1. Делимость чисел

1 секунда 20 очков

Написать программу, которая найдёт из трёх заданных целых чисел все различные пары делящихся чисел.

Числа x и y будем называть парой делящихся чисел среди чисел A_1, A_2 и A_3, если число y делится на число x и найдутся i и j, при которых $x = A_i$, $y = A_j$ и $i \neq j$. Пару x_1 и y_1 и пару x_2 и y_2 будем называть различными, если $x_1 \neq x_2$ или $y_1 \neq y_2$ (или оба неравенства вместе).

Входные данные. На единственной строке текстового файла `jagu.sis` даны три разделённых пробелом целых числа A_1, A_2, A_3, принимающие значения $1 \ldots 10\,000$.

Выходные данные. На каждую строку текстового файла `jagu.val` вывести два разделённых пробелом целых числа x и y — одну пару удовлетворяющую привидённым выше условиям. Порядок строк в файле не важен, но порядок чисел в строке важен. Если нет ни одной пары делящихся чисел, на единственную строку файла вывести слово `POLE`.

Пример.

jagu.sis	jagu.val
1 2 3	1 2 1 3

Пример.

jagu.sis	jagu.val
1 2 2	1 2 2 2

Пример.

jagu.sis	jagu.val
2 3 5	POLE

Оценивание. В этом задании за тесты с ответом `POLE` очки получат только те программы, которые правильно решат по крайней мере один тест, в котором пары делящихся чисел найдутся.

2. Очередь

1 секунда 40 очков

В точку обслуживания, где обслуживание одного клиента занимает M минут, в заранее неизвестные моменты времени прибывают клиенты. Если по прибытии клиента персонал свободен, то он сразу приступает к обслуживанию этого клиента. В противном случае клиент становится в очередь, а персонал начинает обслуживать следующего клиента в очереди как только заканчивает работу с предыдущим клиентом.

Написать программу, которая находит время завершения работы с последним клиентом на основании моментов времени прибытия клиентов.

Входные данные. На первой строке текстового файла `saba.sis` дано два разделённых пробелом целых числа — количество клиентов N ($1 \leqslant N \leqslant 100$) и время, затрачиваемое на обслуживание одного клиента M ($1 \leqslant M \leqslant 100$). На каждой из следующих N строк дано время прибытия одного клиента в формате $HH{:}MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$). Строки идут в порядке прибытия клиентов в очередь.

Выходные данные. На единственную строку текстового файла `saba.val` вывести время завершения обслуживания последнего клиента тоже в формате $HH{:}MM$. Можно учитывать, что ответ не превышает 23:59.

Пример.

saba.sis	saba.val
3 20	13:50
12:20	
13:10	
13:20	

Обслуживание первого клиента завершается в 12:40. Обслуживание второго клиента завершается в 13:30. Таким образом третий клиент должен ждать до этого времени, и его обслуживание завершится в 13:50.

3. Наличность

3 секунды 40 очков

Как известно, в Эстонии последние недели в обращении находятся 1-, 2-, 5-, 10-, 25-, 50-, 100- и 500-кроновые купюры.

В память о них, напишите программу, которая сможет найти, сколько есть различных способов выплатить ими данную сумму.

Две способа считаются разными, если по крайней мере один номинал использован разное количество раз. Использование тех же купюр в другом порядке не считается отличным способом.

Входные данные. На единственной строке текстового файла `raha.sis` дано целое число S ($0 \leqslant S \leqslant 250$) — выплачиваемая сумма.

Выходные данные. На единственную строку текстового файла `raha.val` вывести количество различных способов выплаты.

Пример.

raha.sis	raha.val
6	5

Для выплаты суммы в 6 крон есть следующие способы:
$6 = 1 + 1 + 1 + 1 + 1 + 1$,
$6 = 2 + 1 + 1 + 1 + 1$,
$6 = 2 + 2 + 1 + 1$,
$6 = 2 + 2 + 2$,
$6 = 5 + 1$.

1. Предыдущее число

1 секунда 20 очков

Рассмотрим следующее правило для получения из четырёхзначного числа нового четырёхзначного числа:

$$2345 \to 23 + 2 + 3 = 28,\ 45 + 4 + 5 = 54 \to 2854.$$

Правило применимо только если результаты обойх сложении двухзначные числа.

Написать программу для нахождения предыдущего числа по отношению к данному в последовательности, заданной по описанному выше правилу.

Входные данные. На единственной строке текстового файла `eelm.sis` находится четырёхзначное положительное целое число N.

Выходные данные. На единственную строку текстового файла `eelm.val` вывести четырёхзначное положительное целое число M, из которого по описанному выше правилу получается число N. Если подходящих вариантов несколько, вывести любой из них. Если нет ни одного подходящего варианта, вывести слово `POLE`.

Пример.

