Технокубок 2017/2018. Четвёртый отборочный раунд. Условия и разбор задач.

A. Маша и медведи

ограничение по времени на тест

2 секунды

ограничение по памяти на тест

256 мегабайт

ввод

стандартный ввод

вывод

стандартный вывод

У семьи из папы-медведя, мамы-медведя и сына-медведя есть три автомобиля. Известно, что в самый большой автомобиль папа-медведь влезает и он ему нравится. Аналогично, в средний автомобиль влезает мама-медведь, и он ей нравится. Сын-медведь влезает в самый маленький автомобиль автомобиль, он ему нравится. Известно, что самый большой автомобиль по объему строго больше среднего, а тот — строго больше самого маленького.

Когда протестировать автомобили пришла Маша, она смогла влезть во все, но только самый маленький понравился ей.

Известно, что персонаж объема a влезает в автомобиль объема b, если и только если $a \le b$; если влезть в машину удалось, этот автомобиль персонажу нравится, если и только если $2a \ge b$.

Вам даны объемы Маши и медведей. Найдите какие-нибудь подходящие целочисленные неотрицательные объемы автомобилей.

Входные данные

Вам дано 4 целых числа V_1, V_2, V_3, $V_m (1 \le V_i \le 100)$ — объёмы папы-медведя, мамы-медведя, сына-медведя и Маши соответственно. Гарантируется что $V_1 > V_2 > V_3$

Выходные данные

Выведите 3 целых числа — размеры автомобилей папы-медведя, мамы-медведя и сына-медведя соответственно. Если существует несколько решений, выведите любое. Если решения не существует, в единственной строке выведите «`-1`» (без кавычек).

Примеры

входные данные

```
50 30 10 10
```

выходные данные

```
50
30
10
```

входные данные

```
100 50 10 21
```

выходные данные

```
-1
```

Примечание

Легко видеть, что в первом тесте все условия на размеры автомобилей удовлетворены.

Во втором тестовом примере ответа не существует, так как в самый маленький автомобиль должна помещаться Маша (т.е. его минимальный размер — 21), но в тоже время сыну-медведю он должен нравиться, а значит его максимальный размер — 20.

907A - Маша и медведи

Размеры автомобилей должны удовлетворять следующим ограничениям:

1. В i-й автомобиль могут сесть Маша и соответствующий медведь, поэтому его размер не меньше $max(V_i, V_m)$;
2. Каждому медведю нравится его автомобиль, поэтому размер i-го автомобиля не больше, чем $2 \cdot V_i$;
3. Маше не понравились первые два автомобиля, поэтому их размеры строго больше, чем $2 \cdot V_m$;
4. Маше понравился третий автомобиль, так что его размер не больше чем $2 \cdot V_m$;
5. Размеры автомобилей строго упорядочены. Это обозначает, что размер автомобиля папы-медведя строго больше размера автомобиля мамы-медведя, который в свою очередь больше, чем размер автомобиля сына-медведя.

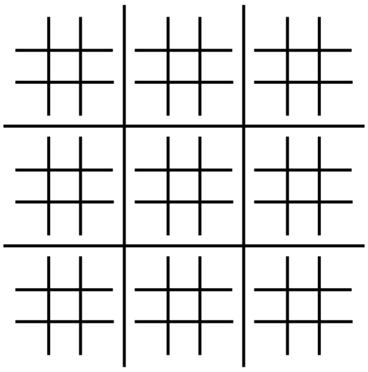
Размеры медведей не превосходят 100, поэтому размеры автомобилей не превосходят 200 и существует не более 200^3возможных вариантов размеров автомобилей. В данных ограничениях можно просто перебрать все возможные тройки размеров и проверить, удовлетворяет ли каждая из них вышеперечисленным ограничениям или нет.

B. Крестики-нолики

ограничение по времени на тест
2 секунды
ограничение по памяти на тест
256 мегабайт
ввод
стандартный ввод
вывод
стандартный вывод

Два медведя играют в крестики-нолики по переписке. Медведям давно надоели обычные крестики-нолики на доске три на три, поэтому они играют в чуть более сложную версию этой игры. Опишем её правила.

