Разбор задач 1-го тура окружного (муниципального) этапа
Всероссийской предметной олимпиады школьников по информатике

Задача A. Отбросить нельзя округлить.

Обозначим как SD сумму в тысячах литров, которую подсчитывает директор, и SM – сумму в литрах, которую подсчитывают менеджеры. По условию задачи каждый день должно выполняться соотношение SD = <математическое округление>(SM / 1000). Именно на основании этого соотношения и принимается решение, как округлять объем продаж в очередной день.

Покажем, как это происходит, на примере.
Первый пробный тест был разобран в условии задачи. Разберем второй тест:
Первый день: продано 3300 л. Округление 3.3 дает 3; менеджеры сообщают директору о 3 тысячах литров. SD = 3, SM = 3300. В строку записывается F

Второй день: продано 3285 л. SM = 6585, округление 6.585 дает 7, и, чтобы у директора получилось столько, ему сообщают о 4 тысячах литров (т.е. округляют 3.285 “вверх”, в строку пишется C).

Третий день: продано 4000 л. SM = 10585, при округлении 10.585 дает 11, т.е. директору нужно сообщить о 4 тысячах литров (SD = 11 = 7 + 4). Поскольку число изменять не нужно, в строчку записывается F.

Четвертый день: продано 3175 л. SM = 13760 л. Округление 13.760 дает 14, и директору нужно сообщить о 3 тысячах литров, чтобы SD стало равно 14. Для числа 3.175 это округление “вниз”, поэтому в строку записывается F.

Пятый день: продано 3775 л. SM = 17535 л. Округление 17.535 дает 18. У директора SD = 14. Поэтому число 3.775 нужно округлить до 4. В строку записывается C.

Таким образом, получается ответ: FCFFC

Задача B. В четырех соснах.

Существует 4 принципиально различных случая:
(1) 4 точки лежат на одной прямой. Ответ 0.
(2) Ровно 3 точки лежат на одной прямой. Ответ 2.
(3) Никакие 3 точки не лежат на одной прямой, и существует треугольник, внутри которого расположена четвертая точка. В этом случае ответ 3.
(4) Никакие 3 точки не лежат на одной прямой, и 4 точки образуют выпуклый 4х-угольник. В этом случае ответ 2.
Научимся строить множество треугольников. Случай (1) очевиден – треугольников нет.
В случае (3) необходимо вывести три треугольника. Понятно, что они будут образованы точкой внутри треугольника и его ребрами. Так, если в треугольнике, образованном вершинами 1, 2 и 3 находится вершина 4, то ответом, например, будет
4 1 2
4 1 3
4 2 3
В случаях (2) и (4) надо вывести два треугольника. При этом в случае (2) ответ будет однозначен, а в случае (4) – нет (достаточно вывести любой возможный ответ).

Можно предложить несколько вариантов поиска искомых треугольников; самый простой (по мнению жюри) – выбрать две точки таким образом, чтобы оставшиеся две точки оказались по разные стороны от соединяющей выбранные точки прямой. Тогда две выбранные точки будут общим ребром двух треугольников, которые и надо вывести.

Задача C. Самый точный прогноз погоды.

В этой задаче требовалось аккуратно воспроизвести ежедневный процесс оценки прогнозов погоды различных служб. Поскольку все значения температур содержат ровно два знака после запятой, то, умножив их на 100, получим целые числа.
Также полезно сразу найти наибольший общий делитель (НОД) чисел dn и dd, и “сократить” дробь.
Точность прогноза погоды службы #j определяется с помощью формулы |Tj -R| * dd <= dn * 100 (где dd и dn уже после деления на НОД(dn, dd)).
Важно было обратить внимание на ограничения и воспользоваться типом, позволяющим хранить 64-разрядные целые числа для хранения результатов (как промежуточных, так и окончательных).
После того, как коэффициенты доверия к последнему дню будут вычислены, останется посчитать разницу между прогнозом Маши и реальной температурой. Поскольку ответ нужно вывести в формате “знак” “числитель” “знаменатель”, выделим их из формулы (k1 * T1 + k2 * T2 + ... + kN * TN)/(k1 + k2 + ... + kN) – R
Числитель примет вид k1 * T1 + k2 * T2 + ... + kN * TN – R*(k1 + k2 + ... + kN), а знаменатель – (k1 + k2 + ... + kN) * 100 (умножение знаменателя на 100 компенсирует умножение всех значений температур на 100). Теперь остается определить знак числа, найти наибольший общий делитель числителя и знаменателя и, выполнив деление, вывести ответ.

