35 Subsets finding

#include <stdio.h>

void findSubsets(int arr[], int n, int target, int current\_index, int current\_subset[], int all\_subsets[][n]) {

"""

Finds all subsets of the given array that sum up to the target sum using backtracking.

Args:

arr: The array of numbers.

n: The size of the array.

target: The target sum to find.

current\_index: The current index in the array.

current\_subset: The current subset being built.

all\_subsets: A 2D array to store all the found subsets.

"""

// Base case: If the target sum is reached, add the current subset to the results

if (target == 0) {

for (int i = 0; i < n; i++) {

all\_subsets[current\_index][i] = current\_subset[i];

}

current\_index++;

return;

}

// Base case: If the current index is out of bounds or the current sum exceeds the target, return

if (current\_index >= n || target < 0) {

return;

}

// Include the current element in the subset

current\_subset[current\_index] = 1;

findSubsets(arr, n, target - arr[current\_index], current\_index + 1, current\_subset, all\_subsets);

// Exclude the current element from the subset

current\_subset[current\_index] = 0;

findSubsets(arr, n, target, current\_index + 1, current\_subset, all\_subsets);

}

int main() {

int arr[] = {3, 2, 7, 1};

int target = 5;

int n = sizeof(arr) / sizeof(arr[0]);

int all\_subsets[n][n]; // Array to store all subsets (adjust size as needed)

int current\_subset[n] = {0}; // To keep track of the current subset

int current\_index = 0;

findSubsets(arr, n, target, current\_index, current\_subset, all\_subsets);

if (current\_index > 0) {

printf("All subsets that sum up to %d are:\n", target);

for (int i = 0; i < current\_index; i++) {

for (int j = 0; j < n; j++) {

if (all\_subsets[i][j] == 1) {

printf("%d ", arr[j]);

}

}

printf("\n");

}

} else {

printf("No subsets found that sum up to %d\n", target);

}

return 0;

}