Игра происходит на следующем поле:



Игроки ходят по очереди. На первом ходу первый игрок может поставить свою фишку в любую клетку любого маленького поля. Далее на ходы накладываются ограничения — если на предыдущем ходу оппонент поставил свою фишку в клетку с координатами (x_l, y_l) в некотором маленьком поле, то свой текущий ход игрок обязан совершить в маленькое поле с координатами (x_l, y_l). Например, если первый игрок своим ходом поставит свою фишку в нижнюю левую клетку центрального поля, то второй игрок обязан своим ходом поставить свою фишку в нижнее левое поле (обратите внимание на первый тестовый пример). Если в поле, в которое игрок должен ставить фишку, нет свободных клеток, то он может сделать ход в любую свободную клетку любого поля.

Вам дано текущее поле игры, а также координаты клетки, в которую был сделан последний ход. Вам необходимо определить, в какие клетки может поставить свою фишку текущий игрок.

Почтальоном в лесу работает заяц и он любит пакостить медведям. Поэтому он иногда немного перерисовывает игровое поле, но делает это так, чтобы медведи не заметили. А именно: в клетке, в которую был сделан последний ход, обязательно стоит фишка одного из игроков. Поэтому текущая позиция может быть недостижимой. Вам не нужно выяснять, достижима текущая позиция или нет, просто выведите возможные ходы следующего игрока.

Входные данные

В первых 11 строках вам задана таблица 9 на 9, которая пробелами и переводами строк разбита на 9 маленьких полей, внутри каждого из которых описано соответствующее поле без пробелов и переводов строк. Символ «x» (ASCII-код 120) обозначает, что соответствующая клетка занята фишками первого игрока, «o» (ASCII-код 111) — фишками второго, символ «.» (ASCII-код 46) обозначает пустую клетку.

После таблицы в единственной строке записаны два целых числа x и y $(1 \le x, y \le 9)$ — координаты клетки, в которую был сделан последний ход. Строки в таблице нумеруются сверху вниз, а столбцы — слева направо. Гарантируется, что в клетке, в которую был совершен последний ход находится либо символ «x», либо символ «o».

Также гарантируется, что существует хотя бы одна свободная клетка. Не гарантируется, что данная позиция достижима.

Выходные данные

Выведите поле в аналогичном формате, где все клетки, в которые может поставить свою фишку очередной игрок заменены на «!» (ASCII-код 33). Все остальные клетки должны остаться без изменений.

Примеры

входные данные

```
... ... ...
... ... ...
... ... ...

... ... ...
... ... ...
... x.. ...

... ... ...
... ... ...
... ... ...
6 4
```

выходные данные

```
... ... ...
... ... ...
... ... ...

... ... ...
... ... ...
... x.. ...

!!! ... ...
!!! ... ...
!!! ... ...
```

входные данные

```
xoo x.. x..
ooo ... ...
ooo ... ...

x.. x.. x..
... ... ...
... ... ...

x.. x.. x..
... ... ...
... ... ...
7 4
```

выходные данные

```
xoo x!! x!!
ooo !!! !!!
ooo !!! !!!

x!! x!! x!!
!!! !!! !!!
!!! !!! !!!

x!! x!! x!!
!!! !!! !!!
!!! !!! !!!
```

входные данные

```
o.. ... ...
... ... ...
... ... ...

... xxx ...
... xox ...
... ooo ...

... ... ...
... ... ...
... ... ...
5 5
```

выходные данные

```
o!! !!! !!!
!!! !!! !!!
!!! !!! !!!

!!! xxx !!!
!!! xox !!!
!!! ooo !!!
```

```
!!! !!! !!!
!!! !!! !!!
!!! !!! !!!
```

Примечание

Разберём первый тестовый пример.

