پیدا کردن مجموع

این مساله شبیه سؤال کوله پشتی صفر و یک است، زیرا در هر دو مساله ما به دنبال این هستیم که مجموع برابر عدد مشخصی شود. البته در این مساله ارزشی برای حداکثر کردن وجود ندارد.
برای تعریف کردن زیرمساله به دو متغیر i به عنوان ایندکس آخرین عددی از آرایه که در نظر می‌گیریم و j به عنوان جمع اعداد نیاز داریم. dp[i][j] را اینگونه تعریف می‌کنیم که آیا می‌توان مجموع j را با استفاده از i عنصر اول بسازیم. جواب مساله dp[n][k] است.
برای نوشتن رابطه بازگشتی روی برداشتن و برنداشتن عنصر i ام حالت‌بندی می‌کنیم. اگر عنصر برنداریم:

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

اگر برداریم:

```
dp[i][j] = dp[i-1][j - arr[i-1]
```

پس رابطه بازگشتی به این صورت می‌شود:

```
dp[i][j] = dp[i-1][j] or dp[i-1][j - arr[i-1]]
```

حالت‌های پایه:

```
dp[i][0] = true
```

چون با برنداشتن هیچ عنصری می‌توانیم مجموع صفر را بسازیم.

```
dp[0][i] = false
```

با صفر عنصر هیچ مجموعی را نمی‌توانیم بسازیم.

```
function sum_knapsack(arr, k):
    n = size of arr
    dp = 2D array with size (n+1, k+1)

    for I from 0 to n:
        dp[i][0] = True

    for j from 1 to k:
        dp[0][j] = False

    for I from 1 to n:
        for j from 1 to k:
            if( j > arr[i-1] ):
                dp[i][j] = dp[i-1][j] or dp[i-1][j-arr[i-1]]
            else
                dp[i][j] = dp[i-1][j]
    return dp[n][k]
```