### Algorithm: Subset Sum Problem (Dynamic Programming)

To unlock the Gate of Eternal Wisdom, Sir Cedric and Ember needed to find a subset of ancient runes that summed to a specific magical number using the Subset Sum algorithm.

#### Initialize Data Structures:

* Sir Cedric used a mystical grid (2D array) to track possible sums.

#### Build the Solution:

* He evaluated each rune, updating the grid based on achievable sums.

#### Retrieve the Result:

* The grid revealed whether the magical number could be achieved.

#### Implementation:

| **def** subset\_sum(runes: List[int], target: int) -> bool:  n = len(runes)  dp = [[**False** **for** \_ **in** range(target + 1)] **for** \_ **in** range(n + 1)]  **for** i **in** range(n + 1):  dp[i][0] = **True**  **for** i **in** range(1, n + 1):  **for** j **in** range(1, target + 1):  **if** runes[i - 1] <= j:  dp[i][j] = dp[i - 1][j] **or** dp[i - 1][j - runes[i - 1]]  **else**:  dp[i][j] = dp[i - 1][j]  **return** dp[n][target]  *# Example usage:*  runes = [3, 34, 4, 12, 5, 2]  target = 9  print(subset\_sum(runes, target)) *# Output: True* |
| --- |

#### Explanation:

Initialize:

* dp: A mystical grid to track possible sums.

Build the Solution:

* Sir Cedric evaluated each rune, updating the grid based on achievable sums.

Retrieve the Result:

### The grid revealed whether the magical number could be achieved.