В нём первый игрок сделал свой первый ход в нижнюю левую клетку центрального поля, поэтому второй игрок может ставить свою фишку только в нижнее левое поле.

Во втором тесте последний ход был сделан в верхнюю левую клетку нижнего центрального поля, однако все клетки в верхнем левом поле уже заняты, поэтому второй игрок может ставить свою фишку в любую свободную клетку любого поля.

В третьем тесте последний ход был сделан в центральную клетку центрального поля, однако все клетки там уже заняты, поэтому первый игрок может ставить фишку в любую свободную клетку любого поля. Обратите внимание, что такая позиция недостижима.

907B - Крестики-нолики

Опишем каждую ячейку поля четырьмя числами (x_b, y_b, x_s, y_s), $0 \le x_b, y_b, x_s, y_s \le 2)$ где (x_b, y_b) являются координатами маленького поля в большом, а (x_s, y_s) — координаты ячейки в маленьком поле. Легко видеть, что «обычные» координаты (x, y), $1 \le x, y \le 9$ и наши новые (x_b, y_b, x_s, y_s) связаны друг с другом следующими равенствами, с помощью которых между ними можно осуществлять переход:

- $x_b = \lfloor((x - 1) / 3)\rfloor, y_b = \lfloor((y - 1) / 3)\rfloor;$
- $x_s = (x - 1)\ mod\ 3, y_b = (y - 1)\ mod\ 3;$
- $x = 3 * x_b + x_s + 1, y = 3 * y_b + y_s + 1.$

В терминах новых координат, если последний ход был (x_b, y_b, x_s, y_s), то следующий ход должен быть сделан в произвольную свободную клетку с координатами (x_s, y_s, x', y'), для некоторых $x', y' \in [0, 2]$, если это возможно. В противном случае ход может быть сделан в любую свободную ячейку. Чтобы решить задачу достаточно перебрать все пары (x', y') и записать « ! » в каждую свободную ячейку (x_s, y_s, x', y'). Если таких свободных ячеек нет, то следует записать « ! » во все свободные клетки на поле.

C. Шокеры

ограничение по времени на тест

2 секунды

ограничение по памяти на тест

256 мегабайт

ввод

стандартный ввод

вывод

стандартный вывод

Валентин участвует в шоу, которое называется «Шокеры». Суть проста — жюри загадало некоторую букву, неизвестную Валентину. Он должен выступить с некоторой речью, но за каждое произнесённое слово, содержащее загаданную букву, его бьют током. Он может делать предположения, какая буква загадана, однако за каждую неправильную гипотезу его также будут бить током. Шоу заканчивается, когда Валентин угадывает букву правильно.

У Валентина не идеальная память, поэтому он мог угадать букву существенно позже, чем она однозначно определялась, то есть получить несколько лишних ударов током. Лишние удары током — это те удары, которые он получил после того момента, как загаданную букву можно было однозначно определить. Вам требуется найти количество таких лишних ударов.

Входные данные

В первой строке находится единственное целое число n ($1 \leq n \leq 10^5$) — количество действий Валентина.

В следующих n строках записаны его действия, по одному в строке. Каждое действие может принадлежать одному из трёх типов:

1. Слово, за которое его не ударили током. Такое действие описывается строкой «`. w`» (без кавычек), где «`.`» — точка (ASCII-код 46), а w — сказанное Валентином слово.
2. Слово, за которое его ударили током. Такое действие описывается строкой «`! w`» (без кавычек), где «`!`» — восклицательный знак (ASCII-код 33), а w — сказанное Валентином слово.
3. Гипотеза о загаданной букве. Такое действие описывается строкой «`? s`» (без кавычек), где «`?`» — вопросительный знак (ASCII-код 63), а s — гипотеза, маленькая буква латинского алфавита.

Все сказанные слова состоят из строчных букв латинского алфавита, суммарная длина всех слов не превосходит 10^5.

