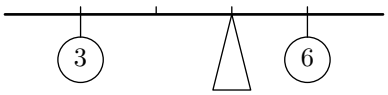
1. Рычаг

1 секунда 10 очков

Рассмотрим рычаг длиной L, на стороны которого подвешено N гирь. Для каждой гири известны место её прикрепления X_i (расстояние от места прикрепления до левого конца рычага) и масса M_i.

Написать программу, которая найдёт, где (на каком расстоянии от левого конца рычага) нужно поставить опору, чтобы рычаг был в равновесии. Считать, что собственная масса рычага равна нулю.



Входные данные. На первой строке текстового файла `kangsis.txt` находятся два разделённых пробелом целых числа: длина рычага L ($0 < L \leqslant 100$) и число гирь N ($1 \leqslant N \leqslant 100$). На каждой из следующих N строк находятся два разделённых пробелом целых числа: место прикрепления X_i ($0 \leqslant X_i \leqslant L$) и масса M_i ($0 < M_i \leqslant 100$) гири i.

Выходные данные. На единственную строку текстового файла `kangval.txt` вывести одно действительное число: искомое расположение опоры (её расстояние от левого конца рычага). Выведенное значение не должно отличаться от точного ответа более чем на 0,01.

Пример.

`kangsis.txt`	`kangval.txt`

`5 2`
`1 3`
`4 6`
	`3.0`

Оценивание. В тестах общей ценностью 5 очков дополнительно выполняется $N \leqslant 5$.

2. Сортировка карт

1 секунда 20 очков

Для записи положения в карточных играх чаще всего используется система обозначений, где каждая карта задаётся двумя знаками, которые показывают её достоинство и масть. Достоинства обозначаются как `AKQJT98765432` (*ace, king, queen, jack, ten, 9, . . . , 2* — туз, король, дама, валет, 10, 9, . . . , 2), а масти как `shdc` (*spades, hearts, diamonds, clubs* — пики, черви, бубни, крести). Например, пиковая дама обозначается `Qs`.

Написать программу, которая упорядочит заданные карты сначала по масти, а затем карты каждой масти между собой по достоинству (в порядке, приведённом в предыдущем абзаце).

Входные данные. На первой строке текстового файла `sortsis.txt` записано число карт N ($1 \leqslant N \leqslant 52$), а на второй строке состоящее из $2 \cdot N$ знаков выражение, которое описывает N карт (ни одна карта не повторяется).

Выходные данные. На единственную строку текстового файла `sortval.txt` вывести состоящее из $2 \cdot N$ знаков выражение, которое опишет заданные карты в требуемом порядке.

Пример.

`sortsis.txt`	`sortval.txt`
`5` `3d8sAdTd6c`	`8sAdTd3d6c`

3. Ханойская башня

1 секунда 30 очков

Ханойская башня — это головоломка, в которой даны три стержня и N дисков. Диаметры дисков $1 \dots N$, и дисков каждого диаметра ровно один. В центре каждого диска есть отверстие, через которое свободно проходит стержень. Первоначально все диски находятся на левом стержне в порядке убывания. Цель игрока — переложить их на правый стержень, используя средний стержень для временного хранения. На каждом шагу можно поднять самый верхний диск с одного стержня и разместить его сверху дисков, лежащих на каком-либо другом стержне. При этом никогда нельзя класть больший диск на меньший.

Написать программу, которая прочитает положение игры и найдёт наименьшее число перекладываний, с помощью которых из этого положения игру можно доиграть до конца (т.е. разместить все диски на правом стержне в правильном порядке).

Входные данные. На первой строке текстового файла `tornsis.txt` в начале записано число дисков на левом стержне N_v, а затем N_v разделённых пробелами целых чисел: диаметры дисков, перечисленные снизу вверх; на второй строке файла в таком же виде записано число дисков на центральном стержне N_k и их диаметры, а на третьей строке — число дисков на правом стержне N_p и их диаметры ($0 \leqslant N_v$, $0 \leqslant N_k$, $0 \leqslant N_p$, $N_v + N_k + N_p \leqslant 30$). Можно предполагать, что положение соответствует правилам игры (каждый диск $1 \dots N_v + N_p + N_k$ присутствует ровно один раз, и нигде больший диск не лежит на меньшем).

Выходные данные. На единственную строку текстового файла `tornval.txt` вывести одно целое число: наименьшее количество шагов, за которое игру из заданного положения можно доиграть до конца.

Пример.