`eelm.sis`	`eelm.val`
`2854`	`2345`

Пример.

`eelm.sis`	`eelm.val`
`1010`	`POLE`

Оценивание. В этом задании за тесты с ответом `POLE` очки получат только те программы, которые правильно решат по крайней мере один тест, в котором предыдущее число найдётся.

2. Развоз товара

1 секунда 40 очков

Машина развозит товар со склада в несколько магазинов. Известно, сколько времени уходит на погрузку товара на складе и сколько на разгрузку товара в каждом магазине. Также известно, как долго длится поездка из одного места в другое.

Кроме того, для склада и каждого магазина известен промежуток времени, в течение которого можно производить там погрузку или разгрузку. Если машина оказывается у склада или магазина раньше начала этого промежутка времени, ей придётся подождать. Если машина оказывается у какого-либо магазина так поздно, что завершить разгрузку в отведённое время она уже не успеет, то в этот магазин она и не сможет доставить товар.

Написать программу, которая проверит, сможет ли машина доставить товар во все магазины и отвести отчет обратно на склад до его закрытия, а также найдёт возможность сделать это за наименьшее время для водителя.

Входные данные. На первой строке текстового файла `kaup.sis` находится одно целое число — количество магазинов N ($1 \leqslant N \leqslant 100$). На второй строке — данные склада, и на последующих N строках — данные магазинов в порядке их посещения.

На каждой из этих $N + 1$ строк находятся четыре разделённых пробелом поля данных, время открытия и закрытия склада или магазина, количество минут, затрачиваемое на погрузку или разгрузку товара и на поездку в следующее место (для склада — в первый магазин, для последнего магазина — обратно на склад, в другом случае — в следующий магазин). Первые два поля даны в формате $HH:MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$).

Выходные данные. На первую строчку текстового файла `kaup.val` вывести слово `JAH` или `EI` соответственно, если у машины есть возможность доставить товар во все магазины или нет. Если возможность есть, то вывести на вторую строку файла наименьшее необходимое количество минут, то есть наименьшую длину промежутка времени с прибытия на складе для погрузки до прибытия обратно на склад после посещения всех магазинов, а на третью строчку вывести самое раннее время начала погрузки на складе для описанного выше оптимального плана.

Пример.

`kaup.sis`
```
2
06:00 23:00 20 10
07:00 08:00 10 20
08:00 09:00 10 20
```

`kaup.val`
```
JAH
90
07:00
```

Оптимальный план: машина прибывает на склад в 07:00, загружается 20 минут и 10 минут едет к первому магазину; прибывает к первому магазину в 07:30, разгружается 10 минут и 20 минут едет ко второму магазину; прибывает ко второму магазину в 08:00, разгружается 10 минут и 20 минут едет обратно на склад; прибывает обратно на склад в 08:30, то есть спустя 90 минут после первого прибытия.

Пример.

`kaup.sis`
```
1
07:00 08:00 20 20
07:00 08:00 20 20
```

`kaup.val`
```
EI
```

Так как на погрузку, разгрузку и поездки уходит 80 минут, то невозможно прибыть обратно на склад раньше его закрытия.

3. Замкнутый круг

1 секунда 40 очков

Большие программы часто разделяются на модули. Каждый модуль содержит функции и классы, и если в каком-либо другом модуле будут нужны эти функии или классы, то этот другой модуль может импортировать первый модуль.

Например для того, чтобы модуль `Game`, отвечающий за игровую логику, смог использовать функции из модулей `Math` и `IO`, в коде модуля `Game` можно написать команду "`import Math, IO`". Модуль `IO` может в свою очередь импортировать модули `Screen` и `File` с помощью команды "`import Screen, File`".

Если один модуль импортирует второй, он автоматически получает доступ и ко всем модулям, которые импортирует второй модуль, а также и ко всем, которые импортируют модули, импортированные вторым модулем, и т.д.

Скажем, что модуль X использует модуль Y, если модуль X импортирует модуль Y или найдётся такая последовательность модулей A_1, A_2, ..., A_n, что X импортирует A_1, A_1 импортирует A_2, ..., A_n импортирует Y.

Написать программу, проверяющую, составляют ли модули "замкнутый круг", при котором модуль X использует модуль Y, а модуль Y в свою очередь использует модуль X.

Входные данные. На первой строке текстового файла `ring.sis` дано целое число N ($1 \leqslant N \leqslant 1000$) — количество модулей в программе. Модули обозначены целыми числами $1 \ldots N$.

На каждой из следующих N строк файла дано описание одного модуля. На строке $1 + i$ сначала дано количество K_i ($0 \leqslant K_i \leqslant 20$) модулей, импортируемых модулем i, а после — обозначения $A_{i,1}$, $A_{i,2}$, ..., A_{i,K_i} этих модулей, разделённые пробелами.