Гарантируется, что последним действием Валентин отгадывает загаданную букву. Также гарантируется, что ранее он её не отгадывал. Помимо того гарантируется, что в каждом слове, за которое его били током, такая буква присутствует, а в словах, за которые его не били — отсутствует.

Выходные данные

Выведите единственное целое число — количество ударов тока, которых Валентин мог избежать, если бы назвал загаданную букву сразу после того, как она однозначно определилась.

Примеры

входные данные

```
5
! abc
. ad
. b
! cd
? c
```

выходные данные

```
1
```

входные данные

```
8
! hello
! codeforces
? c
. o
? d
? h
. l
? e
```

выходные данные

```
2
```

входные данные

```
7
! ababahalamaha
? a
? b
? a
? b
? a
? h
```

выходные данные

```
0
```

Примечание

Разберём тестовые примеры.

В первом тесте после первого слова становится понятно, что загадана одна из букв a, b, c. Из второго слова мы узнаём, что загадана не буква a. Валентин произносит слово b, и не получает удар током. В этот момент мы понимаем, что загадана буква c, однако Валентин говорит слово cd, тем самым получает один лишний удар током.

Во втором тесте после первых двух ударов мы понимаем, что загадана либо буква e, либо буква o. Валентин начинает перебирать буквы из этих двух слов, и после второй его попытки угадать букву мы понимаем, что загадана буква e. Но Валентин делает ещё 3 попытки, прежде чем добирается до этой буквы.

В третьем тесте загаданная буква однозначно определяется только когда Валентин её угадал, поэтому он не получил лишних ударов током.

906C - Шокеры

Из последнего действия можно понять, какая буква была загадана. Пусть, без потери общности, это будет c. Для всех остальных 25 букв ответы на некоторые действия противоречат предположению, что эта буква выбрана. Более того, для каждой буквы d, не совпадающей с c мы можем найти самое первое такое действие с номером A_d (для каждого действия мы легко можем проверить, не противоречит ли оно утверждению «d загадана» за линейное время). Таким образом, ответ есть количество ударов током после действия с номером $max(A_d)$ по всем $d \neq c$.

D. Пересаживание студентов

ограничение по времени на тест

2 секунды

ограничение по памяти на тест

256 мегабайт

ввод

стандартный ввод

вывод

стандартный вывод

Студенты пришли в аудиторию писать контрольную и расселись некоторым образом. Преподаватель подумал: «Наверняка они сели так, чтобы им было удобней списывать друг у друга. Надо их пересадить так, чтобы ранее сидевшие рядом студенты рядом не сидели.»

Можно представить себе аудиторию как прямоугольное поле n на m, в каждой клетке сидит по студенту. Два студента сидят рядом друг с другом, если соответствующие им клетки смежны по стороне.

Пронумеруем студентов от 1 до $n \cdot m$ по строчкам. Т.е. студент, изначально сидящий в клетке в i-й строчке и j-м столбце будет иметь номер $(i - 1) \cdot m + j$. Вам требуется найти таблицу, в которой будет n строчек и m столбцов, в которой каждое число от 1 до $n \cdot m$ будет записано ровно один раз, а также соседние числа в изначальной таблице не должны быть соседними в получившейся, либо сказать, что такой не существует.

Входные данные

В единственной строке находятся два целых числа n и m ($1 \le n, m \le 10^5$; $n \cdot m \le 10^5$) — количество строк и столбцов в искомой таблице соответственно.

Выходные данные

Если искомой таблицы не существует, в единственной строке выведите «NO» (без кавычек).

Если же она существует, то в первой строке выведите «YES» (без кавычек), а в следующих n строках выведите по m целых чисел — искомую таблицу.

Примеры

входные данные

```
2 4
```

выходные данные

```
YES
5 4 7 2
3 6 1 8
```

входные данные

```
2 1
```

выходные данные

```
NO
```

Примечание

Разберём первый тест. Изначальная таблица выглядит так:

```
1 2 3 4
5 6 7 8
```

Легко видеть, что никакие две пары студентов не являются соседями одновременно в обеих таблицах.

