```c
/**
 * Return an array of arrays of size *returnSize.
 * The sizes of the arrays are returned as *returnColumnSizes array.
 * Note: Both returned array and *columnSizes array must be malloced, assume caller calls free().
 */
#define SOLUTION_FIRST          (1)
#define SOLUTION_SECOND         (0)


#if(SOLUTION_FIRST)
void _subsets(int* nums, int numsSize, int* returnSize, int** returnColumnSizes, int pos, int tmp_index, int** result, int* tmp) {

    if(tmp_index >= numsSize)
    {
        result[*returnSize] = (int*)malloc(sizeof(int)*(pos+1));
        memcpy(result[*returnSize], tmp, sizeof(int)*(pos+1));
        (*returnColumnSizes)[*returnSize] = pos+1;
        (*returnSize)++;
    }else
    {
        tmp[pos+1] = nums[tmp_index];
        _subsets(nums, numsSize, returnSize, returnColumnSizes, pos+1, tmp_index+1, result, tmp);
        _subsets(nums, numsSize, returnSize, returnColumnSizes, pos, tmp_index+1, result, tmp);
    }
}

int** subsets(int* nums, int numsSize, int* returnSize, int** returnColumnSizes) {
    int** result;
    int* tmp;

    tmp = (int*)malloc(sizeof(int)*numsSize);
    result = (int**)malloc(sizeof(int*) * (1<<numsSize));
    *returnColumnSizes = (int*)malloc(sizeof(int) * (1<<numsSize));
    *returnSize = 0;
    _subsets(nums, numsSize, returnSize, returnColumnSizes, -1, 0, result, tmp);

    return result;
}
#elif(SOLUTION_SECOND)
void _subsets(int* nums, int numsSize, int* returnSize, int** returnColumnSizes, int pos, int tmp_index, int** result, int* tmp) {

    if(tmp_index >= numsSize)
    {
        result[*returnSize] = (int*)malloc(sizeof(int)*(pos+1));
        memcpy(result[*returnSize], tmp, sizeof(int)*(pos+1));
        (*returnColumnSizes)[*returnSize] = pos+1;
        (*returnSize)++;
    }else
    {
        tmp[pos+1] = nums[tmp_index];
        _subsets(nums, numsSize, returnSize, returnColumnSizes, pos+1, tmp_index+1, result, tmp);
        _subsets(nums, numsSize, returnSize, returnColumnSizes, pos, tmp_index+1, result, tmp);
    }
}

int** subsets(int* nums, int numsSize, int* returnSize, int** returnColumnSizes) {
    int** result;
    int* tmp;
    int index;
    int bit_index;
    int tmp_index;

    tmp = (int*)malloc(sizeof(int)*numsSize);
    result = (int**)malloc(sizeof(int*) * (1<<numsSize));
    *returnColumnSizes = (int*)malloc(sizeof(int) * (1<<numsSize));
    *returnSize = 0;


    for(index = 0; index < (1<<numsSize); index++)
    {
        tmp_index = 0;

        for(bit_index = 0; bit_index < numsSize; bit_index++)
        {
            if( (index & (1 << bit_index)) )
            {
                 tmp[tmp_index] = nums[bit_index];
                 tmp_index++;
            }

        }

        result[*returnSize] = (int*)malloc(sizeof(int)*(tmp_index));
        memcpy(result[*returnSize], tmp, sizeof(int)*(tmp_index));
        (*returnColumnSizes)[*returnSize] = tmp_index;
        (*returnSize)++;
    }

    return result;
}
#endif
```