1 Обязательная часть

1. ***Max отрезок без циклов.***

   Сначала сортим ребра. Это $m \log n$. Затем фигачим два указателя. Что с ними делать? Строим дерево эйлерового обхода. Сначала оно пустное. Когда двигаем R указатель вправо - добавляем в него ребро за логарифм. Двигаем L вправо - удаляем ребро.

   Как узнавать за логарифм, есть ли цикл? Если у добавленного ребра оба конца уже находятся в одной компоненте (узнаем это функцией findRoot, например), то между ними уже есть путь, мы делаем цикл. В этом случае двигаем указатель L вправо до тех пор, пока цикл не пропадет. Цикл не появился - релаксируем ответ получившейся длиной отрезка и двигаем указатель R вправо.

2. ***Статически оптимальный HLD.***

   Как строим пути. Пусть сейчас мы стоим в какой-то вершине и надо решить, куда вести наш путь дальше. Рассмотрим поддеревья этой вершины и запросы на них. Если какой-то запрос лежит целиком в одном поддереве, то он нас пока что не интересует, на данном этапе скачков между пути у них не будет.

   Посмотрим на запросы, которые лежат в поддереве лишь одним концом, а второй где-то в другом месте. Вот для таких запросов и будут совершены прыжки по путям. Раз мы их хотим минимизировать, то надо просто вести путь в то поддерево, в котором больше всего таких "неполных"запросов.

3. ***Доказать Expose в Link-Cut.***

   Заметим, что время expose(v) - суммарное количество легких и тяжелых ребер, соединяющих пути, лежащие по дороге от v к root (короче те ребра, по которым мы и прыгаем между путями). Скажем, что $t = heavy + light$.

   Теперь надо понять, что с изменением потенциала. В сумме покрыто стало столько же ребер, сколько перестало быть покрыто (одно ребро добавляем - одно убираем). На сколько увеличился потенциал? На количество новопокрытых тяжелых ребер. На сколько уменьшился? На количество тяжелых ребер, которые престали быть покрытыми. Заметим, что таких ребер не больше, чем легких (ведь в суме ребер поровну, и не могло оказаться так, что тяжелое ребро стало покрытым, а в то же время перестало быть покрытым тоже тяжелое... из одной врешины выходит не больше одного тяжелого). Итог: $\Delta\varphi \geq heavy - leght$.

   А теперь посмотрим на амортизированное время:

$$A = t - \Delta\varphi$$
$$A \geq heavy + light - (heavy - light) = 2 \cdot light = \mathcal{O}(\log n)$$

   Почему легких ребер $\mathcal{O}(\log n)$ мы вроде уже доказывали - во всем виноваты размеры поддеревьев.

4. ***MST за линию.***

Какая у нас есть рекурента:

$$T(n,m) = (n+m) + T(\tfrac{n}{8}, \tfrac{m}{2}) + T(\tfrac{n}{8}, |A_1| + |A_2|)$$

Второе слагаемое понятно, с ним ничего не сделать. Давайте посмотрим на слагаемое $|A_1| + |A_2|$. Заметим, что для него есть плохой случай — когда он равен $\frac{m}{2}$. Это случай, когда рандом сработал очень плохо, при переборе фором мы смогли всеми нашими $\frac{m}{2}$ ребер улучший остов. Получили $T(n,m) = (n+m) + 2T(\frac{n}{8}, \frac{m}{2})$. Заметим, что это $\mathcal{O}(m \log n)$, так как на каждом уровне рекурсии кол-во ребер уменьшется в 8 раз, и по итогам на каждом уровне будут рассмотрены все ребра.

Давайте посмотрим на $(n+m)$. Худший случай для m — полный граф — n^2. Получили $T(n,m) = (n+n^2) + 2T(\frac{n}{8}, \frac{n^2}{64})$. Можно свести теперь это все дело к такой записи: $T(n) = n^2 + 2T(\frac{n}{8})$, а это $\mathcal{O}(n^2)$.

Ну по итогам и получаем оценку на $\mathcal{O}(min(n^2, m \log n))$.