Во втором тесте существует всего два варианта рассадить студентов, и в любом случае пара 1 2 будет сидеть рядом.

906B - Пересаживание студентов

Задача имеет множество решений, в том числе случайные. Разберём одно детерминированное решение.

Симметричные таблицы разбираются одинаково, поэтому будем полагать, что $n \le m$.

Есть несколько частных случаев:

1. $n = 1, m = 1$. В этом случае ответ, очевидно, существует.
2. $n = 1, m = 2$. В этом случае ответа, очевидно, не существует.
3. $n = 1, m = 3$. В этом случае ответа не существует, так как рядом со студентом 2 всегда будет сидеть один из его бывших соседей.
4. $n = 2, m = 2$, то ответа также нет, так как рядом со студентом 1 может находиться только студент 4, но рядом с любым студентом всегда должны сидеть по меньшей мере двое.
5. $n = 2, m = 3$, Докажем, что в этом случае ответа также нет. Так как у 5 и 2 в изначальной таблице 3 соседа, то после пересаживания они могут сидеть только по углам (так как у каждого из них осталось по 2 незадействованных соседа). Пусть 5 сидит в нижнем левом, а 2 — в верхнем правом углах (если они сидят в углах в одном ряду, то на клеточку между ними нельзя ничего поставить). На нижнем центральном месте может сидеть только 1 или 3 (из-за 5 в углу). Если там сидит 1, то 4 некуда сесть, а если 3, то 6 не может найти себе место. Получается, что в любом случае рассадить студентов невозможно.
6. $n = 1, m = 4$. В таком случае ответ существует, и он `2 4 1 3`.

7. $n = 1; 5 \le m$. В таком случае на первую половину строчки посадим всех студентов с нечётными номерами в порядке возрастания, а на вторую - студентов с чётными. Например, при $m = 7$ получится `1 3 5 7 2 4 6`.
8. $n = m = 3$ Студентов можно рассадить, например, так:

```
6 1 8

7 5 3

2 9 4
```

9. Если $2 \le n; 4 \le m$, то предлагается каждую чётную строчку сдвинуть циклически на два вправо, а затем каждый чётный столбец циклически сдвинуть на 1 вверх. Если два студента ранее были соседями по столбцу, то после операций они будут сидеть в двух разных столбцах на расстоянии 2. Если же они были соседями по строчке, то после пересаживаний будут сидеть в соседней строчке и соседнем столбце. Таким образом, для $2 \le n, 4 \le m$ мы построили пример.

E. Вечеринка

ограничение по времени на тест
1 секунда
ограничение по памяти на тест
256 мегабайт
ввод
стандартный ввод
вывод
стандартный вывод

Арсений любит собирать людей на вечеринки. Однако к нему помимо друзей приходят также друзья друзей, друзья друзей друзей и так далее. Таким образом даже Арсений может не знать всех пришедших к нему гостей. Он решил исправить это недоразумение при помощи следующего процесса:

На каждом шаге выбирается один человек A, который в свою очередь попарно знакомит всех своих знакомых. После этого все знакомые A попарно становятся друзьями. Это продолжается до тех пор, пока все гости не будут попарно знакомы.

Арсению очень хочется провести поменьше времени за этим процессом, поэтому он хочет его завершить за наименьшее число шагов. Помогите ему в этом.

Входные данные

В первой строке находятся два целых числа n и m ($1 \le n \le 22$; $0 \le m \le \frac{n \cdot (n-1)}{2}$) — количество человек на вечеринке, включая Арсения, и количество пар знакомых людей.

В следующих m строках записаны по два целых числа u и v ($1 \le u, v \le n$; $u \ne v$). Каждая такая пара обозначает, что люди с номерами u и v знакомы. Гарантируется, что каждая пара знакомств описывается не более одного раза, а также гарантируется, что данный граф знакомств связен.

Выходные данные

В первую строку запишите наименьшее количество ходов, за которое можно попарно познакомить всех людей. Во второй строке для каждого ходы выведите номер гостя, который будет знакомить своих друзей на этом шаге. Если решений несколько, выведите любое.