Выходные данные. Если модули не составляют ни одного "замкнутого круга", вывести на единственную строку текстового файла `ring.val` слово `EI`. В другом случае вывести на первую строку файла слово `JAH`, а на вторую — разделённые пробелом обозначения двух модулей, которые используют друг друга. Если "замкнутых кругов" несколько, вывести любой из них.

Пример.

`ring.sis`
```
4
2 2 3
2 3 4
0
0
```

`ring.val`
```
EI
```

Модуль 1 импортирует два модуля — 2 и 3. Модуль 2 также импортирует два модуля — 3 и 4. Модули 3 и 4 ничего не импортируют.

Пример.

`ring.sis`
```
4
2 4 2
1 3
1 2
0
```

`ring.val`
```
JAH
2 3
```

Оценивание. В этом задании за тесты с ответом `EI` очки получат только те программы, которые правильно решат по крайней мере один тест с ответом `JAH`.

1. Делимость чисел

1 секунда 20 очков

Написать программу, которая найдёт из трёх заданных целых чисел все различные пары делящихся чисел.

Числа x и y будем называть парой делящихся чисел среди чисел A_1, A_2 и A_3, если число y делится на число x и найдутся i и j, при которых $x = A_i$, $y = A_j$ и $i \neq j$. Пару x_1 и y_1 и пару x_2 и y_2 будем называть различными, если $x_1 \neq x_2$ или $y_1 \neq y_2$ (или оба неравенства вместе).

Входные данные. На единственной строке текстового файла `jagu.sis` даны три разделённых пробелом целых числа A_1, A_2, A_3, принимающие значения $1 \ldots 10\,000$.

Выходные данные. На каждую строку текстового файла `jagu.val` вывести два разделённых пробелом целых числа x и y — одну пару удовлетворяющую привидённым выше условиям. Порядок строк в файле не важен, но порядок чисел в строке важен. Если нет ни одной пары делящихся чисел, на единственную строку файла вывести слово `POLE`.

Пример.

jagu.sis	jagu.val
1 2 3	1 2 1 3

Пример.

jagu.sis	jagu.val
1 2 2	1 2 2 2

Пример.

jagu.sis	jagu.val
2 3 5	POLE

Оценивание. В этом задании за тесты с ответом `POLE` очки получат только те программы, которые правильно решат по крайней мере один тест, в котором пары делящихся чисел найдутся.

2. Очередь

1 секунда 40 очков

В точку обслуживания, где обслуживание одного клиента занимает M минут, в заранее неизвестные моменты времени прибывают клиенты. Если по прибытии клиента персонал свободен, то он сразу приступает к обслуживанию этого клиента. В противном случае клиент становится в очередь, а персонал начинает обслуживать следующего клиента в очереди как только заканчивает работу с предыдущим клиентом.

Написать программу, которая находит время завершения работы с последним клиентом на основании моментов времени прибытия клиентов.

Входные данные. На первой строке текстового файла `saba.sis` дано два разделённых пробелом целых числа — количество клиентов N ($1 \leqslant N \leqslant 100$) и время, затрачиваемое на обслуживание одного клиента M ($1 \leqslant M \leqslant 100$). На каждой из следующих N строк дано время прибытия одного клиента в формате $HH{:}MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$). Строки идут в порядке прибытия клиентов в очередь.

Выходные данные. На единственную строку текстового файла `saba.val` вывести время завершения обслуживания последнего клиента тоже в формате $HH{:}MM$. Можно учитывать, что ответ не превышает 23:59.

Пример.

saba.sis	saba.val
3 20	13:50
12:20	
13:10	
13:20	

Обслуживание первого клиента завершается в 12:40. Обслуживание второго клиента завершается в 13:30. Таким образом третий клиент должен ждать до этого времени, и его обслуживание завершится в 13:50.

3. Наличность

3 секунды 40 очков

Как известно, в Эстонии последние недели в обращении находятся 1-, 2-, 5-, 10-, 25-, 50-, 100- и 500-кроновые купюры.

В память о них, напишите программу, которая сможет найти, сколько есть различных способов выплатить ими данную сумму.

Две способа считаются разными, если по крайней мере один номинал использован разное количество раз. Использование тех же купюр в другом порядке не считается отличным способом.

Входные данные. На единственной строке текстового файла `raha.sis` дано целое число S ($0 \leqslant S \leqslant 250$) — выплачиваемая сумма.

Выходные данные. На единственную строку текстового файла `raha.val` вывести количество различных способов выплаты.

Пример.

raha.sis	raha.val
6	5

Для выплаты суммы в 6 крон есть следующие способы:
$6 = 1 + 1 + 1 + 1 + 1 + 1$,
$6 = 2 + 1 + 1 + 1 + 1$,
$6 = 2 + 2 + 1 + 1$,
$6 = 2 + 2 + 2$,
$6 = 5 + 1$.