15. 3 Sum:

Ilp: nums = [-1,0,1,2,-1,-4]. OP: [[-1, -1, 2], [-1, 0, 1])

Explanation:

nums [0] a nums [1] + nums [2] = (-1)+0 +1 =0 hums [1] + hums [2] + nums [4] = 0+1+(-1)=0 nums [0] + nums [3] + nums [4] = (-1)+2+(1)=0 The distinct triplets are [-1,0,1] and [-1,-1,2] Notice that the order of output and order of triplets does not matter.

Brute force;

Check every possible triplet using three loops:

[i] enum spick w

* pick nuns [3] after i

* pick nums (x) after;

* Check if their sum is o

& store triples (but make sure to avoid deplicates).

(1) some of spenier

-> This is simplest way but it is slow?

optimal approach (two pointers + sorting)

1. sort the array.

Eq: [-1,0,1,2,-1,-4] -> [-4,-1,-1,0,1,2] sorting helps:

use two pointer technique Avoid duplicate triplets easily.

```
one number -> then find 2 numbers using
pointers.
 for each? (0 to h-3):
    * skip displicate values of nems[i]
    * set!
left = i +1
         right = n-1
        som = orums[i]+ hums[left] + nums[right].
     s check
  If sum = = 0 store triplet, skip duplicates, nove
3. Bared on sum:
   If sun < 0 move left ++ (to get bigget sum)
both pointers.
   It sum > 0 move right -- (to get smaller sum).
import gava. util .*;
 class Solution {

public List & List & (Integer >> three Sum (int []

public List & List & (Integer >> three Sum (int []
         Arrays. sort (nums);
List 2 list 2 Integer ?? result = new Arraylist
2>();
     for (int i = 0; ic nums. length-2; i++) {
        26 (i) 082 nurs[i] = = nurs [i-1] continue,
       int left = ?+1;
int right = nums-length -1;
       evhile (left / right) {
        int sum = neums[i] + muns[left] + nuns
[right].
```

```
if ( sum = =0) {
    result add ( Arrays as list ( nums [ : ], nums [ left]
          num (zight]));
  evhile [left < right 22 mms [left] = = nevns[left +i])
  while (left a right 22 nums (right) ==

nums (right - J) right --;
     left ++;
                 tolgist mate o - nun
     right --;
 we that the one of the fire of the fire of
 else if (sun co)?
   ] else {
                    ( * . dite . over trogmi
 right --;
                egetate a till a tell sidery
return result;
                   : (insur) tron . special .
                 competits till a till
  nurs [[] Constan
              space > 0(1).
  71me -> 0 (n3)
             11 - Spred wares - type the
       British & Held alides
  incert Eget James + [James : mis the
· [ expir.]
```