Примеры

входные данные

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

выходные данные

```
2
2 3
```

входные данные

```
4 4
1 2
1 3
1 4
3 4
```

выходные данные

```
1
1
```

Примечание

В первом тестовом примере нет человека, который знал бы всех, поэтому для выполнения задачи требуется по меньшей мере два хода. После того, как второй перезнакомит всех своих друзей останутся незнакомыми только пары $(4, 1)$ и $(4, 2)$. Познакомить их может либо 3, либо 5.

Во втором тестовом примере гость номер 1 знает всех, поэтому за один ход может всех попарно познакомить.

[906C - Вечеринка]

Сформулируем и докажем несколько фактов.

1. От изменения порядка операций результат не меняется. Рассмотрим две вершины. Если они не соединены ребром, то независимо от порядка их вызова в итоге получится, что знакомые каждой вершины образуют клику.

Если же две наши вершины смежны, то в итоге независимо от порядка получится клика из двух вершин, а также всех их соседей.

2. Если граф является деревом, то достаточно взять в качестве ответа все его вершины, не являющиеся листьями. Действительно, если рассмотреть две любые вершины дерева, то получится, что все вершины на пути между ними взяты в ответ. Каждая такая уменьшает расстояние между двумя нашими вершинами на 1, значит в итоге расстояние между ними равно 1.

3. Давайте в исходном графе выделим остовное дерево, в котором наибольшее количество листьев. Утверждается, что в качестве ответа можно взять все не висячие вершины этого дерева.

Из пункта 2 очевидно, что после всех операций с таким множеством граф станет полным. Покажем, что такое количество вершин является минимальным.

Пусть мы выделили некоторое множество вершин, которое является ответом. Тогда граф, ограниченный на выделенное множество, обязан быть связным. В противном случае между компонентами связности никогда не появится рёбер, и граф не станет полным. Также, каждая из вершин, не вошедших в выбранное множество, должна среди своих соседей иметь хотя бы одну выбранную. Иначе к ней никогда не добавится новых рёбер. Теперь давайте выделим в выбранном множестве остовное дерево(это возможно, так как наше множество связно). Получается, что наш ответ можно представить как остовное дерево, в котором все выбранные вершины не являются листами(иначе их можно было бы выбросить из выбранного множества — противоречие с минимальностью). Так как количество выбранных вершин минимально, то количество листьев — максимально, что и требовалось доказать.

4. Реализация. Требовалось реализовать алгоритм, который работал бы за $2^n \cdot n$, либо алгоритм с большей асимптотикой и не асимптотическими оптимизациями. Предлагается делать следующее. Будем перебирать маску-ответ, и дополнительно при помощи динамики вперёд поддерживать, верно ли, что текущие выбранные вершины связаны. Если текущие вершины не связаны, то можно ничего не делать и переходить к следующей маске, а если они образуют связное множество, то за $O(n)$ при помощи битовых операций можно найти все вершины, смежные хотя бы с одной из вершин множества, и затем для каждой маски, которая получается из текущей путём добавления одной вершины из найденного множества сказать, что она является связной. Также надо проверить, что полученное множество соседей дополняет выбранное множество до множества всех вершин. Если это свойство выполнено, то наше множество является кандидатом на ответ. В конце необходимо среди всех кандидатов выбрать того, в котором количество вершин минимально.

F. Башня Мощи

ограничение по времени на тест

4.5 секунд

ограничение по памяти на тест

256 мегабайт

ввод

стандартный ввод

вывод

стандартный вывод

Жрецы культа Кетцалькоатля хотят построить величественную башню, символизирующую мощь их бога. Башня обычно строится из заряженных мощью камней. Она строится с помощью редкой магии, которая заставляет верхнюю часть башни левитировать над землёй, в то время как очередной камень пристраивается снизу башни. Если верхняя часть башни, построенная из $k - 1$камней, обладает мощью p и мы хотим добавить к ней камень, заряженный мощью w_k, то значение мощи новой башни будет равно $\{w_k\}^p$.

