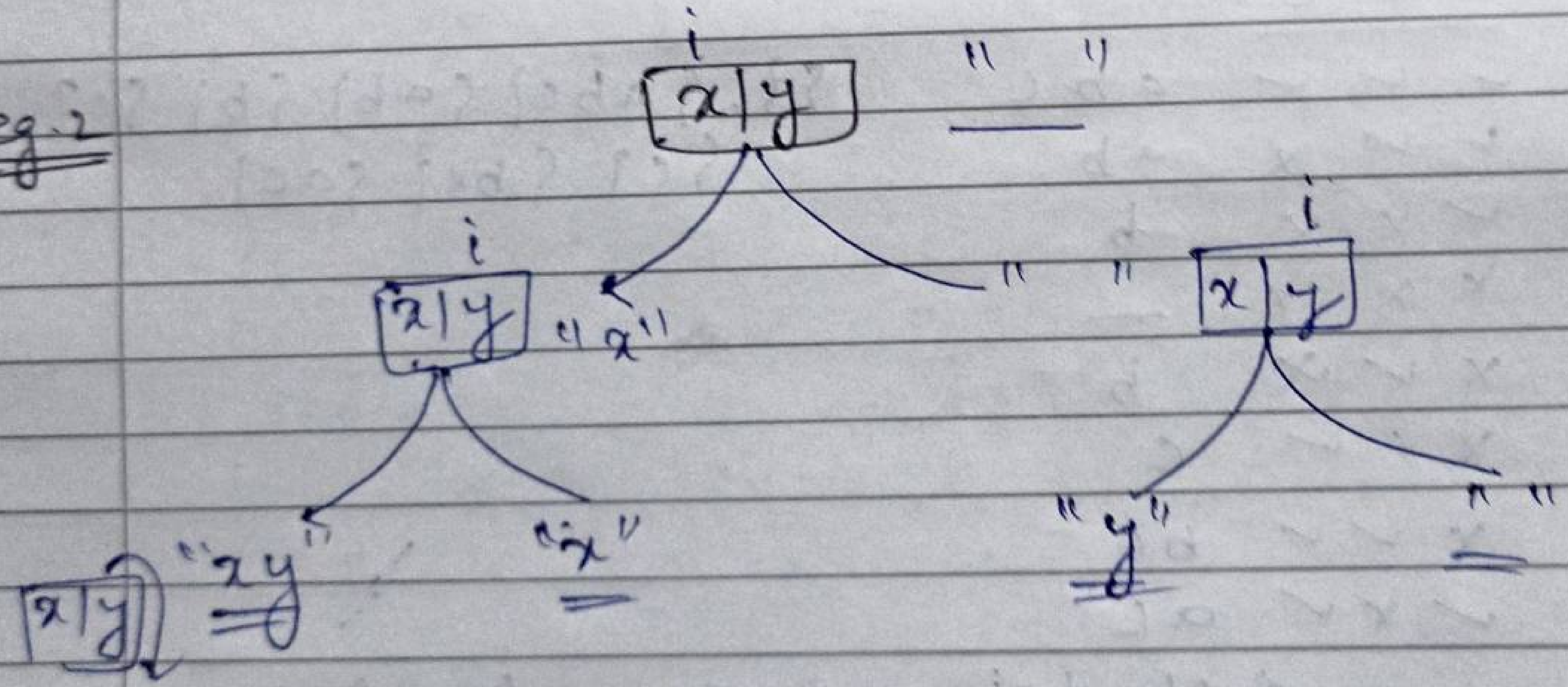
Approach

eg.1



Jb i bahar nikal jae to stop.

eg.2



i bahar aagya

Subsequence → Using bitmasking for without recursion.

Code

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

void printsubsequences(string str, string output
                        , int i) {

   if (i >= str.length()) {
         cout << output << endl;
          return;
      }
  // exclude
   printSubsequences(str, output, i+1);

  // include
   output.push_back(str[i]);
  printsubsequences(str, output, i+1);
}
```