Алгоритмы и модели вычислений, группы $774 - 775$, задание 5

- Кормен–Лейзерсон–Ривест–Штайн, глава 31, параграфы $1 - 7$. ***ОБЯЗАТЕЛЬНО ЧИТАТЬ ВСЁ!!!***.
- Кормен–Лейзерсон–Ривест–Штайн, глава 31, параграф 8. Несмотря на то, что этот параграф в Кормене помечен звездочкой, он может помочь для решения задачи из задания.
- Конспект 3-ей лекций Мусатова по курсу теории сложностей у ФИВТов:
  http://ru.discrete-mathematics.org/fall2017/3/complexity/compl-book.pdf.
  Здесь всё по-прежнему. Лекцию 3 теперь уже нужно прочитать полностью.
- Хорошая книжка по теории сложностей на английском - http://theory.cs.princeton.edu/complexity/book.pdf. Читать то же, что и раньше: параграфы 2.2, 2.3, 2.4. Параграфы 2.5 и 2.7 тоже полезно почитать. Не бойтесь математического английского! Он простой, серьёзно.
- Также рекомендуем книгу блестящих математиков Гача и Ловаса по вычислительной сложности:
  http://www.cs.elte.hu/∼lovasz/complexity.pdf. Здесь вам стоит разобраться с параграфами 4.3.1 - полиномиальность алгоритма Евклида и быстрое вычисление степени по модулю, 4.3.2 - полиномиальность метода Гаусса, 6.4.2 - ещё раз о сертификатах Пратта, 6.4.3 - здесь рассказывается про задачу о линейных неравенствах и её принадлежность классу $\mathcal{NP}$, 6.6.2 - здесь про $\mathcal{NP}$-полноты двух языков - систем линейных диофантовых неравенств и языка $SUBSET\ SUM$.

Задание 5

Задачи достаточно простые по сравнению с прошлой домашкой, в основном технические.

1. При помощи теоремы Эйлера (и, возможно, КТО) вычислите:
   (i) $12^{14^{18^3}} \pmod{19}$
   (ii) $14^{11^{17^2}} \pmod{25}$
   (iii) $7^{14^{20^9}} \pmod{24}$
2. Предложите полиномиальный алгоритм нахождения периода десятичной дроби $\frac{n}{m}$, докажите его корректность и оцените асимптотику.
3. Найдите все возможные (в зависимости от m) значения выражения $\left(\sum\limits_{i=1}^{m} i\right) \pmod{m}$
4. Придумайте эффективный алгоритм разрешения языка, каждое слово в котором представляет из себя набор натуральных пар (a_i, d_i) таких, что арифметические прогрессии, задаваемые ими как первыми членами и разностями, имеют непустое пересечение.
5. Решите систему:
$$\begin{cases} x \pmod{36} = 24 \\ x \pmod{54} = 45 \\ x \pmod{107} = 53 \end{cases}$$
6. Как известно, при шифровании информации с помощью алгоритма RSA требуется, чтобы изначально число было взаимно просто с модулем. Что будет, если нарушить это правило?
7. Как вам известно, RSA может быть так же использован в качестве алгоритма ЭЦП (электронной цифровой подписи). Рассмотрим некоторую «игрушечную» схему атаки на этот алгоритм. Пусть (e, n) - некоторый открытый ключ, известный вам как злоумышленнику, а (d, n) - закрытый, известный лишь некоему господину К, имеющему большое влияние и увлекающимся покупкой всяких антикварных безделушек. Вы хотите заставить господина К подписать некоторое сообщение x, которое является договором о переводе средств на ваш счет. Вы начинаете перебирать произвольные $0 < r < n$ и вычислять число $l = r^e \cdot x \pmod{n}$, причем делаете это до тех пор, пока сообщение l не станет похожим на некоторый чек о покупке безделушки. После чего вы отправляете l господину К, и он с радостью его подписывает, то есть вычисляет $w = l^d \pmod{n}$ и возвращает w вам.

