

Муниципальный этап
Всероссийской олимпиады школьников
по информатике

в 2014 – 2015 учебном году

Разборы решений и идеи тестов

**Муниципальный этап Всероссийской олимпиады
школьников по информатике
в 2014 – 2015 учебном году
9 класс**

Время выполнения задач — 4 часа

Ограничение по времени — 2 секунды на тест

Ограничение по памяти — 64 мегабайта

9.1. «Собираем урожай». В огороде на даче у Пети Торопыжкина выросло n кочанов капусты, которые нужно перетаскать в дом. Петя за одну ходку относит не более k_1 кочанов, что занимает у него t_1 минут. Его брат может унести не более k_2 кочанов, причём у него ходка занимает t_2 минут. Уборкой капусты займётся один из братьев (другой будет убирать морковь). За какое наименьшее время можно убрать всю капусту?

Формат входа: В единственной строке через пробел заданы пять целых чисел n, k_1, t_1, k_2, t_2 , каждое из диапазона от 1 до 10^4 .

Формат выхода: Выведите единственное целое число — количество минут, которое займёт наискорейшая уборка капусты одним из братьев.

Пример

Вход: Выход:

15 4 3 5 10 12

9.2. «Нужная сумма». Иногда Пете Торопыжкину приходится решать задачи, которые имеют формализованную постановку. Вот одна из них: напишите программу, которая из заданного набора натуральных чисел выделяет два элемента, кратных 17, так, чтобы их сумма была кратна 3.

Формат входа: В первой строке задано единственное целое число n — размер набора ($2 \leq n \leq 10^5$). В следующей строке через пробел заданы элементы этого набора — целые числа из диапазона от 1 до 10^5 . Числа в наборе могут повторяться.

Формат выхода: Если искомую пару чисел выделить можно, выведите эти числа через пробел в любом порядке. Если ответов несколько, выведите любой из них. Если выделить пару невозможно, выведите единственное число 0.

Пример 1

Вход:

4
17 51 102 1

Выход:

102 51

Пример 2

Вход:

4
17 2 85 17

Выход:

17 85

Пример 3

Вход:

4
17 51 5 17

Выход:

0

9.3. «Обеспечение продовольствием». Во время, свободное от учёбы и решения задач по информатике, Петя Торопыжкин иногда играет. В том числе и в

стратегии. Однажды он решил аккуратно проанализировать свой любимый экономический симулятор. В нём имеется n складов и m магазинов, в которых можно продавать продовольствие, хранящееся на складах. На i -м складе имеется запас в s_i единиц продовольствия. В j -м магазине потребность в продовольствии составляет d_j единиц, которые могут быть проданы по p_j денежных единиц каждая. Какую максимальную прибыль может получить Петя в этой игре? В один магазин продовольствие может доставляться с любого количества складов и с одного склада в любое количество магазинов.

Формат входа: В первой строке задано единственное целое число n — количество складов ($0 \leq n \leq 10^5$). В следующей строке через пробел перечислены запасы продовольствия s_i на складах ($0 \leq s_i \leq 10^4$). Общий запас продовольствия не превышает 10^6 единиц. В третьей строке задано целое число m — количество магазинов ($0 \leq m \leq 10^5$). В четвёртой строке через пробел перечислены цены p_j на продовольствие в магазинах ($0 \leq p_j \leq 10^3$). В пятой строке перечислены потребности магазинов в продовольствии d_j ($0 \leq d_j \leq 10^4$).

Формат выхода: Выведите единственное целое число — максимальную прибыль от продажи продовольствия, которую может получить Петя в заданных условиях.

Пример 1

Вход:

3
5 100 10
4
20 30 40 10
30 10 20 10

Выход:

1800

Пример 2

Вход:

3
5 50 10
4
20 30 40 10
30 10 20 10

Выход:

1750

9.4. «Переносы в расписании». Расписание дел Пети Торопыжкина очень плотное. И все дела важные. Но тут внезапно возникло ОЧЕНЬ ВАЖНОЕ дело, которое (возможно) пересекается с частью уже запланированных мероприятий. По имеющемуся петиному расписанию на день и срокам начала и конца очень важного дела определите, с каким количеством дел пересекается по времени очень важное дело? (Если очень важное дело примыкает спереди или с конца к другому делу, считается, что они не пересекаются.)

