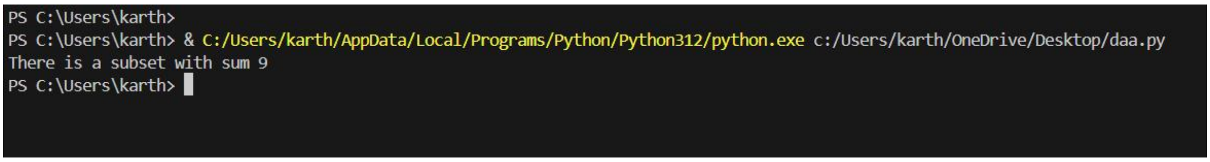
9. Meet in middle technique.

Code:

```
from itertools import combinations
def subset_sum(arr,target):
    n = len(arr)
    if n == 0:
        return target==0
    mid=n//2
    left_half=arr[:mid]
    right_half=arr[mid:]
    left_sums=set()
    right_sums=set()
    for i in range(len(left_half) + 1):
        for comb in combinations(left_half, i):
            left_sums.add(sum(comb))
    for i in range(len(right_half) + 1):
        for comb in combinations(right_half, i):
            right_sums.add(sum(comb))
    for s in left_sums:
        if (target - s) in right_sums:
            return True
    return False
arr = [3, 34, 4, 12, 5, 2]
target = 9
if subset_sum(arr, target):
    print(f"There is a subset with sum {target}")
else:
    print(f"There is no subset with sum {target}")
```

output:



```
PS C:\Users\karth>
PS C:\Users\karth> & C:/Users/karth/AppData/Local/Programs/Python/Python312/python.exe c:/Users/karth/OneDrive/Desktop/daa.py
There is a subset with sum 9
PS C:\Users\karth> 
```

Time complexity:

F(n)=o(2^n)