VIII открытый командный online турнир по программированию
Юго-Восточного образовательного округа
Центр цифрового образования «ИТ-куб» на базе КОГОАУ «Вятский многопрофильный лицей» г. Вятские Поляны
5 декабря 2024 года

Разбор задач

Странная парковка

Очевидно, для достижения максимального результата нужно разместиться через одно парковочное место.

Ответом будет половина числа, а для нечетного значения на 1 больше.

Возможная формула ans = n // 2 + n % 2.

Пассажиры

Количество эгоистов равно количеству роботакси для них.

Количество коллективистов делим на 3 и добавляем к промежуточному количеству роботакси. Далее, если оставшееся количество коллективистов (а это 1 или 2) больше, чем количество пофигистов, то выводим -1, ответ невозможен, коллективисты не могут ехать в неполном роботакси. Иначе, к оставшимся коллективистам добавляем пофигистов и находим количество робоавтомобилей для них (делим на 3). Если их количество не делится на 3 нацело, то не забываем добавить единицу.

Добавляем это количество к общему результату.

Побитовые операции

В задаче нужно для всех левых битов пар цепочки элементов найти минимальное количество среди нулей и единиц. Аналогично выполнить подсчет и для правых битов всех пар. Результат – сумма двух минимумов.

Стопки монет

Первое искомое значение (высота наибольшей стопки монет) – это максимальное значение одинаковых чисел в массиве. Можно воспользоваться сортировкой подсчетом.

Второе искомое значение (количество разных стопок монет) – это количество различных чисел в массиве. Для этого подойдет множество.

Кто больше

Имеются две строки (списки). Чтобы число после произведенных замен цифр получилось максимальным, необходимо отсортировать цифры второй считанной строки по убыванию.

Выгоднее большими цифрами заменять старшие цифры числа. Затем достаточно пройти по заменяемой строке (а это первая строка), сравнивая каждую ее цифру с цифрами упорядоченной второй строки. Если цифра первой строки окажется меньше цифры второй строки, то заменяем ее. Продолжаем работу, пока какая-то из строк не закончится.

Давайте поиграем

Смоделируем все операции, которые описаны в задаче.

Возьмем два счетчика, суммы первого игрока и его противника. Будем сравнивать числа, находящие справа и слева. Начальные индексы – первый и последний. В зависимости от того, какое из чисел больше, добавляем его к одной из сумм и изменяем индекс элемента, либо левый индекс увеличиваем, либо правый уменьшаем.

Суммы, которые будут увеличиваться, определяем по четности хода игрока.

Хочу выше

Для получения результата необходимо сделать копию вводимой строки, назовем ее stroka1 и отсортировать эту копию в лексикографическом порядке. Получилась stroka2. Лучше обе строки превратить в списки.

Затем начать сравнивать буквы исходной строки и отсортированной. Как только обнаружили, что stroka1[i] > stroka2[i], то символ, стоящий на месте stroka1[i], необходимо сохранить, а затем заменить stroka1[i] на stroka2[i].

После этого найти позицию сохраненного символа в stroka2 и поставить его на соответствующую позицию в stroka1. Символы в строке могут повторяться, поэтому нужно найти самое правое вхождение данного символа.

Выводим измененную строку. Если никаких событий не произошло, то строка не изменится.

Битва роботов

Достаточно просмотреть каждую строку таблицы. Если в ней есть 1 или 3, то роботы этой строки не могут выйти в финал. Подсчитаем количество таких строк. Если счетчик для текущей строки не увеличили, то ее номер сохраняем.

Результат – исходное количество строк минус количество найденных выше строк и список сохраненных строк.

Дефицитный товар с нагрузкой

Ответ “YES” будет только в случае, когда a >= n and b >= n. В остальных случаях ответ “NO”.

Экономный электрик

Для минимизации результата необходимо числа отсортировать. Воспользуемся одномерной динамикой.

Будем заполнять динамический массив, находя разность между соседними элементами исходного массива и добавляя их к минимальному из двух предыдущих элементов нового массива.

Таким образом мы будем либо продолжать предыдущую связку проводов, либо начинать новую.

Формула такова:

d[i] = min(d[i - 1], d[i - 2]) + abs(a[i] - a[i-1]).

Новогодний салют

Рассмотрим каждую установку отдельно. Первая установка выпускает салюты в a, a*2, a*3,… минут после начала, вторая установка соответственно в b, b*2, b*3,… минут после запуска.

Максимальное количество салютов можно увидеть с момента времени T = НОК(a, b). Очевидно, что T делится на a и на b.

Таким образом, за m минут можно увидеть одновременно m // a и m // b салютов. Добавим сюда два салюта, выпущенных в момент времени T.

Отсюда результат = m // 2 + m // b + 2.

Просто интересно

Задачу можно решать, перебирая разные значения a, b, c. Но, чтобы вписаться во время при большом n, необходимо организовать перебор двух значений, ограничивая возможные значения до корня из числа n, а третье вычислять по формуле c = n // a // b.

Кроме этого, нужно проверять, чтобы эти значения были делителями n.

Для нахождения нужных результатов, используем формулу x = (a + 1)*(b + 2)*(c + 2) - n.

Параллельно ищем среди них минимум.

Не факт, что a – это нижнее измерение. Поэтому, для нахождения максимальных значений формулу немножко изменим y = (a + 2)*(b + 2)*(c + 1) – n.

Новогодний конкурс

Сначала нужно написать функцию, моделирующую поедание ряда мешочков с конфетами. При этом каждый участник забирает столько мешочков с конфетами, сколько он может съесть за время t. Если не весь конфеты будут съедены, функция возвращает false, иначе true.

С помощью бинарного поиска находим наименьшее t, при котором все конфеты во всех мешочках будут съедены.