

169) Write a program to implement Meet in the Middle Technique. Given a large array of integers and an exact sum E, determine if there is any subset that sums exactly to E. Utilize the Meet in the Middle technique to handle the potentially large size of the array. Return true if there is a subset that sums exactly to E, otherwise return false.

a) E = {1, 3, 9, 2, 7, 12} exact Sum = 15

b) E = {3, 34, 4, 12, 5, 2} exact Sum = 15

AIM : To Return true if there is a subset that sums exactly to E, otherwise return false.

Program:

```
from itertools import chain, combinations
```

```
def subsets(arr):
```

```
    """Generate all possible subsets of a list."""
```

```
    return chain(*map(lambda x: combinations(arr, x), range(len(arr) + 1)))
```

```
def subset_sum(arr, target):
```

```
    """Check if there is a subset that sums to the target using Meet in the Middle."""
```

```
    n = len(arr)
```

```
    left = arr[:n//2]
```

```
    right = arr[n//2:]
```

```
    left_sums = {sum(subset) for subset in subsets(left)}
```

```
    right_sums = {sum(subset) for subset in subsets(right)}
```

```
    if target in left_sums or target in right_sums:
```

```
        return True
```

```
    for s in left_sums:
```

```
        if (target - s) in right_sums:
```

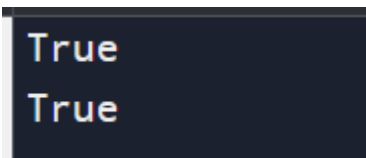
```
            return True
```

```
    return False
```

```
E1 = [1, 3, 9, 2, 7, 12]
```

```
exact_sum1 = 15
```

```
print(subset_sum(E1, exact_sum1))
```



```
True
True
```

OUTPUT:

TIME COMPLEXITY: $O(2^n)$