`tornsis.txt`
```
2 2 1
1 3
0
```

`tornval.txt`
```
4
```

Заданную во входных данных игру можно доиграть до конца четырьмя перекладываниями так: диск 3 из центра направо; диск 1 слева в центр; диск 2 слева направо; диск 1 из центра направо. За три хода игру нельзя закончить, так как тогда нужно было бы каждый диск одним ходом сразу переложить направо; это невозможно, так как тогда диск 1 оказался бы под диском 2.

Оценивание. В тестах общей ценностью 20 очков дополнительно $N_v + N_k + N_p \leqslant 10$.

4. Сумма цифр

1 секунда 40 очков

Для натурального числа можно рассматривать сумму его цифр. Например, сумма цифр числа 123 равна $1 + 2 + 3 = 6$, а сумма цифр 99 равна $9 + 9 = 18$.

Написать программу, которая найдёт количество натуральных чисел меньших заданного числа, сумма цифр каждого из которых равна сумме цифр заданного числа.

Входные данные. На единственной строке текстового файла `ristsis.txt` находится целое число N ($0 \leqslant N \leqslant 10^{18}$).

Выходные данные. На единственную строку текстового файла `ristval.txt` вывести количество натуральных чисел меньших N, сумма цифр каждого из которых равна сумме цифр числа N.

Пример.

`ristsis.txt`
```
123
```

`ristval.txt`
```
9
```

Учитываются числа 6, 15, 24, 33, 42, 51, 60, 105, 114.

Оценивание. В тестах общей ценностью 20 очков дополнительно выполняется $N \leqslant 10^9$, в том числе в тестах общей ценностью 10 очков условие $N \leqslant 10^3$.

5. Поедание шарфиков

5 секунд 50 очков

У дизайнера Мартина в шкафу есть N красивых шарфиков, с которыми по субботам он ходит на встречи дизайнеров. Каждый шарфик имеет свою художественную ценность, и соответственно этому на встрече Мартин получает признание. Придти два раза на встречу с одним и тем же шарфиком было бы большим *faux pas*, и Мартин никогда этого не делает. Уже ношенные шарфики он складывает обратно в шкаф, но больше никогда их не одевает.

В шкафу Мартина также живёт моль Кира, которая каждое воскресенье проедает дырку в одном шарфике. Кира никоим образом не интересуется художественной ценностью и ест шарфики случайным образом. Вероятность проедания шарфика пропорциональна его длине. Кира может проесть один и тот же шарфик и несколько раз.

Ясно, что Мартин может сходить на встречи не более N раз, и то, если Кира будет проедать только уже ношенные шарфики. Если Кира иной раз будет проедать и неношенные шарфики, то число встреч соответственно уменьшится.

Написать программу, которая найдёт, сколько признания в среднем получит Мартин, если он будет использовать наилучшую стратегию в то время, как Кира будет проедать шарфики случайным образом. Процесс начинается в начале недели, поэтому ко времени первой встречи все шарфики еще целые.

Входные данные. На первой строке текстового файла `sallsis.txt` находится количество шарфиков N ($1 \leqslant N \leqslant 20$). На каждой из следующих N строк находятся два целых числа M_i ja P_i ($1 \leqslant M_i \leqslant 100$, $1 \leqslant P_i \leqslant 20$): художественная ценность и длина одного шарфика.

Выходные данные. На единственную строку текстового файла `sallval.txt` вывести одно действительное число: среднее количество признания Мартина, если он выбирает шарфики оптимально, а Кира проедает их случайным образом. Выведенное значение не должно отличаться от точного ответа более чем на 0,001.

Пример.

`sallsis.txt`
```
3
10 3
20 2
30 1
```

`sallval.txt`
```
48.333
```

При этих входных данных оптимальной стратегией Мартина будет сначала взять второй шарфик. Затем:

- с вероятностью $\frac{1}{2}$ дыра будет в первом шарфике, для второго собрания останется только третий шарфик, и всего получим 50 очков признания;
- с вероятностью $\frac{1}{3}$ дыра будет во втором шарфике, тогда останутся два шарфика; возьмём из них сначала третий, тогда с вероятностью $\frac{1}{2}$ сможем одеть и первый; в среднем 55 очков признания;
- с вероятностью $\frac{1}{6}$ дыра будет в третьем шарфике, тогда останется только первый, и получим 30 очков признания.

Взвешенное среднее этих вариантов и будет $48\frac{1}{3}$ очков признания.

Пример.

`sallsis.txt`
```
4
10 10
20 9
30 5
40 1
```

`sallval.txt`
```
84.94144
```

Оценивание. В тестах общей ценностью 25 очков дополнительно выполняется $N \leqslant 10$.

