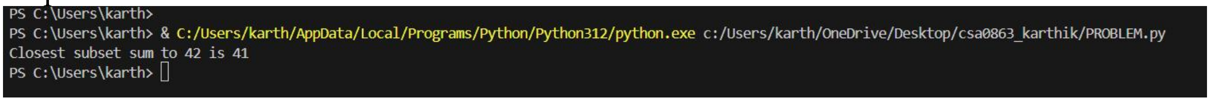
1. 167. Write a program to implement Meet in the Middle Technique. Given an array of integers and a target sum, find the subset whose sum is closest to the target. You will use the Meet in the Middle technique to efficiently find this subset.

   a) Set[] = {45, 34, 4, 12, 5, 2}   Target Sum : 42

code:

```
from itertools import combinations
from bisect import bisect_left
def generate_subsets(arr):
    subsets = []
    n = len(arr)
    for i in range(n + 1):
        for combo in combinations(arr, i):
            subsets.append(sum(combo))
    return subsets
def meet_in_the_middle(arr, target):
    n = len(arr)
    left_half = arr[:n//2]
    right_half = arr[n//2:]
    left_sums = generate_subsets(left_half)
    right_sums = generate_subsets(right_half)
    right_sums.sort()
    closest_sum = float('inf')
    closest_pair = (None, None)
    for s in left_sums:
        remaining = target - s
        pos = bisect_left(right_sums, remaining)
        if pos < len(right_sums):
            current_sum = s + right_sums[pos]
            if abs(target - current_sum) < abs(target - closest_sum):
                closest_sum = current_sum
                closest_pair = (s, right_sums[pos])
        if pos > 0:
            current_sum = s + right_sums[pos - 1]
            if abs(target - current_sum) < abs(target - closest_sum):
                closest_sum = current_sum
                closest_pair = (s, right_sums[pos - 1])
    return closest_sum
arr = [45, 34, 4, 12, 5, 2]
target = 42
output = meet_in_the_middle(arr, target)
print(f"Closest subset sum to {target} is {output}")
```

output:



```
PS C:\Users\karth>
PS C:\Users\karth> & C:/Users/karth/AppData/Local/Programs/Python/Python312/python.exe c:/Users/karth/OneDrive/Desktop/csa0863_karthik/PROBLEM.py
Closest subset sum to 42 is 41
PS C:\Users\karth> 
```

Time complexity:

F(n)=o($2^{n/2}$)