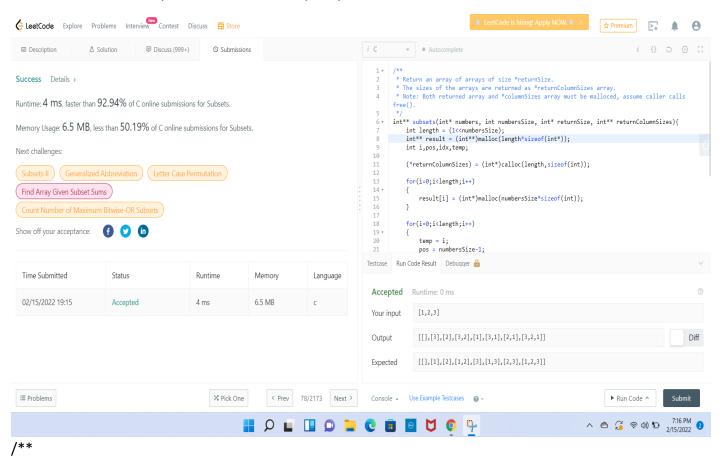
1. Solve the Leet code question no. 78 (Subsets)

Implement a solution that can be accepted.

Provide screen shot of your submission. Also post your source code here.



- * 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** subsets(int* numbers, int numbersSize, int* returnSize, int** returnColumnSizes){
  int length = (1<<numbersSize);
  int** result = (int**)malloc(length*sizeof(int*));</pre>
```

```
int i,pos,idx,temp;
(*returnColumnSizes) = (int*)calloc(length,sizeof(int));
for(i=0;i<length;i++)</pre>
{
  result[i] = (int*)malloc(numbersSize*sizeof(int));
}
for(i=0;i<length;i++)
{
  temp = i;
  pos = numbersSize-1;
  idx = 0;
  while(temp)
    if(temp&1)
      result[i][idx++] = numbers[pos];
    temp>>=1;
    pos--;
  }
  (*returnColumnSizes)[i] = idx;
}
*returnSize = length;
return result;
```

}

2. Solve the Leet code question no. 46 (Permutations)

Implement a solution that can be accepted.

Provide screen shot of your submission. Also post your source code here.

```
☆ Premium 🚉
                                                                                                           int factorial(int n){
   if(n==0)
      return 1;
   return n*factorial(n-1);
 Success Details >
 Runtime: 7\, ms, faster than 98.29\% of C online submissions for Permutations
                                                                                                           void add_curr_permutation(int** res, int *ret_size, int* nums, int n_size ){
  int i=0;
  for(i=0)icn_size;i*+){
    res[*ret_size][i]=nums[i];
}
 Memory Usage: 7.1~MB, less than 84.00\% of C online submissions for Permutations
 Next challenges:
                                                                                                               }
(*ret_size)++;
 Next Permutation Permutations II Permutation Sequence
                                                                                                          void swap(int *a, int *b){
   int temp=*a;
   *a=*b;
   *b=temp;
 Show off your acceptance:
                                                                                     Language
                                                                                                    Testcase Run Code Result Debugger
   02/15/2022 19:27
   02/15/2022 19:20
                            Compile Error
                                                                                                     Your input
                                                                                                                 [1,2,3]
                                                                                                                  [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,2,1],[3,1,2]]
                                                                                                     Output
                                                                                                                  [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]
                                                                                                     Expected
  ≡ Problems
                                                        ≯ Pick One
                                                                        < Prev 46/2173 Next >
                                                                                                                                                                        ▶ Run Code ^
                                                                                                                                                                  ^ △ ☐ ♠ ♠ ♠ № 7:27 PM 2/15/2022
int factorial(int n){
if(n==0)
return 1;
return n*factorial(n-1);
}
void add_curr_permutation(int** res, int *ret_size, int* nums, int n_size ){
int i=0;
for(i=0;i<n_size;i++){</pre>
res[*ret_size][i]=nums[i];
(*ret_size)++;
```

```
return;
}
void swap(int *a, int *b){
int temp=*a;
*a=*b;
*b=temp;
}
// The method which recursively generates the permutations
void permute_all(int *nums, int n_size, int start, int* ret_size, int** result){
if(start>=n_size){
add_curr_permutation(result, ret_size, nums, n_size );
return;
}
int i=0;
for(i=start;i<n_size;i++){</pre>
swap(&nums[start], &nums[i]);
permute_all(nums, n_size, start+1, ret_size, result);
swap(&nums[start], &nums[i]);
}
}
int** permute(int* nums, int numsSize, int* returnSize, int** returnColumnSizes){
if(numsSize==0)
return NULL;
```

```
int res_size=factorial(numsSize); //permuations will be factorial of numsSize
int i=0;
int **result=(int**)(calloc(sizeof(int*), res_size));
*returnColumnSizes=(int*)calloc(res_size,sizeof(int));

for(i=0;i<res_size;i++){
  result[i]=(int*)(calloc(sizeof(int), numsSize));
  (*returnColumnSizes)[i]=numsSize;
}

*returnSize=0;
permute_all(nums, numsSize, 0, returnSize, result);
return result;
}</pre>
```