6. Площадь

Тестирование 50 очков

Инженерное бюро заказала у разработчиков программу, которая должна уметь вычислять площадь пересечения треугольника и четырёхугольника. Программа должна уметь вычислять такие пересечения любой формы.

Составить комплект тестовых данных для инженерного бюро, с помощью которого будет возможно проверить корректность этой программы прежде, чем платить разработчикам.

Входные данные. На первой строке текстового файла `pindsis.txt` находятся шесть разделённых пробелами целых числа A_x, A_y, B_x, B_y, C_x, C_y: координаты вершин треугольника ABC. На второй строке файла подобным образом описаны координаты четырёхугольника $DEFG$ в порядке появления по периметру четырёхугольника. Все координаты — целые числа, абсолютные значения которых не превосходят 100.

Выходные данные. На единственной строке текстового файла `pindval.txt` находится одно действительное число: площадь пересечения описанных во входных данных фигур, которая не должна отличаться от точного ответа более чем на 0,01.

Пример.

`pindsis.txt`
```
1 1 5 1 3 5
1 3 5 3 5 6 1 6
```

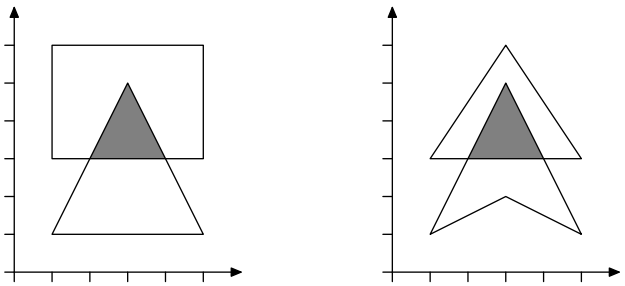
`pindval.txt`
```
2.0
```

Пример.

`pindsis.txt`
```
1 3 5 3 3 6
1 1 3 2 5 1 3 5
```

`pindval.txt`
```
2.0
```

На рисунке слева отображены входные данные первого примера, а справа — второго.



Оценивание. В этом задании в качестве решения нужно предоставить один ZIP-файл, в котором должно быть до 15 файлов со входными данными, названные `pindsis01.txt`, `pindsis02.txt`, ..., и соответствующие им файлы с выходными данными, названные `pindval01.txt`, `pindval02.txt`, ...

Если какая-то пара файлов не состоит из корректного файла со входными данными и соответствующего ему корректного выходного файла, то за эту пару участник очки не получит. Корректные пары файлов будут использованы для тестирования некоторых программ с ошибками, и участник получит определённое количество очков за каждую программу, которая даст неправильный ответ или завершится ошибкой по крайне мере в одном тесте, составленным этим участником.

7. Чёрный ящик

1 секунда 75 очков

Для выявления ошибок при вводе, передаче или сохранении данных часто используются контрольные суммы. Например, если отправитель хочет передать послание M, он вычисляет значение $S = f(M)$, где f — это какая-то заранее оговорённая функция, и вместо послания M передаёт пару $\langle M, S\rangle$. Получатель, приняв из канала связи пару $\langle M', S'\rangle$, проверяет равенство $f(M') = S'$. Если оно не выполняется, значит при передаче данных точно произошла ошибка, и следует попросить послать сообщение заново.

Если равенство выполняется, то полной гарантии, что данные дошли в порядке, всё равно нет. А именно, может произойти так, что в канале связи были повреждены как сообщение M, так и значение S, причём таким (случайным или злоумышленным) образом, что проверка на стороне получателя не может этого определить. Вероятность такого ложноположительного результата можно уменьшить, выбрав функцию f подходящим способом. В этом задании рассмотрим различные типы функций, используемых для вычисления контрольных сумм.

Бит чётности.[1] Самая простая контрольная сумма имеет размер всего 1 бит. Значение этого бита выбирается в зависимости от количества единиц в двоичной записи передаваемых данных. На протяжении времени использовались разные варианты этой схемы: в каком-то варианте договариваются, что в сообщении и контрольной сумме вместе должно быть чётное количество единиц, в другом варианте, что нечётное; в каком-то варианте контрольный бит добавляется в конец данных, в другом — в начало. Так как в этой схеме возможных значений контрольной суммы всего два, вероятность ложноположительности весьма велика — целых 50%. С другой стороны, такую контрольную сумму очень просто реализовать в электронном виде.