Формат входа: В первой строке через пробел заданы моменты начала и окончания очень важного дела. Во второй строке записано единственное неотрицательное целое число n — количество дел в петином расписании ($n \leq 1440$). В следующих n строках через пробел заданы сроки начала и окончания очередного дела. Дела заданы, вообще говоря, в произвольном порядке. Никакие два дела из расписания по времени не пересекаются между собой. Формат задания момента времени — hh:mm ($0 \leq \text{hh} \leq 23$, $0 \leq \text{mm} \leq 59$). Если час или минуты меньше десяти, то они могут снабжаться ведущим нулём, то есть допустимы записи 9:00, 09:00, 9:0 и 09:0. Петя считается занятым с начала минуты момента начала и

до конца минуты момента конца. Гарантируется, что время окончания каждого дела не раньше времени его начала.

Формат выхода: Выведите единственное целое число — количество дел, с которыми пересекается очень важное дело.

Пример

Вход: Выход:

10:00 11:59 2

4

12:00 12:29

10:20 10:39

8:30 09:59

11:30 11:59

9.5. «Делимость на 11». На парте в кабинете математики Петя обнаружил несколько длинных неотрицательных целых чисел. Он решил исследовать каждое число на делимость на 11: делится ли оно на 11 и, если нет, можно ли переставить в нём одну пару соседних цифр так, чтобы оно стало кратным 11. Помогите ему, напишите соответствующую программу. Напомним, что ведущие нули в записи числа не допускаются.

Формат входа: В первой строке задано целое число n — количество чисел, обнаруженных Петей ($1 \leq n \leq 1000$). В следующих n строках по одному на строке даны десятичные записи целых чисел, больших нуля; десятичная запись состоит из не менее чем из 1 и не более чем из 255 цифр.

Формат выхода: На выходе должно быть столько же строк, сколько было чисел на входе. Если очередное число делится на 11, в соответствующей строке выведите 0. Если не делится, но можно переставить две соседних цифры так, чтобы оно стало кратным 11, выведите позицию первой цифры из пары, которую нужно поменять местами. Если число не делится на 11 и его нельзя сделать кратным 11 перестановкой пары соседних цифр, выведите -1. Если допустимых перестановок несколько, выведите минимальную позицию.

Отсчёт позиций в числе идёт от старшего разряда и начинается с 1.

Пример

Вход: Выход:

3 0

396 2

369 -1

397

**Муниципальный этап Всероссийской олимпиады
школьников по информатике
в 2014 – 2015 учебном году
9 класс. Разбор решений и идеи тестов**

Далее в разборах используются обозначения типов: short int — 2-байтовый целый тип (integer в TurboPascal/BorlandPascal и int в BorlandC++), int — 4-байтовый целый тип (longint в TurboPascal/BorlandPascal/FreePascal, long int в BorlandC++, integer в Delphi/PascalABC и int в VisualC++/C++ Builder), int64 — 8-байтный целый (отсутствует в BorlandPascal и BorlandC++, но присутствует в системах программирования под Windows).

Задачи оцениваются по 100 баллов за каждую; стало быть, полная стоимость пакета — 500 баллов. В рамках одной задачи все тесты считать равноценными.

9.1. «Собираем урожай». *В огороде на даче у Пети Торопыжкина выросло n кочанов капусты, которые нужно перетаскать в дом. Петя за одну ходку относит не более k_1 кочанов, что занимает у него t_1 минут. Его брат может унести не более k_2 кочанов, причём у него ходка занимает t_2 минут. Уборкой капусты займётся один из братьев (другой будет убирать морковь). За какое наименьшее время можно убрать всю капусту?*

