题1、给定一个int数组，一个数sum，求数组中和为sum的任意2个数的组合

|  |
| --- |
| @Test  **public** **void** test\_find2() {  **int**[] arr = { -1, 0, 2, 3, 4, 7, 8, 9, 10 };  **int** sum = 9;  Arrays.*sort*(arr);  List<TwoTuple<Integer, Integer>> result = **new** ArrayList<>();  **int** i = 0;  **int** j = arr.length - 1;  **while** (i < j) {  **if** (arr[i] + arr[j] == sum) {  result.add(**new** TwoTuple<Integer, Integer>(arr[i], arr[j]));  i++;  j--;  } **else** **if** (arr[i] + arr[j] < sum) {  i++;  } **else** { // > sum  j--;  }  }  System.***out***.println(result);  } // out: [-1:10, 0:9, 2:7] |

题2、给定一个int数组，一个数sum，求数组中和为sum的任意3个数的组合

|  |
| --- |
| @Test  **public** **void** test\_find3() {  **int**[] arr = { -1, 0, 2, 3, 4, 7, 8, 9, 10 };  **int** sum = 9;  Arrays.*sort*(arr);  Set<ThreeTuple<Integer, Integer, Integer>> result = **new** LinkedHashSet<>();  **for** (**int** i = 0; i < arr.length; i++) {  **int** temp = arr[i];  swap(arr, i, 0);  **int** start = 1;  **int** end = arr.length - 1;  **while** (start < end) {  **if** (arr[start] + arr[end] == sum - temp) {  **int**[] threeNum = {temp, arr[start], arr[end]};  Arrays.*sort*(threeNum);  result.add(**new** ThreeTuple<>(threeNum[0], threeNum[1], threeNum[2]));  start++;  end--;  } **else** **if** (arr[start] + arr[end] > sum - temp) {  end--;  } **else** {  start++;  }  }  swap(arr, i, 0); // 还原  }  System.***out***.println(result);  } // out: [-1:0:10, -1:2:8, -1:3:7, 0:2:7, 2:3:4]  **private** **void** swap(**int**[] a, **int** i, **int** j) {  **int** temp = a[i];  a[i] = a[j];  a[j] = temp;  } |

上面两题用到的元组类：

|  |
| --- |
| **class** TwoTuple<A, B> {  **public** **final** A first;  **public** **final** B second;  **public** TwoTuple(A a, B b) {  **this**.first = a;  **this**.second = b;  }    @Override **public** String toString() {  **return** first + ":" + second;  }  @Override  **public** **int** hashCode() {  **final** **int** prime = 31;  **int** result = 1;  result = prime \* result + ((first == **null**) ? 0 : first.hashCode());  result = prime \* result + ((second == **null**) ? 0 : second.hashCode());  **return** result;  }  @Override  **public** **boolean** equals(Object obj) {  **if** (**this** == obj)  **return** **true**;  **if** (obj == **null**)  **return** **false**;  **if** (getClass() != obj.getClass())  **return** **false**;  TwoTuple<?, ?> other = (TwoTuple<?, ?>) obj;  **if** (first == **null**) {  **if** (other.first != **null**)  **return** **false**;  } **else** **if** (!first.equals(other.first))  **return** **false**;  **if** (second == **null**) {  **if** (other.second != **null**)  **return** **false**;  } **else** **if** (!second.equals(other.second))  **return** **false**;  **return** **true**;  }  } |
| **class** ThreeTuple<A, B, C> **extends** TwoTuple<A, B> {  **public** **final** C third;  **public** ThreeTuple(A a, B b, C c) {  **super**(a, b);  **this**.third = c;  }    @Override **public** String toString() {  **return** **super**.toString() + ":" + third;  }  @Override  **public** **int** hashCode() {  **final** **int** prime = 31;  **int** result = **super**.hashCode();  result = prime \* result + ((third == **null**) ? 0 : third.hashCode());  **return** result;  }  @Override  **public** **boolean** equals(Object obj) {  **if** (**this** == obj)  **return** **true**;  **if** (!**super**.equals(obj))  **return** **false**;  **if** (getClass() != obj.getClass())  **return** **false**;  ThreeTuple<?, ?, ?> other = (ThreeTuple<?, ?, ?>) obj;  **if** (third == **null**) {  **if** (other.third != **null**)  **return** **false**;  } **else** **if** (!third.equals(other.third))  **return** **false**;  **return** **true**;  }  } |

题3、给定一个int数组，一个数sum，求数组中和为sum的k个数的组合有多少种（k任意）。

另开一博客讨论这个问题。指路：【算法习题】数组中和为sum的任意个数的组合数