Задача D. Прогулка.

Это достаточно сложная графовая задача (самая сложная в этом туре).
У нас имеется два графа с общим множеством вершин. Необходимо выбрать два кратчайших пути от вершины 1 до вершины N, таких, чтобы в них было как можно больше общих ребер, которые проходятся "одномоментно" (заметим, что “одномоментно” – это ключевое слово).
Сначала для каждого графа выполним следующие действия:
1. Найдем кратчайшие пути от вершины 1 и до вершины N (например, с помощью алгоритма Флойда). Пусть d[i,j] - кратчайшее расстояние от i до j.
2. Оставим в графе только ребра (i,j), такие, что d[1,i] + 1 (= d[i,j]) + d[i,j] + d[j,N] = d[1,N]. Заметим, что в получившемся (ориентированном!) графе любой путь является кратчайшим.
3. Построим матрицу достижимости (например, алгоритмом Флойда). Отсчет path[i,j] в ней означает, что из вершины i можно добраться до вершины j.

Теперь построим новый граф. Ребро (i,j) будет:
а) присутствовать в новом графе с весом 1, если ребро (i,j) есть в обоих графах (после выполнения вышеуказанной процедуры), причем расстояние от вершины 1 до i одинаково в обоих графах. По этому ребру можно проехать "вместе".
б) (иначе) присутствовать в новом графе с весом 0, если из вершины i можно добраться до вершины j в обоих графах (это проверяется по матрице достижимости).
в) (иначе) отсутствовать в графе.

Заметим, что любые два кратчайших пути в изначальных графах могут быть представлены как путь в новом, причем длина пути в новом графе будет равна времени, которое Маша и Витя едут вместе. Это легко понять, если представить, что у Маши и Вити есть третий друг, который летит на вертолете, если Маша и Витя едут порознь, и едет с ними, если они вместе.

Таким образом, необходимо найти самый длинный путь в новом графе. Заметим, что этот граф ациклический и ориентированный (DAG). Наидлиннейший путь в таком графе строится топологической сортировкой графа и последующим вычислением ответа методом динамического программирования по графу (вершины обходятся в топологическом порядке, в момент обхода известны длины путей от всех вершин, в которые из данной есть ребра).

Осталось вывести ответ. По пути "третьего друга не вертолете" несложно восстановить маршрут каждого из катающихся — действительно, достаточно вывести кратчайший путь из i в j для каждого ребра (i,j), входящего в найденный путь, по графам Маши и Вити соответственно.

Задача E. Играем калькулятор.

В этой задаче необходимо реализовать стековый калькулятор с "игнорированием" ошибок. Поскольку сложность задачи исключительно техническая, приводим описание решения жюри.
Описывается структура данных "стек" со следующими операциями:
push – добавить число
pop – извлечь из стека последнее число. Если стек пуст – вернуть NaN.
top – сообщить о том, какое число было добавлено последним. Если стек пуст - вернуть NaN
Значение NaN хранится как –1 (поскольку гарантируется, что все числа неотрицательные) .
После этого легко реализуются все действия, указанные в задаче.
Например, div:
а) извлечь число из стека (pop). Обозначим его a.
б) извлечь число из стека (pop). Обозначим его b.
в) если a = NaN или b = NaN или же a = 0, то положить в стек NaN, иначе положить в стек число b div a
В любом случае необходимо правильно обрабатывать все возникающие ситуации. Например, условие задачи допускает следующий тест:

```
4
pop
print
add
print
```

Заметим, что команда pop вызывается при пустом стеке, как и add, и неправильная обработка вызывает проблемы, например, выход за границу массива (и неправильный ответ либо аварийное завершение программы).
Домашнее задание: попробуйте реализовать обычный калькулятор выражений вида 1+2*3+5, используя стековый калькулятор и дополнительный стек для команд. Подсказка: прочитайте про польскую запись.