1.问题：
对于给定集合 A，求其所有的子集合，使得其子集合的所有元素之和为值 k。

2.算法：
对于集合 A，遍历其所有的子集，在子集中遍历所有元素求和，若和为给定值 k，则加入到结果集当中。
遍历子集时为避免重复遍历可以利用二进制数的特点：将数组 A 标记为为一个二进制数，数组 A 的长度即为二进制数的位数。可设原数组为全 1 二进制数，在子集中，没有取到的数字对应的二进制位上为 0. 如 A=[1,2,3] 可标记为“111”，则其子集[1,3]则标记为“101”，以此类推。

3.代码：
```
def subset_sum(array,k = 0):
    result = []
    for i in range(1,2**len(array)):
        res = mask(array,bin(i)[2:])
        if sum(res) == k:
            result.append(res)
    return result

def mask(arr,m):
    em = m.zfill(len(arr))
    nlist = []
    for i,v in enumerate(em):
        if v != "0":
            nlist.append(arr[i])
    return nlist
```

4.示例：
```
a = [0,1,2,6,-6,-3,-4,-7]
print(subset_sum(a))
```

输出：
[[6, -6], [1, 6, -7], [1, 6, -3, -4], [1, 2, -3], [1, 2, 6, -6, -3], [0], [0, 6, -6], [0, 1, 6, -7], [0, 1, 6, -3, -4], [0, 1, 2, -3], [0, 1, 2, 6, -6, -3]]