Программный комитет оценивает эту задачу как весьма простую. Вычисляем количество ходок, которые надо сделать Пете, количество ходок, которые нужно сделать брату, затем умножаем их на соответствующие времена и выбираем меньшее из результатов умножения. Некоторая тонкость состоит в вычислении количества ходок; если выражать математически, то это будут величины $\lceil n/k_1 \rceil$ и $\lceil n/k_2 \rceil$, где $\lceil \cdot \rceil$ — операция округления вверх. В большинстве языков эта операция реализована, однако лучше не прибегать к вещественным вычислениям и написать код примерно такой:

```
res := n div k1;  
if n mod k1 <> 0 then res := res+1;
```

Идеи тестов:

1. Минимальный тест $n = k_1 = k_2 = 1, t_1 = t_2 = 1$;
2. Минимальный тест $n = 10, k_1 = k_2 = 1, t_1 = t_2 = 1$;
3. Минимальный тест $n = k_1 = 1, k_2 = 10, t_1 = t_2 = 1$;
4. Минимальный тест $n = 1, k_1 = 10, k_2 = 1, t_1 = t_2 = 1$;
5. Максимальный тест $n = k_1 = k_2 = 10^4, t_1 = t_2 = 10$;
6. Максимальный тест $n = 10, k_1 = k_2 = 10^4, t_1 = t_2 = 10$;
7. Максимальный тест $n = k_1 = 10^4, k_2 = 10^4, t_1 = t_2 = 10$;
8. Максимальный тест $n = 10^4, k_1 = 10, k_2 = 10^4, t_1 = t_2 = 10$;
9. Максимальный тест $n = k_1 = k_2 = 10^4, t_1 = t_2 = 10$;

10. Максимальный тест $n = 10$, $k_1 = k_2 = 10^4$, $t_1 = t_2 = 10^4$;
11. Максимальный тест $n = k_1 = 10^4$, $k_2 = 10$, $t_1 = t_2 = 10^4$;
12. Максимальный тест $n = 10^4$, $k_1 = 10$, $k_2 = 10^4$, $t_1 = t_2 = 10^4$;
- 13–20. Случайные тесты.

9.2. «Нужная сумма». Иногда Пете Торпыжскину приходится решать задачи, которые имеют формализованную постановку. Вот одна из них: напишите программу, которая из заданного набора натуральных чисел выделяет два элемента, кратных 17, так, чтобы их сумма была кратна 3.

Сложность этой задачи программному комитету представляется ниже среднего, поскольку её решение достаточно технично.

Вначале читаем числа, игнорируя не кратные 17. Далее можно применить «наивный» алгоритм, перебирающий все возможные пары чисел и проверяющий их сумму на кратность 3. Однако если чисел, кратных 17, достаточно много (больше 14000–15000), то «наивный» алгоритм не уложится в ограничения по времени. Более разумный алгоритм использует следующую идею: чтобы сумма двух чисел была кратна 3, либо они оба должны быть кратны 3, либо одно из них даёт остаток 1 при делении на 3, а другое — остаток 2.

Реализовать эту идею можно следующим образом: заведём массив из трёх элементов с индексами 0, 1 и 2. Вначале заполним все ячейки признаком отсутствия чисел, дающих соответствующий остаток при делении на 3, например, значениями -1. При появлении очередного числа, кратного 17, проверяем его остаток при делении на 3. Если он равен 0, то при отсутствии ранее чисел, кратных 3, записываем его в 0-ю ячейку массива и продолжаем поиск. Если же число, кратное 3, уже встречалось, то заканчиваем поиск, выводим то число и последнее прочитанное.

Если остаток при делении на 3 нового числа равен 1 (соответственно, 2), то проверяем, было ли ранее найдено число с остатком 2 (соответственно, 1). Если было найдено, то прекращаем поиск и выводим эту пару чисел. Если же не было найдено, то запоминаем найденное число в соответствующей ячейке массива и продолжаем поиск.

Если мы прочитали весь набор чисел до конца и не прервали поиск, то подходящей пары чисел нет, о чём надо сообщить.