(i) Предложите алгорим, с помощью которого по w можно подделать подпись сообщения x

(ii) Всегда ли вы сможете сгенерировать сообщение о покупке антиквариата из произвольного сообщения x. Формально, правда ли, что $\forall (e,n),(d,n)$-correct RSA pair $\forall x, l\ \exists r : r^e \pmod n = l$

(iii) Оцените время подбора r и поймите, почему эта атака неправдоподобна :-)

8. Следующая задача показывает использование вероятностного подхода для построения алгоритмов, проверяющих простоту чисел:

(i) Пусть $gcd(a,N) = 1, a^{N-1} \neq 1 \pmod N$. Докажите, что тогда по крайней мере для половины чисел из промежутка $1 \leq b < N$ выполнено $b^{N-1} \neq 1 \pmod N$.

Результаты этого простого, но важного упражнения позволяют строить быстрые **вероятностные** алгоритмы, проверяющие простоту чисел.

Речь идет о процедурах, которые, получив на вход n-битовую двоичную запись числа, могут быстро[1] проверять, является ли рассматриваемое число простым или составным. При этом вероятностным алгоритмам разрешается по ходу вычислений совершать переходы в зависимости от результатов бросания монетки. Тут, конечно, нужно уточнить, как следует понимать время работы вероятностного алгоритма, поскольку каждому исходу бросания монеток отвечает свое (возможно, очень длинное) вычисление. В случае алгоритмов проверки простоты речь идет о построении полиномиальных процедур типа "Лас-Вегас" (это термин), которые не ошибаются, если число составное, а если число простое, то алгоритм может выдать неправильный ответ, но вероятность этого события меньше, чем фиксированная константа, скажем $\frac{3}{4}$. Поэтому, если независимо повторить такой Лас-Вегас алгоритм k раз и во всех случаях он выдаст ответ "простое", то с вероятностью $1-(\frac{3}{4})^k$ число действительно будет простым. При таком подходе вероятность $1-0.75^{1000}$ нужно рассматривать как практическую достоверность, и именно такими методами были на сегодняшний день построены самые большие доказуемо простые числа.

Рассмотрим один подобный алгоритм ТЕСТ ФЕРМА.

Algorithm 1: ТЕСТ ФЕРМА(N)

Input: натуральное число N
Output: ДА, если N — простое, НЕТ в противном случае.

```
1 Случайно выбираем положительное число 1 < a < N.
2 if gcd(a, N) > 1 then
3     return НЕТ
4 else
5     if a^{N-1} = 1 (mod N) then
6         return ДА
7     else
8         return НЕТ
```

Описанный алгоритм — это "почти" полиномиальный вероятностный алгоритм проверки простоты

(ii) Покажите, что "ТЕСТ ФЕРМА" может быть реализован за полиномиальное по входу число операций.

(iii) Пусть известно, что составное число N не является числом Кармайкла[2], т. е. для некоторого натурального a, взаимно-простого с N, выполнено $a^{N-1} \neq 1 \pmod N$, тогда ТЕСТ ФЕРМА выдает правильный ответ с вероятностью[3] $\geq \frac{1}{2}$.

Необязательные задачи

(можно сдавать до конца семестра)

1. Если использовать один и тот же модуль N в разных протоколах RSA (с разными e,d), то образуется уязвимость, так как пользователь протокола знает не только общеизвестную пару (N,e), но и d. Оказывается, зная одну пару (e,d) можно легко найти секретные ключи всех остальных пользователей протокола с тем же модулем (естественно, зная их открытые N, но не зная p,q). Предложите эффективный алгоритм (полиномиальный от длины описания задачи), который решает эту задачу, докажите его корректность и оцените асимптотику.

[1] За полиномиальное **по длине записи** — n —- время (число операций). Помните, что само число может быть порядка 2^n.

[2] Числа Кармайкла — это составные числа, для которых тест Ферма выполняется для всех чисел a, взаимно простых с модулем N. Встречаются они редко. Везде приводится первое число Кармайкла — 561, попробуйте найти второе. Известно, что чисел Кармайкла бесконечно много, но доказан этот факт был совсем недавно. Как с ними бороться и как построить корректный полиномиальный вероятностный алгоритм проверки простоты можно прочитать в книге Кормена.

[3] Вероятность понимается в простейшем смысле как отношение числа благоприятных исходов к общему числу исходов.