Сумма цифр.[2] Немного более надёжная схема состоит в том, чтобы вместе с данными посылать и сумму их цифр. Например, если вследствие проблем с каналом связи вместо настоящего значения каждого бита передаётся бит 0, и такое ошибочное состояние продолжается как раз столько времени, что битом 0 окажется заменено чётное количество битов 1, то чётный бит не поможет распознать эту ошибку, тогда как сумма цифр при такой ошибке обязательно станет меньше.

Взвешенная сумма цифр.[3] Простая сумма цифр не поможет понять, что в сообщении изменился порядок знаков (что легко может случиться, например, когда человек вводит данные). Для выявления таких ошибок, в банках, например, к номерам счетов и ссылок добавляется контрольная цифра такая, что если умножать значения цифр на коэфициенты $3, 7, 1, 3, 7, 1, \ldots$, то при корректном номере счёта или ссылки полученная в результате сумма будет делится на 10.

CRC.[4] Суммы CRC значительно сложнее предыдущих способов. При их вычислении биты рассматриваются как множители многочлена, а контрольная сумма вычисляется как остаток при делении многочлена P, заданного битами передаваемого сообщения, на какой-то зафиксированный многочлен G (как для целых чисел P и G можно найти частное Q и остаток R такие, что $P = G \cdot Q + R$, так и многочлены можно делить друг на друга с остатком; следует обратить внимание на то, что множители многочлена рассматриваются как однобитные целые числа, и при их сложении вследствие переполнения имеет место равенство $1+1 = 0$). Используются различные вариации этой схемы: с различной длиной контрольной суммы или различными делителями; также варьируется порядок чтения как битов в одном байте, так и байтов в строке из нескольких байтов.

Криптографические хеш-функции.[5] Вышеописанные контрольные суммы задуманы для выявления случайных ошибок и не защищают от целенаправленной атаки. Действительно, после изменения сообщения злоумышленник может пересчитать контрольную сумму и отправить её вместе с изменённым сообщением. Чтобы исключить такую возможность от схемы вычисления контрольной суммы требуют два дополнительных свойства.

Во-первых, нужно использовать такую функцию f, при которой сложно изменить сообщение так, чтобы контрольная сумма изменённого сообщения оказалась такой же как и изначального. Крипто-

[1] http://en.wikipedia.org/wiki/Parity_bit
[2] http://en.wikipedia.org/wiki/Digit_sum
[3] http://en.wikipedia.org/wiki/Check_digit
[4] http://en.wikipedia.org/wiki/Cyclic_redundancy_check
[5] http://en.wikipedia.org/wiki/Cryptographic_hash_function

графические хеш-функции, из которых наиболее известны MD5, SHA1 и семейство SHA2, как раз такие.

Во-вторых, у отправителя и получателя должен быть секрет (секретное слово), которого не знает злоумышленник. В простейшем случае секрет P можно использовать так: вместо $f(M)$ отправитель вычисляет контрольную сумму в виде $S = f(MP)$ (при вычислении контрольной суммы секрет P дописывается в конец сообщения M), а передаёт дальше всё так же $\langle M, S\rangle$. Если теперь злоумышленник захочет поменять сообщение M на сообщение M', то он не сможет этого сделать, так как не знает секрета P и следовательно не может вычислить значение $f(M'P)$. Зная секрет, получатель сможет вычислить контрольную сумму и проверить её.

По адресу `http://prog.offline.ee/html2/kast.cgi` находятся пять веб-приложений, которые вычисляют контрольные суммы по описанным выше схемам. Для каждого приложения найти, какой вариант какой схемы (для хеш-функции также и какой секрет) использует данное приложение, и написать программу, которая будет вычислять точно такие же контрольные суммы.

Оценивание. Каждое приложение — отдельное подзадание. В качестве решения каждого подзадания нужно предоставить отдельную программу, которая будет и оцениваться отдельно. Программы тестируются с помощью входных и выходных данных, описанных в файле. Веб-интерфейс для своих решений делать не требуется.

Входные данные. На первой строке текстового файла `kastsis.txt` находится число сообщений N ($1 \leqslant N \leqslant 100$) и на каждой из следующих N строк — одно сообщение. Можно считать, что

- все сообщения состоят из печатаемых знаков 7-битной ASCII таблицы;
- длина ни одного сообщения не превышает 80 знаков;
- для тестирования используются только те сообщения, которые принимает соответствующее веб-приложение.

Выходные данные. В текстовый файл `kastval.txt` вывести ровно N строк. На i строку вывести сообщение, находившееся во входных данных на $i + 1$ строке вместе с контрольной суммой в точно таком же формате, который используется в соответствующем веб-приложении.

Пример.

`kastsis.txt`	`kastval.txt`
`2`	`246`
`123`	`912`
`456`	