int cmp(const void \*a,const void \*b){

    return \*(int\*)a -\*(int\*)b;

}

/\*\*

 \* 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().

 \*/

int\*\* threeSum(int\* nums, int numsSize, int\* returnSize, int\*\* returnColumnSizes){

    int\*\* ans=malloc(sizeof(int\*)\*1);

    int c=0;

    \* returnColumnSizes=NULL;

    \* returnColumnSizes=realloc(\*returnColumnSizes,sizeof(int)\*(c+1));

    \* returnSize=c;

    int i=0;

    int k=i+1;

    int j=numsSize-1;

    int check=0;

    int mid=k;

    //printf("%d \n",numsSize);

    qsort(nums,numsSize,sizeof(int),cmp);

/\*\*\*

    for(int m = 0 ; m < numsSize ; m++){

        printf("%d ",nums[m]);

    }

    printf("\n");

\*\*\*/

    while(1){

        int sum=nums[i]+nums[k]+nums[j];

        //printf("i=%d k=%d j=%d mid=%d\n",i,k,j,mid);

        if(\* returnSize >0 ){

            //if( ans[\*returnSize-1][0] == nums[i] &&  ans[\*returnSize-1][1] == nums[k] &&  ans[\*returnSize-1][2] == nums[j]){

                //printf("check1=%d\n",check);

            //    check=1;

            //}

            for(int ii=\*returnSize-1 ; ii >=0 ; ii-- ){

                printf("%d ",\* returnSize);

                //printf("%d  ",ans[ii][0]);

                //printf("%d  ",ans[ii][1]);

                //printf("%d  ",ans[ii][2]);

                if( ans[ii][0] == nums[i] &&  ans[ii][1] == nums[k] &&  ans[ii][2] == nums[j]){

                    //printf("check1=%d\n",check);

                    check=1;

                    break;

                }

                //printf("\n");

            }

        }

        printf("%d, nums[i=%d]=%d + nums[j=%d]=%d + nums[k=%d]=%d\n",sum,i,nums[i],j,nums[j],k,nums[k]);

        printf("check2=%d\n",check);

        if( sum==0 && check != 1 ){

            //printf("000000c=%d\n",c);

            if(c>=1){

                int\*\* ans2=realloc(ans,sizeof(int\*)\*(c+1));

                ans=ans2;

            }

            \* returnColumnSizes=realloc(\*returnColumnSizes,sizeof(int)\*(c+1));

            \* returnSize=c+1;

            ans[c]=malloc(sizeof(int)\*3);

            ans[c][0]=nums[i];

            ans[c][1]=nums[k];

            ans[c][2]=nums[j];

            //printf("%d %d %d\n",ans[c][0],ans[c][1],ans[c][2]);

            (\* returnColumnSizes)[c]=3;

            c++;

        }

        if( k== i+1 && j==k+1){

            break;

        }else if( sum > 0){

            j--;

        }else if( sum < 0){

            k++;

        }else{

            j--;

            k++;

        }

        if( j <= k){

            do{

                i++;

                k=i+1;

                j=numsSize-1;

            }while( nums[i]==nums[i-1] && i > numsSize-2 );

        }

        check=0;

    }

    return ans;

}