Задача А. Тема: структуры данных. Структура: массив векторов.  
Алгоритм: на вводе данных при вводе команды вида 1 добавляем в каждый отсек с L по R значение С. При вводе команды вида 2 удаляем в отсеках с L по R последний элемент. Затем проходим по массиву и, если вектор не пуст, выводим значение последнего элемента в нем. Если пуст, выводим 0.  
Проблема: не смог придумать более оптимальное решение, чем алгоритм, описанный выше

Задача В. Тема: обход массива при помощи 2 указателей /DFS. Структура: строка char, set < pair < char, int > >.  
Алгоритм: Вначале оба указателя равны 0 (первый символ строки ). Идем по строке, добавляя при встрече нового символа в set пары <новый символ, 0>, а при повторной встрече увеличивая второе значение пары на 1, пока st.size()<k(при этом второй указатель увеличивается на 1). После этого повторно проходим по строке, уменьшая второе значение пар с встречающимися символами на 1, пока, st.size()==k(при этом первый указатель увеличивается на 1). Затем, если ans>(разность второго и первого указателя), то присваиваем значению ans разность второго и первого указателя. Повторяем алгоритм уже с новыми начальными значениями указателей и заполненным set-ом.  
Проблема: зацикленность на том, что задачи «очень сложные» и помешанность на решении в лоб при помощи циклов.

Задача С. Тема: динамика. Структура: массив.  
Алгоритм:   
Проблема:

Задача D. Тема: арифметическая прогрессия, нахождение всех множителей числа . Структура: int, массивы.  
Алгоритм: для начала надо сузить границы поиска. Если R>A=C, то R=A-C; если L<C, то L=C.   
Проблема:

Задача E. Тема: динамика. Структура: массив.  
Алгоритм:   
Проблема:

Задача F. Тема: динамика. Структура: массив.  
Алгоритм:   
Проблема: