Algorytmy Algebry i Teorii Liczb

Wybrane Zagadnienia

„Zrobią sobie Państwo dużo wrogów ale i tak zachęcam”

Popełnione przez

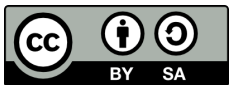
Chrząszczowy Pomp

Animowany Ponton

Kraków
Anno Domini 2023

Spis treści

Licencja



Rozdział 1

Wykłady

1.1 Algorytm Karatsuby

1.2 Algorytm Tooma-Cooka

Mamy do pomnożenia liczby A i B.

Zapisujemy $A = A_0 + A_1 \cdot K + A_2 \cdot K^2$ oraz $B = B_0 + B_1 \cdot K + B_2 \cdot K^2$ (gdzie $K = \frac{n}{3}$ co podejrzanie wygląda jak wielomiany drugiego stopnia.

Możemy zatem potraktować te liczby jako wielomiany i wymnożyć $C(X) = A(X) \cdot B(X)$, a następnie wyliczyć $C(K)$.

C jest wielomianem czwartego stopnia, więc jak wyliczymy wartości w pięciu punktach to będziemy w stanie wyliczyć z układu równań jego współczynniki.

Ewaluujemy zatem:

1. $C(-2) = A(-2) \cdot B(-2)$
2. $C(-1)$
3. $C(0) = A(0) \cdot B(0) = C_0$
4. $C(1) = A(1) \cdot B(1) = C_0 + C_1 + \cdots + C_4$
5. $C(2) = C_0 + 2C_1 + 4C_2 + 8C_3 + 16C_4$

Wartości $C(-2), \ldots, C(2)$ wyliczamy rekurencyjnie, a potem to już układ równań (nie przejmując się tym, że na ANach nam nie było wolno).

1.3 Algorytm Euklidesa

Rozdział 2

Rozwiązania Zadań

2.1 Zestaw 1

2.1.1 Zadanie 1

2.1.1.1 Wzór na $\phi(n)$

Zauważmy, że liczb mniejszych równych od n które sa kandydatami na bycie wzglednie pierwszymi z n jest n. Więc mamy:

$$n$$

Wśród tych n liczb n/p_1 dzieli sie przez p_1, wiec nie sa one wzglednie pierwsze z n. Zatem musimy pozbyć sie ich z naszego n:

$$n - \frac{n}{p_1}$$

Teraz wypadałłby pozbyć sie liczb podzielnych przez p_2. Jest ich dokładnie n/p_2. I wszystko byłoby good, gdyby nie to że w naszym n sa też liczby które dziela sie przez p_1p_2 i zostały one już odjete wcześniej (gdy odejmowaliśmy p_1). Zatem musimy je dodać z powrotem, by nie odejmować ich podwójnie. Wiec mamy:

$$n - \frac{n}{p_1} - \frac{n}{p_2} + \frac{n}{p_1p_2}$$

Dla p_3 chcemy odjać n/p_3 ale musimy dodać zarówno $n/(p_3p_2)$, jak i $n/(p_3p_1)$ (bo wcześniej je odjeliśmy, a teraz chcemy je dodać z p_3). Trzeba zwrócić uwage też na liczbe $p_1p_2p_3$, która została odjeta od nas raz (n/p_3) a dodana dwa razy $(n/(p_1p_2)$ i $n/(p_1p_2))$. Wiec trzeba ja dodać by wyjść na zero (chcemy wyjść na zero bo ta liczba została odjeta w pierwszym kroku przy pomocy p_1):

$$n(1 - \frac{1}{p_1} - \frac{1}{p_2} - \frac{1}{p_3} + \frac{1}{p_1p_2} + \frac{1}{p_1p_3} + \frac{1}{p_2p_3} - \frac{1}{p_1p_2p_3})$$

No i łatwo zauważyć patern. Idac kolejnymi liczbami odpowiednio je dodajemy i usuwamy. Stosujemy zasade właczen i wyłaczen i otrzymujemy wzór (można to zapisać sumami, ale wtedy brzydko to wyglada):

$$\phi(n) = n(1 - \frac{1}{p_1} - ... - \frac{1}{p_s} + \frac{1}{p_1p_2} + ... + \frac{1}{p_sp_{s-1}} + ... + (-1)^s\frac{1}{p_1...p_s})$$

Co jak sie przyjrzymy teleskopuje sie do:

$$\phi(n) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})...(1 - \frac{1}{p_s})$$

Co było do pokazania. ■

2.1.1.2 Multiplikatywnosc

Niech $n = ab$, gdzie a, b sa wzglednie pierwsze. Przyjmijmy poniższe oznaczenia:

$$a = p_1^{\alpha_1} p_2^{\alpha_2} ... p_s^{\alpha_s}$$

$$b = q_1^{\beta_1} q_2^{\beta_2} ... q_r^{\beta_r}$$

Jako ze $n = ab$ to:

$$n = p_1^{\alpha_1} ... p_s^{\alpha_s} q_1^{\beta_1} ... q_r^{\beta_r}$$

Z pierwszego popunktu wiemy też że:

$$\phi(a) = a(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})...(1 - \frac{1}{p_s})$$

$$\phi(b) = b(1 - \frac{1}{q_1})(1 - \frac{1}{q_2})...(1 - \frac{1}{q_r})$$

Jako że a i b sa wzglednie pierwsze, to $p_i \neq q_j$ dla każdego i, j.

$$\phi(n) = n(1 - \frac{1}{p_1})...(1 - \frac{1}{p_s})(1 - \frac{1}{q_1})...(1 - \frac{1}{q_r})$$

Ale $\phi(n)$ można zapisać również jako:

$$\phi(n) = a(1 - \frac{1}{p_1})...(1 - \frac{1}{p_s})b(1 - \frac{1}{q_1})...(1 - \frac{1}{q_r})$$

Po czym zauważamy że lewa strona iloczynu to $\phi(a)$, a prawa to $\phi(b)$. Zatem otrzymujemy:

$$\phi(n) = \phi(a)\phi(b)$$

Co było do pokazania. ■

2.1.2 Zadanie 2

2.1.2.1 Suma dzielników n

Niech $n = ab$, gdzie a, b sa wzglednie pierwsze. Przyjmijmy poniższe oznaczenia:

$$a = p_1^{\alpha_1} p_2^{\alpha_2} ... p_s^{\alpha_s}$$

$$b = q_1^{\beta_1} q_2^{\beta_2} ... q_r^{\beta_r}$$

Jako ze $n = ab$ to:

$$n = p_1^{\alpha_1} ... p_s^{\alpha_s} q_1^{\beta_1} ... q_r^{\beta_r}$$

Patrzac na rozkład liczby a zauważamy że jej dzielniki sa postaci: $p_1^{i_1} \cdot p_2^{i_2} \cdot ... \cdot p_s^{i_s}$, gdzie $0 \leq i_j \leq \alpha_j$ dla $j \leq s$. Zatem suma dzielnikow liczby a to suma wszystkich mozliwych kombinacji $i_1 ... i_s$. Zatem wzór na sume dzielnikow a ma postac:

$$\sigma(a) = (p_1^0 + p_1^1 + ... + p_1^{\alpha_1})(p_2^0 + p_1^1 + ... + p_2^{\alpha_2})...(p_s^0 + p_s^1 + ... + p_s^{\alpha_s})$$

Analogicznie dla b i n otrzymujemy:

$$\sigma(b) = (q_1^0 + q_1^1 + ... + q_1^{\beta_1})(q_2^0 + q_1^1 + ... + q_2^{\beta_2})...(q_r^0 + q_r^1 + ... + q_r^{\beta_r})$$

$$\sigma(n) = (p_1^0 + p_1^1 + ... + p_1^{\alpha_1})...(p_s^0 + p_s^1 + ... + p_s^{\alpha_s})(q_1^0 + q_1^1 + ... + q_1^{\beta_1})...(q_r^0 + q_r^1 + ... + q_r^{\beta_r})$$

Zauważany że prawa strona wyrażenia to $\sigma(b)$, a lewa to $\sigma(a)$, zatem otrzymujemy:

$$\sigma(n) = \sigma(a)\sigma(b)$$

Co było do pokazania. ■

2.1.2.2 Liczba dzielników n

Dowód na liczbe dzielników przeprowadzamy jak wyżej, zauważajac, że:

$$d(a) = (1 + \alpha_1)(1 + \alpha_2)...(1 + \alpha_s)$$

Czemu? Patrzymy na $\sigma(a)$. Czynnik zawierajacy p_i wyprodukuje wszystkie potegi p_i, ktorych jest $\alpha_i + 1$ Zatem liczby dzielników liczb b, n sa dane wzorem:

$$d(b) = (1 + \beta_1)(1 + \beta_2)...(1 + \beta_r)$$

$$d(n) = (1 + \alpha_1)...(1 + \alpha_s)(1 + \beta_1)...(1 + \beta_r)$$

Zatem otrzymujemy:

$$d(n) = d(a)d(b)$$

Co było do pokazania. ■

2.1.3 Zadanie 3

Gdyby nie była właśnie godzina 2 to bym zrobił to zadanie...

2.1.4 Zadanie 4

BSO $b \leq a$. Niech F_n to bedzie pierwsza liczba Fibonacciego wieksza lub równa a. Spójrzmy jak działa algorytm Euklidesa w zależnoci od tego jakiej wielkości jest b:

1. $b \leq F_{n-2}$
   Ponieważ $a\%b < b \leq F_{n-2}$ to w nastepnym wywołaniu funkcji otrzymujemy że wiekszy element z a, b jest mniejszy lub równy F_{n-2}.
2. $F_{n-2} < b \leq F_{n-1}$
   Ponieważ $a\%b < b \leq F_{n-1}$ to w nastepnym wywołaniu funkcji otrzymujemy że wiekszy element z a, b jest mniejszy lub równy F_{n-1}.
3. $F_{n-1} < b \leq F_n$
   Jako że zachodzi:

$$F_{n-1} < b$$

$$2F_{n-1} < 2b$$

$$F_n = F_{n-1} + F_{n-2} \leq 2F_{n-1} < 2b$$

$$b \leq a \leq F_n < 2b$$

To mamy $a\%b = a - b$. Z kolejnych nierównoci;

$$a \leq F_{n-1} + F_{n-2} = F_n$$

$$F_{n-2} \leq F_{n-1}$$

$$F_{n-1} < b$$

Otrzymujemy:

$$a\%b = a - b \leq F_{n-2}$$

Zatem w nastepnym wykonaniu funkcji dostaniemy przypadek pierwszy

Ponieważ w każdym kroku schodzimy o 1 lub o 2 w dół (a jak nie, to w nastepnym nadrabiamy 2 krokami) to funkcja wywoła sie co najwyżej n razy. Zauwazmy, że n-ta liczba Fibonacciego jest przedstawiana wzorem (Binet's formula):

$$F_n = \frac{\phi^n}{\sqrt{5}} - \frac{(-\phi)^{-n}}{\sqrt{5}}$$

Przy czym zauważmy że element który odejmujemy jest ≤ 1. Wiec możemy sobie przybliżyć (z góry):

$$F_n = \phi^n$$

$$log_\phi(F_n) = n$$

Zatem funkcja wywoła sie$log_\phi(F_n) + O(1)$ razy. Przy czym $F_n = a$.

2.1.5 Zadanie 5

2.1.5.1 Teza

Używamy rozszerzonego algorytmu Euklidesa by otrzymać liczby r i s takie że

$$g = gcd(m_1, m_2) = rm_1 + sm_2$$

Wtedy rozwiazaniem równania bedzie:

$$x = \frac{a_1 s m_2 + a_2 r m_1}{g}$$

2.1.5.2 Dowód

$$x = \frac{a_1 s m_2 + a_2 r m_1}{g} = \frac{a_1(g - r m_1) + a_2 r m_1}{g} = \frac{a_1 g + (-a_1 + a_2) r m_1}{g} =$$

$$= a_1 + \frac{(-a_1 + a_2) r m_1}{g}$$

$$a_1 + \frac{(-a_1 + a_2) r m_1}{g} \equiv a_1 mod(m_1)$$

Analogicznie

$$a_2 + \frac{(-a_2 + a_1) s m_2}{g} \equiv a_2 mod(m_2)$$

Zaprezentowawne rozwiazanie działa. ■

2.2 Zestaw 2 - Liczby pierwsze, grupy cykliczne

2.2.1 Zadanie 1

Treść:

Pokazać, że w grupie cyklicznej rzędu n jest dokładnie $\phi(n)$ generatorów.

Rozwiązanie:

Wiemy, że skoro mamy grupe cykliczną to istnieje g generator, a więc weźmy sobie owy generator. Teraz wszystkie elementy grupy są w postaci g^k, gdzie $1 \leq k \leq n$.

Oznaczmy sobie $s = \frac{n}{\gcd(n,k)}$ teraz g^{k^s} na pewno wynosi e (element neutralny) gdyż potęga przy g dzieli n, ale my chcemy aby n-krotne złożenie było najmniejszym który otrzymuje element neutralny a z tego wynika że $\gcd(n,k) = 1$, a z definicji $\phi(n)$ to jest liczba takich $k \leq n$, że $\gcd(n,k) = 1$, a więc $\phi(n)$ jest liczbą generatorów takiej grupy.

2.2.2 Zadanie 2

Treść:

W protokole Diffiego-Hellmana dana jest grupa $(G, \cdot)$, i pewien jej generator g, jedna strona losuje $a \in \mathbb{Z}$, druga $b \in \mathbb{Z}$, po czym przesyłają sobie odpowiednio g^a i g^b – uzgodnionym kluczem symetrycznym jest g^{ab}.

Pokaż, że wybór grupy $G = (\mathbb{Z}_n, +)$ jest bardzo złym pomysłem (czyli: znajdź szybki algorytm łamania tego protokołu).

Rozwiązanie:

Zauważmy że w grupie $(\mathbb{Z}_n, +)$ g^a oznacza to samo co ga. Oznaczmy sobie $z = ga$. Teraz możemy stworzyć równanie na kongruencji $ga = ga \bmod n = ga = z \bmod n$ znamy g oraz $ga = z$ a chcemy obliczyć a, a więc musimy obliczyć $(g)a = z \bmod n$. Co odpowiada dokładnie Chińskiemu twierdzeniu o resztach ze współczynnikiem liniowym ($a * x = b \bmod c$) co robimy dokładnie w A na satori. Jak robimy na satori:

mamy $ab = c \bmod n$, szukamy $\gcd(a, n)$. Jeżeli $\gcd(a, n) \neq 1$ to sprawdzamy czy $b \bmod \gcd(n, a) = 0$. Jeżeli tak nie jest to układ nie ma rozwiązania czyli takie a NIE MOŻE istnieć co oznacza że g nie był generatorem. W przeciwnym wypadku wydzielam a, b, n przez owe $\gcd(n, a)$ wiemy że nie jest to dzielenie modulo tylko normalne na liczbach bo każdy element jest przez nie podzielny wiec jest ok. Po takiej operacji wiemy że $\gcd(n, a) = 1$ więc możemy zastosować rozszerzony algorytm eulidesa do znalezienia takich u, v, że $au + vn = 1$ przekształcając równanie otrzymujemy, że $au = 1 - vn$ wymnóżmy teraz nasze równanie modularne razy u i otrzymujemy $uax = ub \bmod n$ podkładając za $au = 1 - vn$ otrzymujemy $(1 - vn)x = ub \bmod n$ wiemy że $n|vn$ czyli otrzymujemy $x = ub \bmod n$ a owy szukany x jest szukaną przez nas potęga w owej grupie. Algorytm jest wielomianowy od rozmiaru wejścia czyli jest raczej szybki.

2.2.3 Zadanie 3

Treść:

Pokazać, że $Z_3[i]^* = \{\ a + bi\ :\ a,\ b \in \mathbb{Z}_3,\ (a,\ b) \neq (0,\ 0)\ \}$, gdzie i^2 = -1, z mnożeniem ńaturalnym", jest grupą (i to przemienną).

Rozwiązanie:

Element neutralny: $e = (1, 0)$: $(a + bi)(1 + 0i) = a + bi$

Mnożenie naturalne jest przemienne wiec jest ona przemienna.

A resztę udowodni za nas fakt że $(1 + 2i)$ jest generatorem (zamkniętość na złożenie oraz istnienie elementu odwrotnego)

Dowód $1 + 2i$ jako generatora (przepałowanie 8 mnożeń):

$(1 + 2i)(1 + 2i) = 1 + 4i - 4 = i = g^2$
$i(1 + 2i) = i - 2 = 1 + i = g^3$
$(1 + i)(1 + 2i) = 1 + 3i - 2 = 2 = g^4$
$2(1 + 2i) = 2 + 4i = 2 + i = g^5$
$(2 + i)(1 + 2i) = 2 + 5i - 2 = 2i = g^6$
$2i(1 + 2i) = 2i - 4 = 2 + 2i = g^7$
$(2 + 2i)(1 + 2i) = 2 + 6i - 4 = 1 = g^8$
$1(1 + 2i) = 1 + 2i = g$

Właśnie otrzymaliśmy $1 + 2i$ generuje wszystkie elementy grupy (zamkniętość na złożenie) i teraz jak weźmiemy dowolny element g^k, gdzie $1 \leq k \leq 8$ to wystarczy go wymnożyć przez element g^{8-k}, gdyż $g^k g^{8-k} = g^8 = 1$

2.2.4 Zadanie 4

Treść:

Udowodnij, że istnieje nieskończenie wiele liczb pierwszych postaci $4k + 3$.

Rozwiązanie:

Dowód nie wprost:

Hipoteza: Załóżmy że ilość p pierwszych w postaci $4k + 3$ jest skończona.

A więc weźmy sobie $a = p_1 p_2 ... p_n$, gdzie p_i jest w postaci $4k + 3$ i są to wszystkie takie liczby. Wiemy z Hipotezy w takim razie że a jest skończone. Wiemy jeszcze że liczby pierwsze są nieparzyste (z wyjątkiem 2) więc albo są w postaci $4k + 1$ albo $4k + 3$. W takim razie weźmy sobie na cel $4a - 1$. Jest ono nie podzielne przez żadną z liczb p_i w postaci $4k + 3$ gdż $4(p_1 p_2 ... p_n) - 1 = -1 \bmod p_i$ dla każdego p_i z naszego zbioru (gdyż lewa cześć jest podzielna przez każde p_i a z tego wynika że albo a jest pierwsze albo ma same czynniki pierwsze w postaci $4k+1$ co jest nie możliwe gdyż $a = 3 \bmod 4$ a biorąc iloczyn dowolnej ilosci liczb przystających

do 1 mod 4 otrzymamy liczbe przystającą do 1 mod 4 czyli a jest pierwsze oraz a nie należało do naszego zbioru liczb pierwszych w postaci $4k+3$, a jest w takiej postaci SPRZECZNOŚĆ gdyż wzieliśmy wszystkie takie a było ich skończenie wiele.

Czyli musi być nieskonczenie wiele liczb w postaci $4k+3$.

2.2.5 Zadanie 5

Treść:

Pokaż, że jeśli liczba pierwsza p dzieli n^2+1 dla pewnego n, to $p = 1 \bmod 4$. Wyprowadź z tego dowód, że istnieje nieskończenie wiele liczb pierwszych postaci $4k+1$.

Rozwiązanie:

Dowód pierwszej części:

Fakt: p i n muszą być względnie pierwsze.

Dowód tego faktu (nie wprost):

Zakładamy że nie są względnie pierwsze co oznacza że $p|n$ (gdyż p jest pierwsze czyli ma dokładnie 2 dzielniki siebie oraz 1 a 1 być nie może bo były by względnie pierwsze). A wiec $p|n$. Wynika z tego, że $p|n^2$ czyli $n^2 = 0 \bmod p$ co implikuje że $n^2+1 = 1 \bmod p$ a więc mamy sprzeczność gdyż $p \neq 1$ co dowodzi nam powyższy fakt.

Dowód nie wprost (zakładamy, że p jest w postaci $4k+3$ Z faktu że n i p są względnie pierwsze wiemy, że n^2 i p są względnie pierwsze. Przenosząc 1 na drugą strone mamy $n^2 = -1 \bmod p$ (gdyż z założenia $p|n^2+1$, wymnażając to równanie przez n^2 mamy $n^4 = 1 \bmod p$. Możemy teraz zauważyć że dla każdego k zachodzi $n^{4k} = 1 \bmod p$ (poprostu odpowiednio domnażając n^4)

Z małego twierdzenia fermata mamy również, że $n^{p-1} = 1 \bmod p$, więc z założenia mamy $p = 4k+3$ podstawiając mamy $n^{4k+3-1} = n^{4k+2} = n^{4k}n^2 = 1(-1) = -1 \bmod p$ SPRZECZNOŚĆ czyli p musi być w postaci $4k+1$ (bo jest albo w postaci $4k+1$ albo $4k+3$. Co dowodzi nam pierwszą część.

Dowód drugiej części (będzie nie wprost):

Załóżmy, że jest skończenie wiele liczb pierwszych w postaci $4k+1$. Weźmy sobie zbiór $A = \{p : p = 4k+1$, oraz p pierwsze$\}$ wiemy, że A jest skończone. Weźmy wieć sobie $z = \prod_{p \in A} p$. Wiemy że jest on skończony z założeń. Weźmy sobie teraz liczbę $c = z^2+1$. Z pierwszej części wiemy że liczbę w takiej postaci dzielą tylko liczbe pierwsze w postaci $4k+1$. Ale żadna z naszych liczb jej nie dzieli ($z = 0 \bmod p$, co daje $z^2 = 0 \bmod p$, czyli $z^2+1 = 1 \bmod p$ dla każdego $p \in A$). Co oznacza, że nasze z też jest pierwsze i jest w postaci $4k+1$ i nie należało ono do A. Mamy więc SPRZECZNOŚĆ. Co dowodzi postawionego twierdzenia.

2.3 Zestaw 3 - Liczby pierwsze, pierścienie

2.3.1 Zadanie 1

Treść:

Niech S będzie pierścieniem:

1. Czy zbiór dzielników zera w S musi być ideałem? Udowodnić lub podać kontrprzykład.
2. Czy zbiór nilpotentów S musi być ideałem? Udowodnić lub podać kontrprzykład.

Rozwiązanie:

1. Zbiór dzielników zera w S jest ideałem.

Nie jest. Przykład Pierścień $\mathbb{R} \times \mathbb{R}$ z mnożeniem po współżędnych.
Weźmy sobie element (1,0) oraz (0,1), po wymnożeniu otrzymamy (0,0) czyli nasze 0 w pierścieniu, natomiast gdy dodamy owe liczby po współżędnych otrzymamy (1,1) co nie może być dzielnikiem 0, gdyż jest elementem neutalnym względem mnożenia.

2. Zbiór nilpotentów jest ideałem.

Tak, pokażemy sobie to:
1. $0 \in I$

2. $a, b \in I => a + b \in I$ (zamkniętość na dodawanie)
Weźmy sobie takie r, s że $a^r = 0, b^s = 0$.
Pokażemy sobie że $(a + b)^{r+s} = 0$.
Rozpiszmy sobie $(a + b)^{r+S}$ ze wzoru dwumanowego (z czego dwumiany nie będa nas obchodziły wiec oznaczymy sobie je kolejno przez $c_0, ..., c_{r+s}$. A więć :
$(a+b)^{r+s} = \sum_{i=0}^{r+s} c_i \cdot a^i b^{r+s-i}$ zauważmy że dla $i \leq r$ wyraz $r+s-i \geq s$ co oznacza że $b^{r+s-i} = 0$ czyli czynniki dla $i \leq r$ zerują się. Nastomiast gdy $i > r$ czynnik $a^i = 0$ czyli czynniki również się wyzerują co oznacza że otrzymamy 0. Czyli $(a + b)^{r+s} = 0 => (a + b) \in I$.

3.I posiada właściowść wciągania (po wymnożeniu z każdym elementem z pierścienia element ten będzie w ideale).
Wiemy że dla każego $a \in I$ istnieje sobie n takie że $a^n = 0$, teraz wymnażając go z jakimkolwiek elementes s z pierścienia mamy element $a \cdot s$, a jak podniesiemy to do potęgi n to otrzymamy 0. $(ab)^n = a^n b^n = 0 \cdot b^n = 0$ Czyli jest on ideałem (my zajmujemy się pierścieniami przemiennymi wiec jest to ok).

2.3.2 Zadanie 2

Treść:

Pokaż, że pierścień S jest ciałem wtedy i tylko wtedy, kiedy jego jedynymi ideałami są (0) (ideał z jednym elementem 0) oraz cały S.

Rozwiązanie:

1) Pierścień S jest ciałem
2) Pierścień S ma tylko ideały (0) oraz cały S

$1 => 2$

(0) ma na pewno (trywialny ideał)
Weźmy sobie nie wprost że ma jakiś ideał różny od (0) oraz mniejszy od całego S. Wiemy że skoro S jest ciałem to dla każdego elementu ma element odwrotny. Wieny też że istnieje x w naszym ideale więc z faktu że mamy x^{-1} to znaczy że $x^{-1}x$ należy do ideału (z definicji) co oznacza że 1 należy do ideału. Skoro 1 należy do ideału to możemy przeiterować się po każdym elemencie z S i wymożyć go z 1, która musi być w tym ideale co oznacza że każdy element S będzie w ideale ale założyliśmy że owy ideał nie być całym S SPRZECZNOŚĆ

$2 => 1$

Do tego faktu musimy pokazać że każdy element ma element odwrotny względem mnożenia. Pokażemy sobie że dla dowolnego a znajdziemy dla niego element odwrotny.

Weźmy sobie zbiór w którym narazie jest tylko element a (nie będzie to jeszcze ideał ale nie ma problemu za chwile z niego zrobimy ideał). Teraz wiemy że jakiegoś elementu jeszcze nie osiągneliśmy. A więc weźmy sobie ideał w postaci dla każdego $i \in S$ dodajmy do naszego zbioru $a \cdot i$. Skoro wiemy że dla każdego elementu będzie on w ideale to z definicji jest to ideałem (spełnia własność wciągania do ideału) a to oznacza, że 1 jest w ideale ponieważ nasz ideał musi być całym S z założenia bo S ma tylko 2 ideały (albo siebie albo (0)), A jeżeli 1 jest w ideale oraz jedyne liczby w ideale to $i \cdot a$ to oznacza że jakiś element i po wymnożeniu z a dał 1 co oznacza że jest jego odwrotnością. Taką procedurę możemy powtórzyć dla każdego $a \in S$ co będzie oznaczało że każdy element ma element odwrotny. co należało pokazać

2.3.3 Zadanie 3

Treść:

Pokaż, że test Millera-Rabina działa również wtedy, kiedy testowana liczba n jest potęgą liczby pierwszej (tzn. odpowiada *złożona* z prawdopodobieństwem $> \frac{1}{2}$ bez potrzeby osobnego sprawdzania tego przypadku).

Rozwiązanie:
Korzystam z ćwiczenia 5. Wiemy że liczby Charmichaela są bez kwadratowe co znaczyczy że nie dzieli ich żaden kwadrat liczby natruralnej większej od 1. Co oznacza że liczba w postaci p^k, gdzie p pierwsza i $k > 1$ nie może być liczbą Charmichala gdyż $p^2|p^k$ dla $k \geq 2$. Wynika z tego że istnieje x taki że nie spełnia on testu fermata dla p^k. Oznaczmy sobie dla ułatwienia $n = p^k$. Weźmy sobie więc podgupę $G = \{b \in \mathbb{Z}_n : b^{n-1} = 1 \bmod n \} \subset \mathbb{Z}_n^*$. Wiemy że G jest podgrupą oraz istnieje x, który nie należy do G. Z Lagrange'a rząd podgrupy dzieli rząd grupy czyli oraz z wcześniej wiemy że G nie jest rozmiaru n co oznacza $|G| \leq \frac{n}{2}$. Czyli pradowpodobieństwo że wybierzemy x nie spełniającego testu fermata (którego wykonujemy w końcowym kroku testu millera-rabina) wynosi conajwyżej $\frac{1}{2}$

2.3.4 Zadanie 4

Treść:

Pokaż, że jeśli dla pewnego n liczby $6n+1, 12n+1 i 18n+1$ są pierwsze, to $m = (6n+1)(12n+1)(18n+1)$ jest liczbą Carmichaela

Rozwiązanie:

$m = (6n+1)(12n+1)(18n+1) = pqr$(oznaczmy sobie kolejne czynniki przez takie litery).$m = 1296n^3+396n^2+36n+1$ a wiec $p-1|m-1$ oraz $q-1|m-1$ oraz $r-1|m-1$ (jak przepałujemy to to dostajemy bo wyciągamy $6n, 12n, 18n$ kolejno przed nawias). A więc wiemy że dla każdego a względnie pierwszego z p, q, r, $a^{m-1} = 1 \bmod p$ oraz $a^{m-1} = 1 \bmod q$ oraz $a^{m-1} = 1 \bmod r$ z czego z własności kongruencji dostajemy, że $a^{m-1} = 1 \bmod pqr = a^{m-1} = 1 \bmod m$ co jest definicją liczby Charmichalea

2.3.5 Zadanie 5

Treść:

Pokaż, że liczba Carmichaela musi być bezkwadratowa (niepodzielna przez żaden kwadrat liczby naturalnej większej od 1).

Rozwiązanie:

Dowód nie wprost:
Zakładamy sobie że nie jest bez kwadratowa. Czyli n (liczba Charmichela) możemy zapisać jako n $= p^k \cdot l$, gdzie $\gcd(p, l) = 1$ oraz $k \geq 2$. Weźmy sobie $a = p^{k-1}$. Z małego twierdzenia fermata wiemy że $n|a^n - a$ z czego wynika że $p^k|a^n - a$ oraz z definicji a wiemy że $p^k|a^n$ (gdyż $a = p^{k-1}$ oraz $n \geq 2$ co oznacza że $n \cdot (k-1) \geq k$) z czego również mamy że $p^k|a$ co oznacza żę $p^k|p^{k-1}$ SPRZECZNOŚĆ gdyż $p^k > p^{k-1}$.

2.4 Zestaw 4 - Pierścienie, wielomiany, ciała skończone

2.4.1 Zadanie 1

Treść:

Pokaż, że jeśli pierścień jest euklidesowy, to jest pierścieniem ideałów głównych. Podaj przykład pierścienia, który nie jest pierścieniem ideałów głównych.

Rozwiązanie:

1. $I = (0)$ trywialne.
2. weźmy sobie dowolne $I \neq (0)$

Weźmy sobie takie $a \neq 0$ oraz $N(a)$ jest minimalne względem wszystkich $a \in S$.

Pokażemy że $I \subseteq (a)$.

Weźmy sobie dowolny $x \in S$. Z własności, że S jest euklidesowy wiemy, że $x = aq + r$ oraz $r = 0$ lub $v(r) < v(a)$. Wybraliśmy takie a, że $v(a)$ jest minimalne, więc $v(r) \geq v(a)$ co oznacza, że $r = 0$, co oznacza, że $x = aq$. Czyli $I \subseteq (a)$.

Przykład:

Weźmy sobie pierścień nad ciałem $\mathbb{F}_2[x]$ (pierścień wielomianów stopnia co najwyżej 2). Weźmy sobie ideał generowany przez elementy $(x, 2)$. Teraz aby generował ten ideał jakiś jeden element (nazwijmy do z) to $x|z$ oraz $2|z$. Co oznacza że z musi być w postaci $2k \cdot x$, a więc generuje nie generuje wielomianów stałych.

2.4.2 Zadanie 2

Treść:

Pokazać, że pierścień $\mathbb{Z}[i] = \{a + bi : a, b \in \mathbb{Z}\}$ (pierścień zespolonych liczb całkowitych, zwany też pierścieniem Gaussa) jest euklidesowy.

Rozwiązanie:

Weźmy sobie $v((a, b)) = a^2 + b^2$.

Teraz musimy pokazać zachowanie reszty.

Weźmy sobie $a = a_1 + a_2 i, b = b_1 + b_2 i$. Możemy teraz z własności liczb zespolonych obliczyć:

$$\frac{a}{b} = \frac{a_1 + a_2 i}{b_1 + b_2 i} = \frac{(a_1 + a_2 i)(b_1 - b_2 i)}{b_1^2 - b_2^2} = \frac{(a_1 b_1 - a_2 b_2) + (a_2 b_1 - b_2 a_1)i}{v(b)}$$

Z faktu dzielenia liczb całkowitych wiemy że znajdziemy takie q_1, q_2, r_1, r_2 że:

$$a_1 b_1 - a_2 b_2 = v(b) q_1 + r_1$$
$$a_2 b_1 - b_2 a_1 = v(b) q_2 + r_2$$

możemy dobrać q_1, q_2 tak aby r_1, r_2 spełniały nierówność

$$-\frac{1}{2} v(b) \leq r_1, r_2 \leq \frac{1}{2} v(b)$$

(jak nie spełnia to odpowiednio zwiększamy q_1, q_2 lub zmniejszamy w zależności od potrzeby.)

Oznaczmy sobie teraz $q = q_1 + q_2 i$ oraz $r = r_1 + r_2 i$. Następnie zaobserwujmy że możemy zapisać:

$$\frac{a}{b} = \frac{v(b) q + r}{v(b)}$$

co po uproszczeniu i skorzystaniu z definicji sprzężenia prezentuje się w formie:

$$a = bq + \frac{r}{\overline{b}}$$

Wiemy że $a \in \mathbb{Z}$ oraz $bq \in \mathbb{Z}$ co implikuje, że $\frac{r}{b} \in \mathbb{Z}$. Następnie korzystając z właściwości naszej funkcji v wiemy, że

$$v(\frac{r}{b}) = v(\frac{r}{b}) = v(b)^{-1}v(r)$$

A teraz możemy zapisać

$$r = r_1 + r_2 i => v(r) = r_1^2 = r_2^2$$

a z poprzeniej nierównośći:

$$-\frac{1}{2}v(b) \leq r_1, r_2 \leq \frac{1}{2}v(b)$$

otrzymujemy, że

$$v(r) \leq 2(\frac{1}{2}v(b))^2 = \frac{1}{2}v(b)^2$$

Podkłając to do równania

$$v(\frac{r}{b}) = v(\frac{r}{b}) = v(b)^{-1}v(r) \leq v(b)^{-1}\frac{1}{2}v(b)^2 = \frac{1}{2}v(b)$$

Wiec mamy już wszystko gdyż nasza reszte z dzielenia $\frac{a}{b} = bq + \frac{r}{b}$ wynosi $\frac{a}{b}$ co spełnia definicję.

2.4.3 Zadanie 3

Treść:

Charakterystyką ciała F nazywamy taką liczbę k, że $1 + 1 + ... + 1$ (k razy) $= 0$

(o ile taka istnieje, w przeciwnym wypadku charakterystyka wynosi 0). Pokazać, że charakterystyka ciała skończonego zawsze jest dodatnia i jest liczbą pierwszą.

Rozwiązanie:

1. Nie zerowość.

Nie wprost zakładamy, że nie otrzymujemy 0 a wykonaliśmy więcej niz ilość elementów w ciele dodań 1. Co oznacza, że jakiś element musiał się powtórzyć. Jeżeli jakiś element się powtórzył to z definicji suma między powtórzeniami wynosi 0, gdyż jest ono elementem neutralnym dodawania. Czyli 0 musiało wystąpić sprzeczność.

2. Charakterystaka jest liczbą pierwszą.

Załóżmy nie wprost, że charakterystyka jest liczbą złożoną równą n. Możemy z tego faktu rozbić $n = pq$ gdzie $p, q > 1$. W takim razie $(1 + 1 + ... + 1)(n$ razy$) = (1 + 1 + ... + 1)(q$ razy$) \cdot (1 + 1 + ... + 1)(p$ razy$)$ z czego wynika, że p lub q jest 0. Co oznacza że n nie jest najmniejsza taką liczbą która po dodaniu n razy 1 otrzymamy 0. SPRZECZNOŚĆ z definicji charakterystyki.

2.4.4 Zadanie 4

Treść:

Dany jest wielomian o współczynnikach całkowitych $W(X) = a_0 + a_1X + ... + a_nX^n$ oraz

liczba całkowita s. Podaj algorytm, który w czasie $O(n)$ rozstrzygnie, czy $W(s)$ jest liczbą dodatnią, ujemną, czy zerem. Zakładamy, że liczby s oraz $a_0, ..., a_n$ mieszczą się w słowie maszynowym i można wykonywać na nich operacje w $O(1)$ (można o tym myśleć tak, że w $C++$ wszystkie dane mieściłyby się w typie int). Operacje na większych liczbach nie są już stałe – liczba rzędu s^k będzie potrzebowała $\Omega(k)$ pamięci.

Rozwiązanie:

1. Jak $s = 0$ to zwracamy a_0.

2. Jak $s < 0$ to zmieniamy współczynniki wielomianów przy nieparzystych potęgach na przeciwne i odpalamy się z $s = |s|$ dalej.

3. Jeżeli $s = 1$ to sumujemy współczynniki. (to będzie miało $O(nlogn)$ jeżeli jest n jest nie ograniczony ale to w treści jest blef i n mieści się w słowie maszynyowym)

4. Zauważmy teraz że owy wielomian możemy czytać jako liczbe w systemie o podstawie s. Wystarczy nam teraz tylko usunąć przepełnienia (zrobić tak aby współczynniki przy każdej potędze wielomianu z wyjątkiem najwyższej potęgi były w przedziale $[0, s-1]$ co jest prostą operacją na liczbach i nie wyjdziemy po za słowo maszynowe do jakiejś stałej). Co kończy zadanie bo wystarczy nam sprawdzić czy wszystkie nowe współczynniki są zerami albo znak najbadziej znaczącego współczynnika.

Rozdział 3

Pytania Egzaminacyjne

3.1 Grupa A

3.1.1 Opisać rozszerzony algorytm Euklidesa znajdowania NWD

Dla danych $a, b > 0$, algorytm zwraca $d = \gcd(a, b)$ oraz takie, że $s, t \in \mathbb{Z}$, że $d = s \cdot a + t \cdot b$.

Algorytm Euklidesa

- 1. Jeśli $a < b$, zamień a i b.
- 2. Jeśli $b = 0$, zwróć $d = a$, oraz parę $(1, 0)$.
- 3. Podziel z resztą a przez b, otrzymując $a = q \cdot b + r$.
- 4. Wywołaj $\gcd(b, r)$, otrzymując d oraz parę (s, t), taką, że $s \cdot b + t \cdot r = d$
- 5. Zwróć d oraz parę $(t, s - t \cdot q)$

Dlaczego ten algorytm działa ?
To, że zwraca on poprawne d to chyba już każdy widział wiele razy (pokazuje się, że skoro $d \mid a$ oraz $d \mid b$ to $d \mid (a - b)$, jeżeli $a \geq b$, a to idzie odrazu z przystawania modulo d). Jedyne co musimy pokazać, to fakt, że zwraca on poprawne s, t, czyli takie że zachodzi $s \cdot a + t \cdot b = d$. A więc załóżmy sobie niezmiennik, że w każdym kroku algorytmu nasze s, t dla obecnych a, b jest poprawne.

Baza ($b = 0$):
Zwracamy, że $d = a$, oraz zwracamy $(1, 0)$, z czego wynika, że $s = 1, t = 0$ czyli $1 \cdot a + 0 \cdot b = a = d$. Czyli jest OK

Krok:
Mamy, a, b, oraz $a = q \cdot b + r$ oraz zwrócone z rekursji (na argumentach (b, r)) d, s, t takie, że $s \cdot b + t \cdot r = d$. Chcemy teraz zmienić nasze s, t tak aby warunek zachodził dla a, b. A więc chcemu powiedzieć że dla nas będzie to $x = t$, $y = s - t \cdot q$ (konstruujemy nasze nowe s, t). Sprawdźmy teraz czy dla naszego x, y zachodzi nasz warunek.

$$x \cdot a + y \cdot b = t \cdot a + (s - t \cdot q) \cdot b = t \cdot a + s \cdot b - t \cdot q \cdot b$$

Tutaj skorzystamy z faktu, że $a = q \cdot b + r$, z czego wynika, że $q \cdot b = a - r$.

$$t \cdot a + s \cdot b - t \cdot q \cdot b = t \cdot a + s \cdot b - t \cdot (a - r) = t \cdot a + s \cdot b - t \cdot a + t \cdot r = s \cdot b + t \cdot r = d = x \cdot a + y \cdot b$$

Czyli jak widać nasz niezmiennik jest zachowany czyli algorytm działa.

Złożność: Zauważmy, że każdym przynajmniej jedna z liczb a, b spadnie nam o 2 (casologia):

- 1. $b \leq a/2$ z czego wynika, że $a \mod b < b < a/2$ czyli do rekurencji przekazaliśmy argument conajmniej dwukrotnie mniejszy
- 2. $a > b > a/2$ wynika z tego, że $a - b < a/2$ (a odejmowanie w tym przypadku zachowuje się tak samo jak modulo, gdyż $a = 1 \cdot b + r$, z czego wynika, że $a - b = 1 \cdot b + r - b = r$)

Czyli mamy $\mathcal{O}(\log a)$ (założenie, że $a \geq b$) kroków rekursji co daje nam złożoność $\mathcal{O}(\log a \cdot M(a))$, gdzie $M(a)$ to złożonośc operacji w jednym kroku rekurencji (Według wykładu złożoność całkowita to $\mathcal{O}(\log^2 a)$ ale nie za bardzo wiem czemu jak w każdym kroku wykonujemy mnożenie,dzielenie i modulo co nie jest tanie chyba że da się go jakość zoopcić).

3.1.2 Opisać binarny algorytm Euklidesa znajdowania NWD

Pseudokod:
Binarny Algorytm Euklidesa:

- 1. Jeśli $a < b$, zamień a i b,
- 2. Jeśli $b = 0$, zwróć a,
- 3. Jeśli $2 \mid a$ i $2 \nmid b$, zwróć $gcd(a/2, b)$,
- 4. Jeśli $2 \nmid a$ i $2 \mid b$, zwróć $\gcd(a, b/2)$,
- 5. Jeśli $2 \mid a$ i $2 \mid b$, zwróć $2 \cdot \gcd(a/2, b/2)$,
- 6. Jeśli $2 \nmid a$ i $2 \nmid b$, zwróć $\gcd(b, a - b)$.

Jak możemy zauważyć algorytm ten jest bardzo podobny do normalnego algorytmu euklidesa lecz rozważa on wszystkie możliwe parzystości a oraz b w danym kroku algorytmu.

Dlaczego owy algorytm działa?

Tutaj musimy się mocno wycaseować. Zauważmy na początek, że przypadek 1 oraz 2 są dokładnie takie same jak w zwykłym euklidesie.

Następnie weźmy sobie przypadek 3 i 4 na cel. W tych przypadkach jedna z liczb jest podzielna przez 2 a druga nie jest. Wynika z tego, że w ich nwd na pewno nie jest podzielne przez 2, czyli możemy podzielić tą liczbę, która jest podzielna przez 2 i uruchomić się rekurencyjnie i otrzymamy poprawne gcd.

Następny przypadek jest przypadek 5 i wynika z niego, że $2 \mid a$ oraz $2 \mid b$, co oznacza, że w ich nwd możemy uzględnić czynnik 2 oraz uruchomić się na $a/2$ oraz $b/2$, gdyż wyciągamy ten czynnik przed funkcję gcd i w ten sposób otrzymujemy poprawne gcd.

Ostanim przypadkiem jest, $2 \nmid a$ oraz $2 \nmid b$. Ale w nim wykonujemy normalny krok z algorytmu euklidesa czyli $\gcd(a, b) = \gcd(b, b - a)$, który też oczywiście jest poprawny.

Złożność Algorytmu:
Zauważymy, że w każdym z przypadków od 3 do 5, a lub b spada conajmniej dwukrotnie. Jedynm problemem wydaje się przypadek 6. Ale okazuje się, że nie jest to duży problem, gdyż jeżeli $2 \nmid a$ oraz $2 \nmid b$ to $2 \mid b-a$, czyli w wywołaniu rekurencyjnym zajdzie już przypadek od 3 do 5. A więc w conajwyżej dwóch krokach jedna z liczb spadnie dwukrotnie. A więc otrzymujemy złożoność $\mathcal{O}(\log(a + b) \cdot M(a, b))$, gdzie $M(a, b)$ to koszt wykonania operacji podziel przez 2, wymnóż razy 2, sprawdź podzielność przez 2, oraz odejmij liczby a oraz b a każdą z tych operacji jesteśmy w stanie wykonać liniowo względem zapisu liczby czyli w czasie $\mathcal{O}(\log n)$, czyli całkowita złożność z podliczonymi operacjami wynosi $\mathcal{O}(\log^2(a + b)) = \mathcal{O}(\log^2(a))$, przy założeniu, że $a \geq b$.

3.1.3 Zapisać algorytmy szyfrowania RSA oraz El-Gamal i opisać, na czym polega trudność ich łamania

Szybkie przypomnienie jak szyfrujemy (przynajmniej w tym przypadku) mamy sobie klucz publiczny i klucz prywatny, klucz publiczny udostępniamy i mówimy jak chcesz do mnie coś wysłać to użyj mojego klucza publicznego (no i oczywiście tej samej metody co ja) a ja sobie to odszyfruje moim kluczem prywatnym i przeczytam twoją wiadomość (funkcję szyfrującą oznaczamy zazwyczaj $E(X)$, a deszyfrująca $D(x)$.

RSA Idea:

- Wybiersz sobie liczby pierwsze p, q, oraz oblicz $N = p \cdot q$, oraz zauważ, że $\phi(N) = (p - 1)(q - 1)$ (zamiast $\phi(n)$ można wszędzie użyć $\text{lcm}(p - 1, q - 1)$)
- Wybierz e względnie pierwsze z $\phi(N)$ oraz oblicz takie d, że $e \cdot d = 1 \mod \phi(N)$ (można tu użyć rozszerzonego algorytmu euklidesa)
- Niech $E(x) = x^e \mod N$ oraz $D(x) = x^d \mod N$, czyli nasz klucz publiczny to (N, e)

Dlaczego deszyfracja działa?
Rząd grupy multiplikatywnej modulo N wynosi $\phi(N)$, a więc dla każdego x względnie pierwszego z N zachodzi $(x^e)^d = x^{e \cdot d} = x^1$, gdyż $e \cdot d = 1 \mod \phi(N)$ czyli $e \cdot d$ dzieli rząd naszej grupy multiplikatywnej modulo N.

RSA możemy złamać, gdy rozłożymy N na czynniki. Czyli otrzymamy p, q. Obliczymy wtedy $\phi(N)$ oraz znajdziemy d (rozszerzonym euklidesem), dla którego zachodzi $e \cdot d = 1 \mod \phi(N)$. Obecnie nie umiemy dobrze rozkładać na czynniki dlatego nie umiemy też łatwo łamać RSA o ile ktoś dobrze dobrał liczby.

Idea El-Gamal:

- Wybierzmy sobie jakąś grupę G (Na przykład grupę multiplikatywną ciała skończonego $\mathbb{F}_q$) oraz element $g \in G$ (najlepiej generator).
- Wylosuj liczbę $x \in G$.

- Klucz publiczny to (g, g^x), a prywatny to (g, x)
- Szyfrowanie wygląda w następujący sposób, wylosuj liczbę y oraz oblicz g^y oraz g^{xy} oraz wyślij wiadomość P w postaci $(g^y, P \cdot g^{xy})$.

Znając x oraz g^y możemy wykonać operację $(g^y)^x = g^{xy}$, następnie znaleść odwrotność g^{xy} i odzyskać P, gdyż $P \cdot g^{xy} \cdot g^{-xy} = P$

Problem w odszyfrowaniu polega na tym, że znając g^x, g^y nie potrafimy obliczyć g^{xy}, czyli sprowadza się do to szyfrowanie Diffiego-Hellmana, co rozwiązuje się logarytmem dyskretnym, którego nie potrafimy obecnie szybko liczyć.

3.1.4 Opisać metodę klucza jednorazowego Vernama, trudność jej łamania i praktyczne zastosowanie

Idea klucza jednorazowego Vernama polega na tym, że mamy sobie osobę A oraz osobę B. Osoba A chce przesłać osobie B wiadomość x zapisaną bitowo oraz mają one między sobą bezpieczny kanał i normalny (być może niebezpieczny) kanał do przesyłania wiadomości. Teraz osoba A losuje sobie k o tej samej długości co wiadomość x oraz wykonuje operacje $w = x \oplus k$. Następnie osoba A przesyła bezpiecznym kanałem wartość k oraz normalnym kanałem wartość w. Aby osoba B mogła odczytać x wystarczy, że weźmie i wykona operację $w \oplus k$, gdyż $x = x \oplus k \oplus k$ i xor jest łączny i przemienny a $k \oplus k = 0$.

Dlaczego to było by super bezpieczne?
Okazuje się że losując nasze k to wygląda tak samo jakbyśmy osobno losowali jej każdy bit, a co za tym idzie to to że jeżeli wylosujemy 1 na jakimś miejscu to że mienimy ten sam bit na przeciwny w zapisie x. Czyli w pełni (no być może pseudolosowo) losowo pozmieniemy wszystkie bity x co za tym idzie, że miedzy kolenymi wysłaniami wiadomości nie ma żadnych relacji i jest on nie do złamania.

Jakie są jego przypadki użycia?
Żadne !!! Po co robić cokolwiek jeżeli mamy bezpieczny kanał to poprostu wyślimy x bezpiecznym kanałem i tyle, nie jest wtedy potrzebna żadna kryptografia.

3.1.5 Zdefiniować problemy PRIMES oraz FACTORING i podać ich umiejscowienie w klasach złożoności

Definicja problemu PRIMES:
Mając na wejściu liczbę p stwierdź czy p jest liczbą pierwszą (Odpowiedź TAK/NIE).

PRIMES $\in coNP$:
Zgadnij $d \in \{2, ..., p-1\}$, jeżeli $d \mid p$ odpowiedz NIE.

PRIMES $\in NP$:
Tw. $\mathbb{Z}_p^*$ ma rząd $p-1$ wtedy i tylko wtedy, gdy p jest pierwsze.
Zgadujemy g(generator $\mathbb{Z}_p^*$), oraz rozkład na czynniki pierwsze $p-1$, czyli $p_1^{\alpha_1} \cdot ... \cdot p_s^{\alpha_s} = p-1$.
Następnie sprawdzamy:

- Rekurencyjnie dla każdego zgadniętego p_i czy jest pierwsze.
- Czy $g^{p-1} = 1 \mod p$.

- Czy dla każdego p_i zachodzi, że $g^{\frac{p-1}{p_i}} \neq 1 \mod p$ (Zauważ że nie uwzględniamy potęg liczb pierwszych gdyż są to najwieksze dzielniki, w których brakuje dokładnie czynnika więc jeżeli dla jakiegoś mniejszego dzielnika by to zachodziło to dla jednego z tych też zajdzie, gdyż dzieli on jeden z naszych dzielników).

Jeżeli wszystkie te warunki są prawdziwe to odpowaiadamy TAK.

PRIMES $\in BPP$:
Dowód jest przez pokazanie algorytmu Millera-Rabina.

PRIMES $\in P$:
Dowód to pokazanie algorytmu AKS którego tutaj raczej nie trzeba będzie pokazać.

Definicja problemu FACTORING:
Na wejściu dana jest liczba n oraz liczba k. Stwierdź czy istnieje dzielnik n mniejszy lub równy k. Czyli formalnie $\exists_d : 2 \leq d \leq k \wedge d \mid n$.

FACTORING $\in NP$:
Zgadnij $d \in \{2, ..., k\}$ jeżeli $d|n$ odpowiedz TAK.

FACTORING $\in coNP$:
Zgadujemy rozkład na czynniki pierwsze liczby n, czyli mamy $p_1^{\alpha_1} \cdot ... \cdot p_s^{\alpha_s}$. Następnie sprawdzamy czy $p_1^{\alpha_1} \cdot ... \cdot p_s^{\alpha_s} = n$ oraz czy każdego p_i jest pierwsze (albo poprzez AKS albo odpalamy się na algorytmie w NP). Jeżeli dla każego p_i nasze sprawdzenie odpowiedziało tak to znajdujemy najmniejsze p_i w naszym rozkładzie i zwracamy TAK, jeżeli najmniejsze p_i jest mniejsze lub równe k w przeciwnym odpowiedz NIE.
Więcej o tym problemie nie potrafimy narazie powiedzieć.

3.1.6 Podać efektywną metodę znalezienia liczby pierwszej o zadanej liczbie bitów

Mamy podana liczbę k oraz chcemy znaleść liczbę pierwszą p, która ma k bitów, z czego wynika, że $p \in [2^k, 2^{k+1} - 1]$.

Pierwszym faktem jaki zauważymy jest gęstość liczb pierwszych:
W przedziale od 1 do n jest asymptotycznie $\mathcal{O}(\frac{n}{\log n})$ liczb pierwszych.
Wynika z tego, że liczby pierwsze sa upakowane dosyć gęsto. Wiemy, że w przedziale od 1 do $2^{k+1} - 1$ jest rzędu $c \cdot \frac{2^{k+1}-1}{k+1}$ liczb pierwszych oraz w przedziale od 1 do $2^k - 1$ jest rzędu $c \cdot \frac{2^k-1}{k}$ wynika z tego, że w przedziale $[2^k, 2^{k+1} - 1]$ jest $c \cdot \frac{2^{k+1}-1}{k} - c \cdot \frac{2^k-1}{k}$ około tyle liczb pierwszych. Z czego wynika że mamy tam dalej $\mathcal{O}(\frac{n}{\log n})$ liczb pierwszych (troche machane).

Skoro są one upakowane dosyć gęsto to wykonajmy następującą procedurę.

- Wylosuj p z przedziału $[2^k, 2^{k+1} - 1]$.
- Za pomocą algorytmu Millera-Rabina sprawdź czy p jest liczbą pierwszą
- Jeżeli jest pierwsza to ją zwróć, w przeciwnym wypadku powtórz procedurę.

Skoro wiemy, że w tym przedziale jest $\mathcal{O}(\frac{2^k}{k})$ liczb pierwszych to oznacza że w oczekiwaniu

po $\mathcal{O}(k)$ losowaniach trafimy na liczbę pierwszą. Sprawdzanie z wolnym monożeniem czy liczba jest pierwsza z pomocą algorytmu Millera-Rabina wykonuje się w czasie $\mathcal{O}(k^3)$ czyli w oczekiwaniu otrzymujemy algorytm w złożoności $\mathcal{O}(k^4)$.

3.1.7 Opisać efektywną implementację działań arytmetycznych w ciele skończonym $Z_p/(W)$

- Mamy dodawanie w $\mathcal{O}(n)$ - trywialne.
- Mnożenie standardowe w $\mathcal{O}(n^2)$, można użyć Karatsubę, Tooma-Cooka lub Schonhagego-Strassena by zejść niżej do jakiegoś $\mathcal{O}(n \log n)$
- dzielenie mamy standardowo Hornerem $\mathcal{O}(n^2)$. Można też dzielić szybciej. Jeśli dzielimy $A(X)$ przez $B(X)$, to wtedy dajemy sobie funkcję pomocniczą $rev_k(P(x)) = x^k P(\frac{1}{x})$, a następnie szukamy $rev_m(B(x))^{-1}$ w pierścieniu Taylora (a istnieje ona, bo wyraz wolny u nas to 1, szukamy to Newtonem). I mamy coś takiego

$$rev_n(A) \equiv rev_m(B) \cdot rev_{n-m}(Q) \mod y^{n-m+1}$$

$$rev_n(A) \cdot rev_m(B)^{-1} \equiv rev_{n-m}(Q) \mod y^{n-m+1}$$

  Z tego liczymy $Q = rev_{n-m}(rev_{n-m}(Q))$, a następnie $R = A - BQ$. Lub poprostu robimy rozszerzony algorytm euklidesa bo sensownie i szybko działa i zwraca nam piękinie odwrotność w tym ciele.

3.1.8 Opisać ideę algorytmu AKS (schemat, bez dowodu)

Twierdzenie 1.
Niech $n, a \in \mathbb{Z}$ oraz $\gcd(a, n) = 1$.

$$(X + a)^n = X^n + a \mod n$$

Zachodzi wtedy i tylko wtedy, gdy n jest liczbą pierwsza.
Szkic dowodu na wszelki wypadek:

Jeśli n jest liczbą pierwszą, to wszystkie współczynniki $\binom{n}{k}$ dla $0 < k < n$ są podzielne przez n, a zatem $(X + a)^n = X^n + a^n \mod n$, a dodatkowo z małego twierdzenia Fermata $a^n = a \mod n$. Jeśli n nie jest liczbą pierwszą, to przynajmniej jedno $\binom{n}{k}$ nie jest podzielne przez n, a więc wielomian $(X+a)^n$ zawiera wyraz $\binom{n}{k} X^k a^{n-k}$ (wystarczy wziąć za k najmniejszy dzielnik n), nie może więc być równy $X^n + a \mod n$

Od teraz zacznie się ciekawiej. Niestety obliczenie wielomianu $(X + a)^n$ jest za drogie dlatego obliczenia będziemy prowadzić w pierścieniu ilorazowym $\mathbb{Z}_n[X]/(X^r - 1)$ (r sobie za chwile wyczarujemy). Po tej redukcji okaże się, że jednak jedno a nie wystarczy ale będziemy musieli sprawdzić ich stosunkowo mało.

Idea Algorytmu:

- 1. Sprawdź czy n jest potęgą liczby pierwszej tzn. $n = p^k$ dla pewngo $k \geq 2$ lub czy $2 \mid n$ jeżeli tak to zwróć złożona.
- 2. Znajdź najmniejsze r takie, że rząd $n \mod r$ jest większy niż $\log^2 n$.
- 3. Jeżeli dla jakiegoś $a \leq min(r, n - 1)$ $\gcd(a, n) \neq 1$, to zwróć złożona.

- 4. Jeżeli $n \leq r$ zwróć pierwsza.
- 5. Niech $l = \sqrt{r} \log n$, Dla każdego a takiego, że $1 \leq a \leq l$ sprawdź równość $(X + a)^n = X^n + a \mod (n, X^r - 1)$, jeżeli równość nie zajdzie zwróć złożona.
- 6. Jak nic się wcześniej nie wywaliło to zwróć pierwsza

Szybki argument złożoności:
Najwięcej sprawdzamy w punkcie 5, gdyż r jest rzedu $\mathcal{O}(\log^5 n)$. Z tego wynika, że l jest rzędu $\mathcal{O}(\log^{3.5} n)$, A na obliczenie każdego równania potrzebujesz czasu $\mathcal{O}(r \log^2 n) = \mathcal{O}(log^7 n)$ co wymnażając otrzymujemy $\mathcal{O}(\log^{10.5} n)$.

3.1.9 Pokazać, że wielomianowy algorytm na problem pierwiastka dyskretnego da się zamienić na wielomianowy algorytm na faktoryzację

Twierdzenie 1.
Mamy liczbę n złożoną oraz n nie jest potęgą liczby pierwszej (możemy też założyć że n jest nieparzyste). Wtedy dla dowolnych dla dowolnego u równanie:

$$x = u^2 \mod n$$

Jeżeli to równanie ma jakiegkolwiek rozwiązanie to ma ich conajmniej 4.

Dowód:
Przedstawmy liczbę n jak $n = pq$, takie że p, q są względnie pierwsze (formalnie $\gcd(p, q) = 1$). Skorzystajmy teraz z Twierdzenia o nieresztach kwatradowych czyli z faktu, że każde rozwiązanie ma $x = z^2 \mod w$ ma 0 rozwiazań lub conajmniej 2 (jeżeli ma jedno jakiez l to $-l$ też jest rozwiązaniem) (jeżeli dla p lub q ma 0 to nasz pierwiastek dla n nie istniałby) czyli musi mieć conajmniej 2 rozwiązania. A więc zapiszmy teraz chińskie twierdzenie o resztach:

$$x = u^2 \mod p$$

$$x = u^2 \mod q$$

Z chińskiego twierdzenia o resztach wynika, że rozwiązań równania modulo n jest conajmniej tyle ile rozwiązań modulo q razy ilość rozwiązań modulo p. Czyli wynika z tego, że mamy conajmniej 4 rozwiązania.

Idea algorytmu(funkcją $Root(x, n)$ oznaczamy nasza maszynkę do liczenia pierwiastka dystretnego z $x \mod n$):

- 1. Wylosuj losowo x z przedziału $\{1, ..., n - 1\}$.
- 2. Jeżeli $\gcd(x, n) \neq 1$ znaleźliśmy jakiś dzielnik.
- 3. Obliczmy $y = x^2 \mod n$ oraz $s = Root(y, n)$.
- 4. Jeżeli $s = x$ lub $s = -x$ to powróć do punktu 1.
- 5. (bez straty ogólniości założmy, że $s > x$ (jeżeli nie to $swap(x, s)$)) Mamy teraz $s^2 = x^2 \mod n$ z czego wynika, że $(s + x)(s - x) = 0 \mod n$, co oznacza, że $\gcd(s + x, n)$ lub $\gcd(s - x, n)$ jest nietrywialnym dzielnikiem.

Ogólnie to idee widać dosyć dobrze. Jedyne teraz zobaczmy czemu mamy sensowne prawdobodobieństwo przejścia punktu 4.

Zauważmy, że x losujemy, a algorytmowi podajemy już x^2. Więc algorytm nie wie który z pierwiastków my wylosowaliśmy więc sam odpowiada którymś. A jest są conajmniej 4 różne pierwiastki, a my nie możemy dostać dwóch z nich, co oznacza, że mamy szansę conajmniej $\frac{1}{2}$, że nie pokryjemy się z odpowiedzią naszej czarnej skrzynki.

3.1.10 Zdefiniować problemy Discrete-Log i Diffie-Helman, ich miejsce w klasach złożoności, opisać protokół Diffiego-Helmana

Definicja problemu Discrete-Log:
Mamy dowolną grupę cykliczną G oraz element $g \in G$ będący generatorem wtedy:

Mając na wejściu $a \in G$ znajdź taki x, że $g^x = a$.

Discrete-Log $\in NP$:
Zgadnij x oraz sprawdź czy $g^x = a$, jeżeli tak to zwróć x.

Niestety więcej o tym problemie nie wiemy. W kryptografi zakładamy, że Discrete-Log $\notin P$ oraz Discrete-Log $\notin BPP$ ale tego nie wiemy! Nie wiemy też czy jest to problem trudny w klasie NP.

Definicja problemu Diffie-Helman: Mamy dowolną grupę cykliczną G oraz element $g \in G$ będący generatorem wtedy:

Mając na wejsciu g^x, g^y (x, y nie jest podane) znajdź g^{xy}.

Diffie-Hellman $\in NP$:
Zgadnij x, sprawdź czy g^x równa się temu z wejścia równa się temu z wejścia. Wykonaj teraz $(g^y)^x = g^{xy}$ bo znamy x i zwróc g^{xy}.

Niestety w tym przypadku też nie jesteśmy w stanie powiedzieć wiecej na temat należenia tego problemu do innych klas, które nas interesują. Natomias możemy stwierdzić, że za pomocą Discrete-Log możemy rozwiązać Diffie-Helmana (poprostu obliczmy x i postępujemy tak jak w dowodzie dla NP).

Protokół Diffiego-Helmana:
Jest to protokół symetryczny czyli wyślemy sobie nawzajem klucze publiczne i stworzymy na podstawie go nasz klucz symetryczny. Klucz prywatny to a oraz dla drugiej osoby b. Do publicznej wiadomości dajemy g^a oraz druga osoba g^b. Naszym kluczem symetrycznym będzie g^{ab}. Jak widać jest on dosyć podobny do El-Gammal.

3.1.11 Podać definicję krzywej eliptycznej i grupy z nią związanej

3.1.11.1 Definicja Krzywa eliptyczna:

To zbiór rozwiązań w pewnym ciele $\mathbb{F}$ (punktów (x, y)) równania:

$$y^2 = x^3 + ax + b$$

(Jest to krzywa w postaci Weierstrassa i każdą krzywą można do takiej postaci sprowadzić równanie ogólne wygląda dziko czyli tak: $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$).

My chcemy aby nasz krzywa była "gładka"i nie miała óstrzaćzyli formanie wyznacznik krzywej $\delta = 4a^3 + 27b^2$ musi być różny od 0.

3.1.11.2 Definicja grupy:

Jeśli $P = (x, y)$ leży na krzywej eliptycznej, to $(x, -y)$ też i oznaczmy go przez $-P$. Jeśli P i Q są punktami na krzywej eliptycznej, to prosta PQ musi (prawie zawsze!) przeciąć krzywą w jeszcze jednym punkcie R. Definiujemy sumę punktów $P+Q$ jako $-R$ (uwaga, tam jest minus przed R!). Aby obsłużyć przypadek $Q = -P$, dodajemy zatem do krzywej jeszcze sztuczny punkt O, leżący "w nieskończoności"i definiujemy $P + (-P) = O$ (a także $P + O = P$). Aby z kolei obliczyć sumę $P + P$, rysujemy styczną do krzywej w punkcie P, znajdujemy jej punkt przecięcia Q z krzywą, i bierzemy $P + P = -Q$, chyba że nie ma takiego Q to zwracamy O. (Ogólnie jak nie ma jakiegoś punktu to mówmy w skrócie O, za wyjątkiem $P + O$)

3.1.11.3 Dzikie wzory:

Mając $P = (x_P, y_P)$ oraz $Q = (x_Q, y_Q)$, to możemy obliczć $P + Q = S = (x_S, y_S)$ ze wzorów:

$$x_S = \lambda^2 - x_P - x_Q$$

$$y_S = -y_P - \lambda(x_S - x_P)$$

gdzie $\lambda = \frac{3x_P^2+a}{2y_p}$ jeżeli $P = Q$, w przeciwnym wypadku $\lambda = \frac{y_Q-y_P}{x_Q-x_P}$

3.2 Grupa B

Czyli tak zwana grupa pytań średnich.

3.2.1 Opisać algorytm Karatsuby mnożenia dużych liczb binarnych

Algorytm mnożenia Karatsuby opiera się o technikę, dziel i zwyciężaj oraz zauważeniem jednego ciekawego faktu jak możemy oszczędzić jedno mnożenie.

Idea:
Na wejściu otrzymujemy dwie liczby A oraz B, n-cyfowe (mogą być różnych rozmiarów ale to jest tylko kwestia techniczna jak to rozwiązać), możemy założyć, że są to liczby w systemie binarnym i n jest potęgą dwójki (jak nie to dopychamy zerami od przodu).

Następnie podzielmy nasze liczby na pół (względem długości zapisu) i przedstawmy je jako:

$$A = A_1 \cdot K + A_0$$
$$B = B_1 \cdot K + B_0$$

Gdzie $K = 2^{n/2}$.

Zauważmy teraz, że

$$A \cdot B = A_1B_1K^2 + A_1B_0K + A_0B_1K + A_0B_0$$

Jest to zwykłe mnożenie po rozbiciu A oraz B. Zauważmy, że mnożenie przez K jest w czasie $\mathcal{O}(n)$, gdyż jest to zwykłe przesunięcie bitowe oraz dodawanie jest również w czasie $\mathcal{O}(n)$ więc jedynymi trudnościami są tutaj iloczyny w postaci A_iB_j.

Następną rzeczą, na którą musimy wpać jest jak zaoszczędzić jedno mnożenie.

$$(A_0 + A_1)(B_0 + B_1) = A_0B_0 + A_0B_1 + A_1B_0 + A_1B_1$$

Z czego wynika

$$(A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1 = A_0B_1 + A_1B_0$$

A więc możemy obliczyć rekurencyjnie A_0B_0 oraz A_1B_1. Następnie policzyć rekurencyjnie $(A_0+A_1)(B_0+B_1)$ i odjąć od niego A_0B_0 oraz A_1B_1 i otrzymać $A_0B_1+A_1B_0$. Teraz wystarczy podłożyć to do naszego wzoru:

$$A \cdot B = A_1B_1K^2 + A_1B_0K + A_0B_1K + A_0B_0$$

$$A \cdot B = A_1B_1K^2 + (A_1B_0 + A_0B_1)K + A_0B_0$$

$$A \cdot B = A_1B_1K^2 + ((A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1)K + A_0B_0$$

I jak widać musimy wykonać tylko 3 różne mnożenia rekurencyjnie.

Dzięki temu otrzymujemy następującą postać złożoności $T(n) = 3T(\frac{n}{2}) + \mathcal{O}(n)$. Co dzięki Uniwersalnemu twierdzeniu o rekurencji mówi nam, że złożoność całego algorytmu wynosi $\mathcal{O}(n^{\log_2 3}) \approx \mathcal{O}(n^{1.59})$

3.2.2 Opisać algorytm Tooma-Cooka mnożenia dużych liczb binarnych

Na wejściu dostajemy dwie liczby A i B zapisane binarnie, każda o n bitach. Rozbijmy je podobnie jak w algorytmie Karatsuby tylko tym razem z $K = 2^{\frac{n}{3}}$.

$$A = A_2K^2 + A_1K + A_0$$

$$B = B_2K^2 + B_1K + B_0$$

Potraktujmy teraz A i B jak wielomiany, czyli:

$$A(X) = A_2X^2 + A_1X + A_0$$

$$B(X) = B_2X^2 + B_1X + B_0$$

Obliczmy teraz wartości wielomianów $A(X)$ i $B(x)$ w punktach $0, 1, -1, 2, -2$, czyli:

$$A(0) = A_0$$

$$A(1) = A_2 + A_1 + A_0$$

$$A(-1) = A_2 + -A_1 + A_0$$

$$A(2) = 4A_2 + 2A_1 + A_0$$
$$A(-2) = 4A_2 + -2A_1 + A_0$$

Jak widać złożoność obliczania wartości w każdym z tych punktów wynosi $\mathcal{O}(n)$, gdyż dodawanie i mnożenie przez 2 i 4 możemy wykonywać w czasie $\mathcal{O}(n)$. Wykonajmy również takie same obliczenie dla wielomianu $B(X)$. Skorzystajmy z twierdzenia, że skoro f, g są wielomianami to:

$$f(x) \cdot g(x) = (f \cdot g)(x)$$

A więc jeżeli przez oznaczymy sobie $C(X) = A(X) \cdot B(X)$, ($C(X)$ przy okazji jest wynikiem który chcemy uzyskać) to mamy, że $C(0) = A(0) \cdot B(0), C(1) = A(1) \cdot B(1)$ itd. A więc zauważmy, że każde naszych $A(0), B(0), A(1), ...$ ma długość rzedu $\frac{n}{3}$ i obliczmy każde $C(0), C(1), C(-1), ...$ rekurencyjnie. A więc jak narazie wykonaliśmy $\mathcal{O}(n)$ operacji oraz 5 wywołań rekurencyjnych na liczbach długości $\frac{n}{3}$. Zauważmy, że:

$$A(X) \cdot B(X) = C(X) = C_4X^4 + C_3X^3 + C_2X^2 + C_1X + C_0$$

oraz mamy oblicznone już każde z $C(0), C(1), C(-1), ...$, które możemy rozpisać jako:

$$C(0) = C_0$$
$$C(1) = C_4 + C_3 + C_2 + C_1 + C_0$$
$$C(-1) = C_4 - C_3 + C_2 - C_1 + C_0$$
$$C(2) = 16C_4 + 8C_3 + 4C_2 + 2C_1 + C_0$$
$$C(-2) = 16C_4 - 8C_3 + 4C_2 - 2C_1 + C_0$$

A więc możemy kolejno wyliczać współczynniki Gaussem (w dobrej kolejności lub rozpisać wzroki na pałę jak kto woli) i otrzymać w czasie $\mathcal{O}(n)$ wszystkie współczynniki wielomianu $C(X)$. Wiemy teraz, że:

$$C = C_4K^4 + C_3K^3 + C_2K^2 + C_1K + C_0$$

Więc również liczmy to w czasie $\mathcal{O}(n)$. Podsumowując złożoność mamy: $T(n) = 5T(\frac{n}{3}) + \mathcal{O}(n)$, co z Uniwersalnego twierdzenia o rekurencji daje nam złożoność $\mathcal{O}(n^{\log_3 5}) \approx \mathcal{O}(n^{1.46})$

3.2.3 Zapisać i udowodnić chińskie twierdzenie o resztach

Układ kongruencji

$$x \equiv_{n_1} a_1$$
$$x \equiv_{n_2} a_2$$
$$...$$
$$x \equiv_{n_k} a_k$$

(gdzie n_i i n_j są względnie pierwsze dla $i \neq j$) ma dokładnie jedno rozwiązanie $x < N$ ($N = n_1 \cdot n_2 \cdot ... \cdot n_k$)

3.2.3.1 Unikalność

Załóżmy, że dany układ rekurencji ma dwa rozwiązania x i y.
Zauważmy, że $x \equiv_{n_i} a_i$ oraz $y \equiv_{n_i} a_i$. Zatem otrzymujemy $x - y \equiv_{n_i} 0$.
Jako że n_i są parami względne, to zachodzi również $x - y \equiv_N 0$. Ponieważ $x < N$ i $y < N$ to $x - y$ jest wielokrotnością N tylko dla $x = y$.

3.2.3.2 Istnienie

Korzystamy z twierdzenia Bezuta (dla względnie pierwszych liczb całkowitych x i y istnieją liczby całkowite a i b takie że $ax + by = 1$).
Niech $N_i = N/n_i$ oraz $N_i M_i + n_i m_i = 1$ (z Bezuta). Wtedy otrzymujemy rozwiązanie kongruencji postaci $x = \sum_{i=1}^{k} a_i N_i M_i$.
Czemu to działa? Jak łatwo zauważyć $a_i N_i M_i \equiv_{n_j} 0$ dla $j \neq i$, ponieważ $N_i \equiv_{n_j} 0$. Natomiast $a_i N_i M_i \equiv_{n_i} a_i$ gdyż mamy $N_i M_i + n_i m_i = 1 \implies N_i M_i \equiv_{n_i} 1$.

3.2.4 Zdefiniować pojęcie ideału, pierścienia ilorazowego oraz pokazać, że $Z_p[X]/(W)$ jest ciałem wtw gdy W jest nierozkładalny

3.2.4.1 Definicja Ideału:

Jeżeli R jest pierścieniem to ideał $I \subseteq R$ jest ideałem wtw gdy:

- $x, y \in I \implies x + y \in I$ (zamknięcie na sumę)
- $x \in I, y \in R \implies y \cdot x \in I$ (własność wciągania (jest to silniejsze niż zamkniętość na mnożenie))

3.2.4.2 Definicja Pierścień ilorazowy:

Zdefiniujmy sobie relację $x \sim y \iff x - y \in I$, która jest relacja równoważności. Zbiór jej klas abstrakcji jest to $R/I = \{x + I : x \in R\}$ jest właśnie pierścieniem ilorazowym. Mniej forlmalnie poprostu mówimy, że liczymy modulo ideał I.

3.2.4.3 Twierdzenie:

Pierścień $\mathbb{Z}_p[X]/(W)$ jest ciałem wtw. gdy W jest nierozkładalny (nie ma nietrywialnych dzielników czyli W nie da się przedstawić za pomocą $g_1 \cdot g_2$, gdzie g_1, g_2 nie są wielomianami stałymi).

3.2.4.4 Dowód:

($\implies$)
$\mathbb{Z}_p[X]/W$ jest ciałem to załóżmy nie wprost, że W jest rozkładalny. Wynika z tego, że W można przedstawić jako $W = a \cdot b$, gdzie a, b nie są wielomianami stałymi i oba mają mniejszy stopień niż W więc same nie są zerami. A wynika z tego również, że $a \cdot b = 0$ w ciele $\mathbb{Z}_p[X]/W$, czyli są dzielnikami zera. Z czego wynika że nie mogą mieć elementu odwrotnego co prowadzi do sprzeczności.

