**Subsequences & Subsets**

1. **Little Ponny and 2-Subsequence**

#include <stdio.h>

int count2Subsequences(int arr[], int n) {

int count = 0;

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

for (int j = i + 1; j < n; j++) {

if (arr[i] < arr[j]) {

count++;

}

}

}

return count;

}

int main() {

int n;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter the elements of the array:\n");

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

scanf("%d", &arr[i]);

}

int result = count2Subsequences(arr, n);

printf("Number of 2-subsequences: %d\n", result);

return 0;

}

1. **SUBARRAY OR**

#include <stdio.h>

int main() {

int n, threshold, count = 0;

printf("Enter the size of the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter the elements of the array: ");

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

scanf("%d", &arr[i]);

}

printf("Enter the threshold value: ");

scanf("%d", &threshold);

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

int bitwise\_or = arr[i];

if (bitwise\_or >= threshold) {

count++;

}

for (int j = i + 1; j < n; j++) {

bitwise\_or |= arr[j];

if (bitwise\_or >= threshold) {

count++;

} else {

break;

}

}

}

printf("The number of non-empty subarrays with bitwise OR greater than or equal to %d is %d\n", threshold, count);

return 0;

}

1. **Subset Problem**

#include <stdio.h>

int isSubsetSumPossible(int set[], int n, int sum) {

int dp[n + 1][sum + 1];

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

dp[i][0] = 1; // Subset with sum 0 is always possible

}

for (int i = 1; i <= sum; i++) {

dp[0][i] = 0; // No subset is possible without any elements

}

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= sum; j++) {

if (j < set[i - 1]) {

dp[i][j] = dp[i - 1][j];

} else {

dp[i][j] = dp[i - 1][j] || dp[i - 1][j - set[i - 1]];

}

}

}

return dp[n][sum];

}

int main() {

int n, sum;

printf("Enter the number of elements in the set: ");

scanf("%d", &n);

int set[n];

printf("Enter the elements of the set:\n");

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

scanf("%d", &set[i]);

}

printf("Enter the target sum: ");

scanf("%d", &sum);

if (isSubsetSumPossible(set, n, sum)) {

printf("True");

} else {

printf("False");

}

return 0;

}

1. **Perfect Sum Problem**

#include <stdio.h>

int countSubsetsWithSum(int arr[], int n, int sum) {

int dp[n + 1][sum + 1];

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

dp[i][0] = 1; // Empty subset is always possible

}

for (int i = 1; i <= sum; i++) {

dp[0][i] = 0; // No subset is possible without any elements

}

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= sum; j++) {

if (j < arr[i - 1]) {

dp[i][j] = dp[i - 1][j];

} else {

dp[i][j] = dp[i - 1][j] + dp[i - 1][j - arr[i - 1]];

}

}

}

return dp[n][sum];

}

int main() {

int n, sum;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter the elements of the array:\n");

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

scanf("%d", &arr[i]);

}

printf("Enter the target sum: ");

scanf("%d", &sum);

int result = countSubsetsWithSum(arr, n, sum);

printf("Number of subsets with sum %d: %d\n", sum, result);

return 0;

}

1. **Two sequences are subsequences of one another**

#include <stdio.h>

int isSubsequence(int seqA[], int sizeA, int seqB[], int sizeB) {

int i = 0, j = 0;

while (i < sizeA && j < sizeB) {

if (seqA[i] == seqB[j]) {

i++;

}

j++;

}

return (i == sizeA);

}

int main() {

int sizeA, sizeB;

printf("Enter the size of sequence A: ");

scanf("%d", &sizeA);

int seqA[sizeA];

printf("Enter the elements of sequence A:\n");

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

scanf("%d", &seqA[i]);

}

printf("Enter the size of sequence B: ");

scanf("%d", &sizeB);

int seqB[sizeB];

printf("Enter the elements of sequence B:\n");

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

scanf("%d", &seqB[i]);

}

if (isSubsequence(seqA, sizeA, seqB, sizeB)) {

printf("True");

} else if (isSubsequence(seqB, sizeB, seqA, sizeA)) {

printf("True");

} else {

printf("False");

}

return 0;

}