18)PARTITION ARRAY INTO K SUBSETS OF EQUAL SUM:

METHOD:

1) This sum is same as that of the partitioning array into two subsets with equal sum.
2) Here we first find out the sum of all the elements in the array and check if it is divisible by k or not.If it is divisible by k then partitioning is possible else it is not possible.
3) If sum is divisible by k then one more optimization is that we can check whether there is an element whose value is greater than the sum/k.If there is an element then also the partitioning is not possible.
4) If there is no element greater than the sum/k then the partitioning process starts.
5) We place all the elements in each and every position possible under the condition that the subsetsum[i]+a[index] must not be greater than the sum/k.

CODE OF THE ABOVE METHOD:

```
class Solution{
  public:
    bool check(vector<int> &subsetsum){
        for(int i=1;i<subsetsum.size();i++){
            if(subsetsum[i]!=subsetsum[i-1]){
                return false;
            }
        }
        return true;
    }
    bool fun(int a[],int index,int n,int reqsum,vector<int> &subsetsum){
        if(index==n){
            return check(subsetsum);
        }
        else{
            bool status=false;
            for(int i=0;i<subsetsum.size();i++){
                if(subsetsum[i]+a[index]<=reqsum){
                    subsetsum[i]=subsetsum[i]+a[index];
                    status=status||fun(a,index+1,n,reqsum,subsetsum);
                    subsetsum[i]=subsetsum[i]-a[index];
                }
            }
            return status;
        }
    }
    bool isKPartitionPossible(int a[], int n, int k){
        int sum=0;
        for(int i=0;i<n;i++){
            sum=sum+a[i];
        }
        if(sum%k!=0){
            return false;
        }
```

```
        else{
            vector<int> subsetsum(k);
            for(int i=0;i<n;i++){
                if(a[i]>sum/k){
                    return false;
                }
            }
            return fun(a,0,n,sum/k,subsetsum);
        }
    }
};
```