# Print only one subsequence whose sum is k out of all subsequences available

```cpp
#include<bits/stdc++.h>
using namespace std;

bool printF( int index, vector<int> & ans, int sum, int target, int arr[], int size){

    // base case
    if( index == size){
        if( sum == target){
            for(auto i : ans){
                cout << i << " ";
            }
            cout << endl;

            return true;
        }

        return false;
    }


    ans.push_back(arr[index]);
    sum = sum + arr[index];

    if(printF( index + 1, ans, sum, target, arr, size)){
        return true;
    }

    ans.pop_back();
    sum = sum - arr[index];
    if(printF( index + 1, ans, sum, target, arr, size)){
        return true;
    }

    return false;


}


int main(){
    int arr[3] = { 1,2, 1};
    int size = 3;
    int target = 2;
    vector<int> ans;
    printF(0, ans, 0, target, arr, size );
    return 0;
}
```

Just print the data structure when the sum is achieved and then return true in the base case.

Now in the main function you just need to return true and false, true has printing property associated with it.

What is the essence ?

Essence is that we have made this printSubsequence function as the boolean function and the logic is that if the sum is achieved we return true, and along with this true we print the data structure, and this true just acts as the return medium

We could also use void ans do simply return ?
Yes we could do that but we chose bool as we also used if to segregate and move to the next include or not include and inside if we need true or false to execute the statement so we made simple logic that if ( ) means if expression inside it is true, then just return it recursively to end the function and print the subsequence and if ( false ) i.e. sum is not achieved move to next condition, which is checking of sum under pick and not pick.