Subset Sum

The Subset Sum problem involves searching through a collection of numbers to **find a subset that sums to a certain number.**

Problem statement:

– Input:

• A collection of non-negative integers A

• A non-negative integer b

– Output:

• Boolean value indicating whether some subset of the collection sums to b

e.g.

A = {5, 2, 1, 3} , b = 9

True , {5, 1, 3} = 5 + 1 + 3 = 9

Note : Order of integers in the subset doesn't matter so {5, 1, 3} and {3, 1, 5} are same.

Approach :

1. **Naive Recursive**
2. **Backtracking**
3. **Dynamic Programming**

To get to the solution to the actual problem (given a set of non-negative integers {5, 2, 1, 3} , is there a subset whose sum will result to 9), we will first solve the subproblems.

e.g.

• Given the set {5} , is there a subset whose sum will result to 2? (answer is false)

• Given the set {5} , is there a subset whose sum will result to 5? (answer is true {5})

• Given the set {5, 2} , is there a subset whose sum will result to 2? (answer is true {2})

• Given the set {5, 2, 1} , is there a subset whose sum will result to 6? (answer is true {5,1})

S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}										
1 {5}										
2 {5,2}										
3 {5,2,1}										
4 {5,2,1,3}										

When we don't consider any number from the set, obviously **there will be no subset**, whose subset sum will be greater than 0.

S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}		F	F	F	F	F	F	F	F	F
1 {5}										
2 {5,2}										
3 {5,2,1}										
4 {5,2,1,3}										

If we have to find a subset whose subset sum is 0, we can do that (by not including any integer in the subset), so regardless of how many integers from the set we consider, we can always find a subset whose subset sum is 0.

S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T									
2 {5,2}	T									
3 {5,2,1}	T									
4 {5,2,1,3}	T									

When we consider only the first number from the set {5} S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T									
3 {5,2,1}	T									
4 {5,2,1,3}	T									

When we consider only the first two numbers from the set {5, 2} S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T									
4 {5,2,1,3}	T									

When we consider only the first three numbers from the set {5, 2, 1} S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T	T	T	T	F	T	T	T	T	F
4 {5,2,1,3}	T									

When we consider all numbers from the set {5, 2, 1, 3} S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T	T	T	T	F	T	T	T	T	F
4 {5,2,1,3}	T	T	T	T	T	T	T	T	T	T

When we can't include the i^{th} number to the subset , because i^{th} number is greater than subset sum j

dpTable[i][j] = dpTable[i - 1] [j]

S = **{5, 2, 1, 3}** , Sum = **9**

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T	T	T	T	F	T	T	T	T	F
4 {5,2,1,3}	T	T	T	T	T	T	T	T	T	T

e.g.

dpTable[1][1] = False, dpTable[1][2] = False, dpTable[1][3] = False, dpTable[1][4] = False

dpTable[2][1] = False

When we can include the i^{th} number to the subset, because i^{th} number is not greater than subset sum j

1. If we can achieve a subset with subset sum j, by just using i-1 numbers, then we can definitely achieve subset sum j , by using i numbers.

   e.g. S = {5, 2, 1} Sum = 6 => {5, 1}

   e.g. S = {5, 2, 1, 3} Sum = 6 => {5, 1}

   which means if dpTable[i - 1][j] is true , then dpTable[i][j] will also be true.

S = **{5, 2, 1, 3}** , Sum = **9**

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T	T	T	T	F	T	T	T	T	F
4 {5,2,1,3}	T	T	T	T	T	T	T	T	T	T

2. dpTable[i][j] = dpTable[i - 1][j - i^{th} number]

   e.g.

   dpTable[1][6] = dpTable[0][1] = False

   dpTable[2][7] = dpTable[1][5] = True

   dpTable[4][9] = dpTable[3][6] = True

But which numbers we included in the subset to make the subset sum 9 ? S = {5, 2, 1, 3} , Sum = 9

Numbers/Sum	0	1	2	3	4	5	6	7	8	9
0 {}	T	F	F	F	F	F	F	F	F	F
1 {5}	T	F	F	F	F	T	F	F	F	F
2 {5,2}	T	F	T	F	F	T	F	T	F	F
3 {5,2,1}	T	T	T	T	F	T	T	T	T	F
4 {5,2,1,3}	T	T	T	T	T	T	T	T	T	T

So the subset will be {3, 1, 5}

Implementation

```
public class App {

	public static void main(String[] args) {
		int[] numbers = { 5, 2, 1, 3 };
		int sum = 9;

		SubsetSumProblem subsetSumProblem = new SubsetSumProblem(numbers, sum);
		subsetSumProblem.solveProblem();
		if (subsetSumProblem.hasSolution()) {
			subsetSumProblem.printSolution();
		}
	}
}
```

```java
public class SubsetSumProblem {

	private boolean[][] dpTable;
	private int[] numbers;
	private int sum;

	public SubsetSumProblem(int[] numbers, int sum) {
		this.numbers = numbers;
		this.sum = sum;
		this.dpTable = new boolean[numbers.length + 1][sum + 1];
	}

	public void solveProblem() {
		. . .
	}

	public boolean hasSolution() {
		. . .
	}

	public void printSolution() {
		. . .
	}
}
```

```
public void solveProblem() {

	for(int j=1; j <=this.sum; j++){
			this.dpTable[0][j] = false;
		}

	for (int i = 0; i <= this.numbers.length; i++) {
			this.dpTable[i][0] = true;
		}

	for (int rowIndex = 1; rowIndex <= numbers.length; ++rowIndex) {
	    for (int columnIndex = 1; columnIndex <= sum; ++columnIndex) {

		if (numbers[rowIndex - 1] > columnIndex) {
			this.dpTable[rowIndex][columnIndex] =
					this.dpTable[rowIndex - 1][columnIndex];
		} else {
		   if (this.dpTable[rowIndex - 1][columnIndex]) {
		        this.dpTable[rowIndex][columnIndex] = true;
		   } else {
		        this.dpTable[rowIndex][columnIndex] =
			this.dpTable[rowIndex - 1][columnIndex - numbers[rowIndex - 1]];
		   }
		}
	    }
	}
}
```

```
public boolean hasSolution() {

	if (this.dpTable[numbers.length][sum]) {
			System.out.println("A subset with sum "+ sum + " exists.");
			return true;
	} else {
			System.out.println("No feasible solution");
			return false;
	}
}
```

```
public void printSolution() {

	int columnIndex = this.sum;
	int rowIndex = this.numbers.length;

	while (columnIndex > 0 || rowIndex > 0) {

	   if (this.dpTable[rowIndex][columnIndex] ==
	            this.dpTable[rowIndex - 1][columnIndex]) {
	                  rowIndex = rowIndex - 1;
	   } else {
	       System.out.println("We include number: " + numbers[rowIndex - 1]);
	       columnIndex = columnIndex - numbers[rowIndex - 1];
	       rowIndex = rowIndex - 1;
	    }
	}
}
```