Идеи тестов:

1. минимальный тест, $n = 2$, оба числа не кратны 17;
2. минимальный тест, $n = 2$, только одно число кратно 17;
3. минимальный тест, $n = 2$, оба числа кратны 17, одно кратно 3, другое — нет;
4. минимальный тест, $n = 2$, оба числа кратны 17, оба кратны 3;
5. минимальный тест, $n = 2$, оба числа кратны 17, оба имеют остаток 1 при делении на 3;

6. минимальный тест, $n = 2$, оба числа кратны 17, оба имеют остаток 2 при делении на 3;
7. минимальный тест, $n = 2$, оба числа кратны 17, они имеют разные остатки при делении на 3;
- 8–14. то же, что в тестах 1–7, только $n = 20$ и все числа, кроме описываемых, не кратны 17;
15. $n = 1000$, много чисел кратных 17, при этом все они имеют остаток от деления на 3, равный 0;
16. $n = 1000$, много чисел кратных 17, при этом все они имеют остаток от деления на 3, равный 1;
17. $n = 1000$, много чисел кратных 17, при этом все они имеют остаток от деления на 3, равный 2;
18. $n = 1000$, много чисел кратных 17, при этом они имеют остатки от деления на 3, равные 1 и 2;
- 19–21. случайные тесты такие, что чисел, кратных 17, не слишком много (работает «наивный» алгоритм);
- 22–25. случайные тесты такие, что чисел, кратных 17, много («наивный» алгоритм не работает).

9.3. «Обеспечение продовольствием». *Во время, свободное от учёбы и решения задач по информатике, Петя Торопыжский иногда играет. В том числе и в стратегии. Однажды он решил аккуратно проанализировать свой любимый экономический симулятор. В нём имеется n складов и m магазинов, в которых можно продавать продовольствие, хранящееся на складах. На i -м складе имеется запас в s_i единиц продовольствия. В j -м магазине потребность в продовольствии составляет d_j единиц, которые могут быть проданы по p_j денежных единиц каждая. Какую максимальную прибыль может получить Петя в этой игре? В один магазин продовольствие может доставляться с любого количества складов и с одного склада в любое количество магазинов.*

Программный комитет считает эту задачу средней по сложности, поскольку идейно она несложна, но для своего решения требует реализации сортировки.

В целом, достаточно очевидно, что данную задачу решает «жадный» алгоритм: сначала продукты надо везти в магазин с самыми высокими ценами. Поэтому алгоритм может выглядеть следующим образом: упорядочиваем магазины по убыванию цены, обнуляем накопленное значение выручки, затем начинаем параллельно обрабатывать массивы магазинов (отсортированный) и складов (неотсортированный). Если оставшаяся потребность очередного магазина больше или равна оставшегося предложения очередного склада, то всё продовольствие с этого склада везём в этот магазин: уменьшаем потребность магазина, увеличиваем выручку на произведение предложения склада на цену, переходим к следующему складу. Если оставшаяся потребность очередного магазина меньше оставшегося

предложения очередного склада, то со склада в этот магазин везём объём продовольствия, равный оставшейся потребности, уменьшаем предложение склада, увеличиваем выручку на произведение потребности магазина и цены, переходим к следующему магазину. Останавливаем цикл, когда исчерпан либо массив складов, либо массив магазинов. Накопленное значение выручки является ответом.

Некоторая тонкость заключается в том, что при указанных ограничениях может получиться так, что «наивные» алгоритмы сортировки (пузырьковая сортировка, сортировка вставкой и т.п.) могут не укладываться в ограничения по времени. Для получения полного балла необходимо применять «умные» алгоритмы, например, быструю сортировку (сортировку Хоара), или же использовать библиотечные функции сортировки (в которых, собственно, как раз реализована быстрая сортировка). Те, кто будут использовать неоптимальные алгоритмы, получают около 65% полного балла.

Ещё один момент, потенциально требующий внимания, — ответ может не входить в тип `short int`, хотя обязательно войдёт в тип `int`.

