## Prob1

int[] arrays(int n) {  
 int[] arr = new int[n];  
 for(int i = 0; i < n; ++i){  
 arr[i] = 1;  
 }  
 for(int i = 0; i < n; ++i) {  
 for(int j = i; j < n; ++j){  
 arr[i] += arr[j] + i + j;  
 }  
 }  
 return arr;  
}

**Answer:**

The run time of

for(int i = 0; i < n; ++i){  
 arr[i] = 1;  
}

is O(n)

The run time of

for(int i = 0; i < n; ++i) {  
 for(int j = i; j < n; ++j){  
 arr[i] += arr[j] + i + j;  
 }  
}

is O(n2)

The total run time is O(n) + O(n2) = O(n2)

## Prob2

Algorithm merge(A, B)  
 Input: 2 sorted arrays need to merged A & B  
 Output: The sorted array contains all elements from A & B  
  
 C = new Array[A.length + B.length]  
 for i <- 0 to A.length - 1 do  
 C[i] = A[i]  
  
 for i <- 0 to B.length - 1 do  
 C[A.length + i] = B[i]  
  
 for i <- 0 to C.length - 1 do  
 for j <- i to C.length - 1 do  
 if C[i] > C[j] then  
 temp = C[i]  
 C[i] = C[j]  
 C[j] = temp  
  
 return C

1. The run time is O(n) + O(n) + O(n2) = O(n2)

## Prob5

for (Integer i: arr) {  
 if (!distinctList.contains(i)) {  
 distinctList.add(i);  
 }  
}

The **contains** method of List loop over the List and compare with each element to determine the List contain i or not so the run time is O(n2)