Камни добавляются, начиная с последнего и заканчивая нижним, поэтому для последовательности $w_1, ..., w_m$ значение мощи будет равно

$$w_1^{\left(w_2^{\left(w_3^{\cdot^{\cdot^{\cdot^{w_m}}}}\right)}\right)}$$

После того, как башня построена, её мощь может быть настолько колоссальной, что человеческий разум будет не в силах осознать её. Но жрецы всё ещё хотят иметь какую-то информацию о мощи башни, поэтому они хотят узнать значение приведённой мощи, которое равно истинному значению взятому по модулю m. У жрецов есть n камней пронумерованных от 1 до n. Они просят вас посчитать, каким значением приведённой мощи будет обладать башня, если её построить из камней с номерами $l, l + 1, ..., r$.

Входные данные

Первая строка ввода содержит два целых числа n $(1 \le n \le 10^5)$ и m $(1 \le m \le 10^9)$.

Вторая строка ввода содержит n целых чисел w_k $(1 \le w_k \le 10^9)$, обозначающих значение мощи камней, имеющихся у жрецов.

Третья строка содержит единственное число q $(1 \le q \le 10^5)$, равное числу вопросов, которые вам зададут жрецы.

k^{th} из следующих q строк содержит два целых числа l_k и r_k $(1 \le l_k \le r_k \le n)$.

Выходные данные

Выведите q целых чисел. k-е из них должно быть равно величине приведённой мощи, которая будет у башни, построенной из камней с номерами l_k, l_k+1, ..., r_k.

Пример

входные данные

```
6 1000000000
1 2 2 3 3 3
8
1 1
1 6
2 2
2 3
2 4
4 4
4 5
4 6
```

выходные данные

```
1
1
2
4
256
3
27
597484987
```

Примечание

$3^{27} = 7625597484987$

906D - Башня Мощи

Научимся вычислять выражение $a_1^{a_2^{\cdot^{\cdot^{\cdot^{a_n}}}}} \mod m$. Будем считать, что мы хотим вычислить $n^x \mod m$, где n и m могут не быть взаимно простыми, а x – некоторое очень большое число, которое иначе как по какому-то модулю мы считать не можем.

Мы умеем решать эту задачу с помощью теоремы Эйлера для взаимно простых n и m. Логичным будет свести нашу задачу к такому случаю. Обратим внимание, что $an \mod am = a(n \mod m)$. Действительно, если $n = d \cdot m + r$, $|r| < m$, то $an = d \cdot am + ar$, $|ar| < |am|$. Пусть p_1, ..., p_t – общие простые делители n и m, $a = p_1^{k_1} \dots p_t^{k_t}$ – число, в которое эти простые делители взяты с теми же степенями, что и в m, а k – наименьшее число такое что $n^k = 0 \mod a$. Тогда имеет место цепочка

$$n^x \mod m = \left(\frac{n^k}{a}\right) a n^{x-k} \mod a \left(\frac{m}{a}\right) = n^k \left[n^{x-k} \mod \left(\frac{m}{a}\right)\right] \mod m$$

При этом n и m / a – взаимно простые и можно перейти к вычислению степени по модулю функции Эйлера от модуля. Кроме того, $k \leq \log_2 m$, поэтому случай $x < k$ может быть рассмотрен за $O(\log m)$.

Кроме того, данные рассуждения могут быть доведены до следующей леммы:
пусть $x \geq \log_2 m$, тогда

$$n^x \mod m = n^{\varphi(m) + x \bmod \varphi(m)} \mod m$$

Где $\varphi(m)$ – функция Эйлера от m. Для того, чтобы получить решение исходной задачи, обратим внимание на то, что функция Эйлера за $O(\log n)$ шагов превратится в 1, что позволяет нам отвечать на запросы за $O(\log^2 MAX)$.