1. **TapeEquilibrium**:

Solution: https://codility.com/demo/results/training53ZNF6-HKR/

2. **PermMissingElem**:

Solution: https://codility.com/demo/results/trainingFQDT3Q-YVG/

3. FrogJmp:

Solution: https://codility.com/demo/results/trainingKFBVSX-MY9/

4. FrogRiverOne

Solution: https://codility.com/demo/results/training4SGAEP-M6Q/

5. PermCheck

Solution: https://codility.com/demo/results/trainingXNUJVQ-YB4/

6. MissingInteger

Solution: https://codility.com/demo/results/trainingJDS55W-XMM/

7. MaxCounters

Solution: https://codility.com/demo/results/trainingUUEYK4-87T/

特点就是，在过程中不进行统一更新，来节省速度，而是在遇到的时候，看看是否要更新到最新进度，然后再加一。

8. CountDiv

Solution: https://codility.com/demo/results/trainingQBDNK5-PSY/

9. PassingCars

Solution: https://codility.com/demo/results/training98K6RU-QC9/

10. **MinAvgTwoSlice**

https://codility.com/demo/results/trainingUUYB9H-QDV/

11. **GenomicRangeQuery**

http://codesays.com/2014/solution-to-genomic-range-query-by-codility/

利用 prefix sums 的概念，将求和的过程转换为改存一个 list 里面，这样当我们给定一个 range 的时候，我们可以利用这个 list，来轻易的将过程中的值取出。在这个题目中，目的是得到在给定的 range 中，存在多少个 A,G,C,T。因此，我们可以将 AGCT 的个数与 range 关系，分别存在四个 list 里面，然后用给定的 range 来判断。

https://codility.com/demo/results/trainingZRXXP2-HGG/

12. **MaxProductOfThree**

Solution: https://codility.com/demo/results/trainingCX3M7W-EGM/

对付负数，简单的判断一下即可

13. **Triangle**

Solution: https://codility.com/demo/results/trainingA9DT5P-KEB/

最容易达到的能够达到了才有意义，从最难达到的情况进行判断不可取

14. **Distinct**

Solution: https://codility.com/demo/results/training3MGYEH-7JU/

15. **NumberOfDiscIntersections**

Solution: https://codility.com/demo/results/trainingVG4B3Y-KP8/

将半径，圆心等条件转换成圆的开始，终止坐标，然后进行判断。当一个新的圆开始，则增加一，一个旧的圆结束，则减去一。

16. **Nesting**

Solution: https://codility.com/demo/results/training52XJXG-ARF/

17. **StoneWall**

Solution: https://codility.com/demo/results/training5FJFPV-NWQ/

18. **Brackets**

Solution: https://codility.com/demo/results/trainingXMZVDC-HTE/

19. **Fish**

Solution: https://codility.com/demo/results/trainingGRDQMV-CNJ/

20. **Dominator**

Solution: https://codility.com/demo/results/trainingU2YXJ6-GBC/

21. **EquiLeader**

https://codility.com/demo/results/trainingKRJVKB-VHS/

22. **MaxDoubleSliceSum**

https://codesolutiony.wordpress.com/2015/01/05/codility-7-1-max-double-slice-sum/

Solution: http://codesays.com/2014/solution-to-max-double-slice-sum-by-codility/

https://codility.com/demo/results/trainingE39WGY-KP8/

23. **MaxProfit**

先找最高的卖出收益，然后同时更新最高的卖价。

Solution: https://codility.com/demo/results/trainingXTWU9Q-2YU/

24. **MaxSliceSum**

Solution: https://codility.com/demo/results/trainingJ6SFRF-FCS/

http://liweithu.me/maximum_slice_problem/

左边界的关键在于，证明它比加上之前的和还大，因此它就能成为新的左边界。

25. **MinPerimeterRectangle**

Solution: https://codility.com/demo/results/trainingC82Z94-Q75/

26. **CountFactors**

Solution: https://codility.com/demo/results/trainingHKVXDZ-NQM/

27. **Peaks**

Solution: https://codility.com/demo/results/trainingGF7AEJ-7FD/

28. **Flags**

Solution: https://codility.com/demo/results/trainingK8YW4X-YFD/

29. **CountSemiprimes**

Solution: https://codility.com/demo/results/trainingU8QVCD-7TP/

30. **CountNonDivisible**

Solution: https://codility.com/demo/results/training6NCZ7P-3HY/

31. **ChocolatesByNumbers**

Solution: https://codility.com/demo/results/trainingA6XTVD-NXM/

32. **CommonPrimeDivisors**

Solution: https://codility.com/demo/results/trainingAHFCPK-YUB/

33. **Ladder**

Solution: https://codility.com/demo/results/trainingFSKDK9-WSR/

34. **FibFrog**

Solution: https://codility.com/demo/results/training98TSXQ-MPU/

35. **MinMaxDivision**

两个函数，子函数为求 在当前给定的可能的最大解的时候，看在原 list 中需要多少个元素来组成。如果返回的元素个数超过我们给定的分组的限制，则说明这个猜的中间解无法使用。我们要根据情况来移位。因此这种题型，解是猜，然后进行验证。看解是否能够满足。

Solution: https://codility.com/demo/results/training6H82R3-A6E/

36. NailingPlanks

可以使用 binary search 方法或者不用

Solution (Binary Search): https://codility.com/demo/results/trainingDPXVKB-ZGY/

Solution (Non-Binary Search): https://codility.com/demo/results/training75SUK9-57H/

37. AbsDistinct

如果不用 Caterpillar，有：

Solution: https://codility.com/demo/results/trainingYEBGGV-MQC/

如果使用 Caterpillar，则

Solution: https://codility.com/demo/results/trainingNYHXAY-BU7/

38. CountDistinctSlices

Solution: https://codility.com/demo/results/trainingS6CGPV-YBA/

39. CountTriangles

Solution: https://codility.com/demo/results/trainingEGTNGB-SKD/

40. MinAbsSumOfTwo

Solution: https://codility.com/demo/results/trainingXXMR7H-CT3/

41. MaxNonoverlappingSegments

Solution: https://codility.com/demo/results/trainingM6PWHM-EJS/

42. TieRopes

Solution: https://codility.com/demo/results/training2BKYF4-RYV/

43. NumberSolitaire

Solution: https://codility.com/demo/results/trainingZF685A-MXJ/

44. MinAbsSum

Solution: https://codility.com/demo/results/trainingZRD475-YNC/

https://codility.com/media/train/solution-min-abs-sum.pdf

45. StrSymmetryPoint

Solution: https://codility.com/demo/results/trainingTZHXJQ-EQ2/

46. TreeHeight

Solution: https://codility.com/demo/results/training3G8C3N-W4E/

47. BinaryGap

Solution: https://codility.com/demo/results/trainingXP2DHW-BXF/

48. OddOccurrencesInArray

Solution: https://codility.com/demo/results/trainingPM7456-EE3/

49. ArrayInversionCount

在排序的时候，直接进行比较，然后计算符合条件的 pair 的个数。

Solution: https://codility.com/demo/results/trainingSACUY2-6YC/

50. PolygonConcavityIndex

使用一个动态规划的方法，先找到 y 坐标最小的点，定为右下角，然后开始按照其余点跟它的关系来找到另外的点。

Solution: https://codility.com/demo/results/trainingJM3HZ2-WX3/