**Вологда, Импульс, смена по информатике, декабрь 2020**

**СНМ, минимальный остов**

**Задача A – Слияние множеств**

 Нужно просто реализовать DSU: <http://e-maxx.ru/algo/dsu>

**Задача B – Объединения с отменами**

Используем систему непересекающихся множеств (DSU): <http://e-maxx.ru/algo/dsu>.

Для реализации откатов добавим к этой структуре стек, в который при объединении будем помещать корень того дерева, который присоединяем к корню другого дерева.

 Поскольку нам нужно отменять и те запросы на объединение, которые фактически не выполнялись, то для таких "объединений" можно класть в стек, например, -1. Отмена такого запроса – просто вытолкнуть минус единицу из стека.

**Задача C – Нуль-транспортировка**

Нужно просто применить алгоритм Крускала: <http://e-maxx.ru/algo/mst_kruskal>

**Задача D – Электрическая задача**

Нужно просто применить алгоритм Прима: <http://e-maxx.ru/algo/mst_prim>

Алгоритм Краскала, вероятно, не пройдёт все тесты по времени, так как для плотных графов сложность *O(m∙log(m))* у Краскала хуже, чем *O(n2)* у Прима.

Для дополнительного повышения скорости можно написать код так, чтобы вычислять не более одного раза расстояние между одной и той же парой вершин (так как функция sqrt довольно "тяжёлая").

**Задача E – Глеб и Макс в лесу**

 Задача со сборов юниоров в Иннополисе.

 Основная идея – использование СНМ с откатами. Пусть вначале каждое дерево – это отдельное множество из одного элемента. На первом шаге с помощью обходов в глубину выделим связанные области из деревьев, имеющих одинаковую высоту и одинаковую скорость роста, и выполним объединение вершин в каждой такой группе.

 Теперь рассмотрим все пары стоящих рядом деревьев из разных групп. Для каждой такой пары вычислим время, когда деревья этих групп станут одинаковой высоты. Для каждого полученного времени запомним все пары соседних деревьев, которые в это время станут одинаковой высоты.

 Будем перебирать эти времена. Взяв очередное время, объединим все группы деревьев, у которых стала одинаковая высота. Запомним размер самой большой области. Затем выполним откат всех сделанных объединений, и перейдём к следующему времени.

 Оценим сложность алгоритма. Поскольку каждая пара соседних деревьев может использоваться для объединения не более одного раза, то получается оценка O(n2\*log(n)).