Идеи тестов:

1. минимальный тест: $n = m = 0$;
2. минимальный тест: $n = 0$;
3. минимальный тест: $m = 0$;
4. минимальный тест: $n \neq 0$, но все предложения равны нулю;
5. минимальный тест: $m \neq 0$, но все потребности равны нулю;
6. минимальный тест: все цены равны нулю;
7. максимальный тест: $n = 10^5$, все $s_i = 10$, m не слишком большое;
8. максимальный тест: $n = 10^5$, сто значений $s_i = 10^4$, остальные $s_i = 0$, m не слишком большое;
9. максимальный тест: $m = 10^5$, n не слишком большое;
10. максимальный тест: $n = 10^5$, все $s_i = 10$, $m = 10^5$;
11. максимальный тест: $n = 10^5$, сто значений $s_i = 10^4$, остальные $s_i = 0$, $m = 10^5$;
- 12–17. случайные тесты с небольшим m ;
- 18–20. случайные тесты с большим m .

9.4. «Переносы в расписании». *Расписание дел Пети Торопыжкина очень плотное. И все дела важные. Но тут внезапно возникло ОЧЕНЬ ВАЖНОЕ дело, которое (возможно) пересекается с частью уже запланированных мероприятий. По имеющемуся петинскому расписанию на день и срокам начала и конца очень важного дела определите, с каким количеством дел пересекается по времени очень важное дело? (Если очень важное дело примыкает спереди или с конца к другому делу, считается, что они не пересекаются.)*

Программный комитет считает, что сложность этой задачи выше средней, поскольку, несмотря на относительную идейную простоту, для её решения привле-

кается не вполне тривиальный для 9-классников алгоритм пересечения отрезков на прямой.

Для начала следует отметить, что нужно организовать аккуратный ввод данных. Для тех, кто работает на языках C/C++, Java или Python, тут особой проблемы не будет — встроенные средства достаточно мощные, чтобы разобрать числа из указанного представления или чтобы разобрать входную строку на отдельные подстроки, представляющие числа. Однако участникам, использующим BASIC или Паскаль, тут надо будет немного потрудиться.

Идея решения оказывается достаточно простой, если сформулировать задачу на геометрическом языке: на прямой задан набор непересекающихся отрезков, нужно посчитать, сколько из них пересекаются с заданным отрезком. Видно, что центральным является алгоритм пересечения отрезков на прямой. Он формулируется следующим образом: пусть даны отрезки $[a, b]$ и $[c, d]$ ($a \leq b, c \leq d$). Найдём величины $\alpha = \max\{a, c\}$, $\beta = \min\{b, d\}$. Если $\alpha \leq \beta$, то исходные отрезки пересекаются и их пересечение есть отрезок $[\alpha, \beta]$. Если же $\alpha > \beta$, то исходные отрезки не пересекаются.

Соответственно, решение заключается в том, что поочерёдно пытаемся пересекать отрезок очень важного дела с каждым из отрезков дел из расписания, подсчитывая количество пересекающихся пар. Для удобства границы отрезков можно из формата `hh:mm` «часы–минуты» переводить в минуты $M = 60 \cdot hh + mm$.

Идеи тестов:

1. минимальный тест, $n = 0$;
2. максимальный тест, важное дело начинается в 00:00 и заканчивается в 23:59, n невелико;
3. максимальный тест, $n = 1440$;
4. максимальный тест, важное дело начинается в 00:00 и заканчивается в 23:59, $n = 1440$;
5. очень важное дело не пересекается ни с одним делом из расписания, $n = 100$;
6. очень важное дело не пересекается ни с одним делом из расписания, $n = 1439$;
- 7–13. случайные тесты, $n < 500$;
- 14–20. случайные тесты, $n > 600$.

9.5. «Делимость на 11». На парте в кабинете математики Петя обнаружил несколько длинных неотрицательных целых чисел. Он решил исследовать каждое число на делимость на 11: делится ли оно на 11 и, если нет, можно ли переставить в нём одну пару соседних цифр так, чтобы оно стало кратным 11. Помогите ему, напишите соответствующую программу. Напомним, что ведущие нули в записи числа не допускаются.

Программный комитет считает эту задачу сложной, так как здесь используется обработка строк вместе с алгоритмами из теории чисел (признак делимости на 11).

Напомним признак делимости на 11: считаем сумму цифр, стоящих на чётных позициях в десятичной записи числа; считаем сумму цифр, стоящих на нечётных позициях. Если разность этих сумм делится на 11, то и само число делится на 11.

Алгоритм обработки одного числа предлагается следующим: считываем строку, переводим для удобства все символы в соответствующие им числа. Первым проходом считаем суммы O и E цифр, стоящих на нечётных и чётных позициях соответственно. Если разность O и E делится на 11 (проверяется непосредственно с использованием операции `mod`, поскольку O и E не превосходят величины $128 \cdot 9 = 1152$), то исходное число делится на 11, о чём сообщаем и прекращаем обработку числа.

В противном случае надо попытаться найти пару цифр, перестановка которых превратит число в делящееся на 11. Пусть мы пытаемся переставить цифры d_1 и d_2 , первая из которых стоит на нечётном месте, а вторая — на чётном. Тогда новая разность будет равна

$$O' - E' = (O - d_1 + d_2) - (E - d_2 + d_1) = (O - E) + 2(d_2 - d_1), \quad (*)$$

то есть при перестановке соседних цифр разность $O - E$ может измениться только на чётную величину. Соответственно, если разность $O - E$ уже чётная, то никакая перестановка пары соседних цифр не превратит число в кратное 11; обработку можно остановить и сообщить о невозможности достижения цели. (Впрочем, эту проверку можно и опустить, сразу перейдя к непосредственному поиску. В этом случае он гарантированно завершится неудачей.)

Наконец, если потенциально нужная пара цифр может существовать, то попытаемся найти её прямым поиском: проходим по массиву цифр и, аккуратно учитывая, какая цифра из очередной пары стоит на чётном месте, а какая на нечётном, по формуле (*) вычисляем разность сумм после перестановки и проверяем её на делимость на 11. Если перестановка очередной пары цифр даёт делимость на 11, выводим индекс первой цифры из пары. Если мы прошли весь массив, но не нашли подходящей пары, сообщаем о невозможности превращения числа в кратное 11. При анализе перестановки нужно не забыть проверить, что она не ставит ноль на первое место в числе.

Заметим, что школьники, знающие языки, в библиотеку которых встроены функции длинной арифметики (например, Java), если и получают выгоду при решении данной задачи, то не слишком большую, поскольку операция перестановки двух цифр в числе не вполне тривиальна; её сложность на уровне указанной процедуры линейной обработки массива цифр.

Также отметим, что в силу наличия достаточно больших чисел неразумно пытаться использовать для их хранения стандартные числовые типы языков программирования. Участники, пренебрегшие этим соображением, получают около 70% полного балла.

Идеи тестов:

1. минимальный тест, все числа однозначные, n невелико;
2. минимальный тест, все числа однозначные, $n = 1000$;
3. все числа двузначные, есть делящиеся на 11, есть не делящиеся; n невелико;
4. все числа двузначные, есть делящиеся на 11, есть не делящиеся; $n = 1000$;
5. все числа трёхзначные, есть делящиеся на 11, есть те, которые можно превратить в кратные 11, есть те, которые нельзя превратить; n невелико;
6. все числа трёхзначные, есть делящиеся на 11, есть те, которые можно превратить в кратные 11, есть те, которые нельзя превратить; $n = 1000$;
7. максимальный тест, все числа 255-значные, есть делящиеся на 11, есть те, которые можно превратить в кратные 11, есть те, которые нельзя превратить; n невелико;
8. максимальный тест, все числа 255-значные, есть делящиеся на 11, есть те, которые можно превратить в кратные 11, есть те, которые нельзя превратить; $n = 1000$;
- 9–11. есть числа, в которых подходящая перестановка ставит на первое место ноль; есть другие подходящие перестановки;
- 12–14. есть числа, в которых подходящая перестановка ставит на первое место ноль; других подходящих перестановок нет;
- 15–20. случайные тесты, все числа не более чем 9-значные;
- 21–25. случайные тесты, есть числа, не входящие в `int64`.