($\impliedby$)
Wiemy, że $\mathbb{Z}_p[X]/W$ jest pierścieniem, wystarczy pokazać, że każde a stopnia mniejszego niż W ma odwrotność. Wiemy również z faktu, że W jest nierozkładalny, że $\gcd(a, W) = 1$. Wynika z tego, że możemy zastosować rozszerzony algorytm eulidesa do znajdowania odwrotnośći, gdyż znajdziemy s, t, takie, że $a \cdot s + W \cdot t = 1 \implies a \cdot s = 1 \mod W$. Czyli nasze s jest elementem odwrotnym z czego wynika że mamy ciało.

3.2.5 Pokazać, że każde ciało skończone musi mieć p^k elementów dla pewnej liczby pierwszej p oraz całkowitego k

3.2.5.1 Twierdzenie 1. (O charakterystyce):

Charakterystyka ciała skończonego zawsze jest dodatnia i jest liczbą pierwszą.

3.2.5.2 Dowód Tw. 1:

1. Nie zerowość.

Nie wprost zakładamy, że nie otrzymujemy 0 (poprzez dodawanie jedynki) a wykonaliśmy więcej niz ilość elementów w ciele dodań 1. Co oznacza, że jakiś element musiał się powtórzyć. Jeżeli jakiś element się powtórzył to z definicji suma między powtórzeniami wynosi 0, gdyż jest ono elementem neutralnym dodawania. Czyli 0 musiało wystąpić sprzeczność.

2. Charakterystaka jest liczbą pierwszą.

Załóżmy nie wprost, że charakterystyka jest liczbą złożoną równą n. Możemy z tego faktu rozbić $n = pq$ gdzie $p, q > 1$. W takim razie $(1 + 1 + ... + 1)(n \text{ razy}) = (1 + 1 + ... + 1)(q \text{ razy}) \cdot (1 + 1 + ... + 1)(p \text{ razy})$ z czego wynika, że p lub q jest 0. Co oznacza że n nie jest najmniejsza taką liczbą która po dodaniu n razy 1 otrzymamy 0. Czyli mamy sprzeczność z definicji charakterystyki.

3.2.5.3 Twierdzenie 2.

Niech $\mathbb{F}$ będzie ciałem skończonym charakterystyki p. Wtedy istnieje takie k, że $|\mathbb{F}| = p^k$.

3.2.5.4 Dowód Tw. 2

Wiemy, że $\mathbb{Z}_p$ jest ciałem. Możemy więc wprowadzić mnożenie przez element z $\mathbb{Z}_p$ elementów z ciała $\mathbb{F}$ zdefiniowane jako $a \cdot x = x + x + ... + x(arazy)$. Wynika z tego, że mamy teraz przestrzeń linową na Z_p gdyż mamy ciało oraz mnożenie przez skalar. Skoro $\mathbb{F}$ jest przestrzenią liniową nad Z_p to ma ona skończony wymiar, nazwijmy go k, oraz jakąś baze $x_1, ..., x_k$. A więc każdy element $x \in \mathbb{F}$ możemy zapisać jako $a_1x_1 + a_2x_2 + ... + a_kx_k$. Różne ciągi $(a_1, ..., a_k)$ dają różne elementy z $\mathbb{F}$ z czego wynika, że liczba elementów $\mathbb{F}$ jest taka sama jak liczba ciągów $(a_1, ..., a_k)$ czyli p^k.

3.2.6 Opisać algorytm faktoryzacji Fermata

Chcemy rozłożyć liczbę n na czynniki pierwsze.
Założenia:
n - nieparzyste (inaczej dzielimy przez 2 dopóki możemy)
n - ma rozkład jakiś rozkład, załóżmy, że $n = pq$ (p, q nie muszą być pierwsze)

Załóżmy, bez stray ogólności że $p > q$. Weźmy sobie teraz $a = \frac{p+q}{2}$ oraz $b = \frac{p-q}{2}$ i zauważmy fakt, że:

$$a^2 - b^2 = (a + b)(a - b) = (\frac{p + q + p - q}{2}) \cdot (\frac{p + q - p + q}{2}) = pq = n$$

A więc jeżeli znajdziemy takie a, że $a^2 - n = b^2$, gdzie b^2 jest dowolnym kwadratem liczby naturalnej to wyciągniemy z nich informację o p oraz q ($p = a + b, q = a - b$).

Idea algorytmu:

- 1. Weź początkowo $a = \lceil \sqrt{n} \rceil$
- 2. Sprawdź czy $a^2 - n$ jest kwadratem liczby naturalnej (binsearch czy cokolwiek), Jeżeli tak to przewij bo masz a oraz b
- 3. W przeciwnym wypadku zwiększ a o 1 i wróć do kroku 1.

Ciekawa własność tego algorytmu to fakt, że znaleźliśmy dzielnik po $a - \sqrt{n}$ krokach (gdyż zaczynaliśmy na $\sqrt{n}$ a skonczyliśmy na a) a więc wykonaliśmy następującą ilość operacji:

$$a - \sqrt{n} = \frac{a^2 - n}{a + \sqrt{n}} = \frac{b^2}{a + \sqrt{n}} \leq \frac{b^2}{\sqrt{n}} \leq \frac{(p-q)^2}{4\sqrt{n}}$$

Czyli dla p, q blisko siebie działa bardzo szybko a dla $p - q \leq \sqrt[4]{n}$ działa w czasie stałym. Niestety jego pesymistyczna złożoność wynosi $\mathcal{O}(n)$, gdyż pesymistycznie z a musimy dojść aż do samego n

3.2.7 Opisać algorytm DSA

3.2.7.1 Przygotowanie do Algorytmu:

- Wbieramy dwie duże liczby pierwsze p, q, takie, że $q \mid p-1$.(Standardowo q ma 256 bitów, a p ma 2048.)
- Znajdujemy element a, taki, że rząd a modulo p wynosi q. (Robimy to losując g oraz podstawiając $a = g^{\frac{p-1}{q}} \mod p$, teraz jeżeli $a \neq 1$, to $a^q = 1 \mod p$, gdyż q jest liczbą pierwszą (A mamy tw. Lagrange)
- Podajemy do wiadomości publicznej p, q, a (Są one stałą częścią algorytmu)

3.2.7.2 Generacja kluczy:

- Wylosuj $x \in \{0, ..., p-1\}$, oraz oblicz $y = a^x$.
- Wartość x to klucz prywatny.
- Wartość $y = a^x$ to klucz publiczny.

Możemy zaobserwować, że aby odtworzyć klucz prywatny z klucza publicznego to musimy rozwiązać problem logarytmu dyskretnego.

3.2.7.3 Podpisywanie wiadomości:

- Generujemy hash H wiadomości, którą chcemy podpisać (Pamiętamy, że wartości a, p, q są znane oraz mamy nasze klucze).
- Losujemy k, takie, że $1 < k < q$.
- Obliczamy $r = (a^k \mod p) \mod q$.
- Obliczamy $s = \frac{H + x \cdot r}{k} \mod q$
- Zwróć parę (r, s).

3.2.7.4 Weryfikacja podpisu:

- Oblicz $\alpha = \frac{H}{s} \mod q$.
- Oblicz $\beta = \frac{r}{s} \mod q$.
- Oblicz $\gamma = (a^{\alpha} \cdot y^{\beta} \mod p) \mod q$.
- Sprawdź czy $\gamma = r$.

3.2.7.5 Dowód działania:

Oznaczmy sobie $w = (H+x\cdot r)^{-1} \mod q$ (czyli odwrotność s ale bez k jeszcze). Teraz $\alpha = w\cdot k\cdot H \mod q, \beta = w \cdot r \cdot k$. Wiemy, że a ma rząd q modulo p co oznacza, że dla dowolnego t zachodzi $a^t \mod p = a^{t \mod q} \mod p$. Rozpiszmy teraz $\gamma = (a^{\alpha} \cdot y^{\beta} \mod p) \mod q = (a^{w\cdot k\cdot H \mod q} \cdot (a^x)^{w\cdot r\cdot k} \mod p) \mod q = a^{k\cdot w\cdot (H+x\cdot r) \mod q} \mod p \mod q$. A z definicji w jest odwrotnością $(H+x\cdot r)$ modulo q, czyli Zachodzi $\gamma = (a^k \mod p) \mod q$ co z definicji wynosi r. Co należało pokazać.

3.2.8 Opisać metodę Baby-Step-Giant-Step

Owy algorytm był na ASD dlatego to zostawiam, jedynie szybka idea. Dzielimy na pierwiastki i sprowadzamy to do formy $i = a \cdot \sqrt{n} + d, d < \sqrt{n}$ (gdzie i bedzie symbolizowało odpowiednią potęge generatora/podstawy), i wyliczamy wartości dla wszystkich możliwych d, a potem iterujemy się po a i sprawdzamy czy istnieje odpowiednie d.

3.2.9 Opisać kryptosystem plecakowy i uzasadnić, dlaczego nie jest stosowany w praktyce

Kryptowaluty... Nie to nie o tym ta część :(

3.2.9.1 SubSet-Sum Definicja:

Mamy sobie zbiór liczb $V = \{v_1, v_2, ..., v_2\}$ oraz liczbę s, stwierdź czy zbiór $A \subseteq V$ taki, że $\sum_{v\in A} v = s$.
Jak już wiadomo z ASD jest to problem NP-zupełny czyli nie spodziewamy się rozwiązania wielomianowego (no chyba, że P = NP).

3.2.9.2 Definicja ciągu nadrosnącego:

Ciągiem nadrosnącym nazywamy taki ciąg $v_1, v_2, ..., v_n$, taki, że dla każdego i zachodzi $v_i > v_1 + v_2 + ... + v_{i-1}$.

Jak możemy zauważyć problem SUBSET-SUM jest prostu dla ciągu nadrosnącego (poprostu robimy zachłana od największych i można prosto udowodnić, że jeżeli możemy wziąc jakiś największy to musimy go wziąć)

3.2.9.3 Idea kryptosystemu plecakowego

- Bierzemy sobie nadrosnący ciąg $v_1, ..., v_n$, $m > \sum_i v_i$ oraz a względnie pierwsze z m.
- Konstruujemy ciąg $w_1, ..., w_n$, tak, że $w_i = a \cdot v_i \mod m$

- Kluczem publicznym jest ciąg $w_1, ..., w_n$.
- Szyfrowanie: chcąc zaszyforwać n-bitową wiadomość $b_1, ..., b_n$ (gdzie b_i to i-ty bit) wyonujemy $s = \sum_i w_i \cdot b_i$, i wysyłamy s.
- Deszyfrowanie: mamy $s = \sum_i b_i \cdot w_i$, zauważmy, że z konstrukcji $w_i = v_i \cdot a \mod m$, więc weźmy sobie odwrotność a modulo m i oznaczmy je jako c. Wynika z tego, że $s \cdot c = \sum_i b_i \cdot v_i \mod m$. A wiemy z definicji, że $m > \sum_i v_i$, czyli możemy to jednozancznie wyliczyć naszym algorytmem zachłannym.

3.2.9.4 Dlaczego nie stosujemy kryptosystemu plecakowego:

Okazuje się, że owy problem jest tylko szczególnym przypadkiem problemu SUBSET-SUM wiec NIE! musi być on NP-zupełny. Co więcej znany jest algorytm wielomianowy go rozwiązujący więc ten problem jest w P(Adi Shamir (1982)).

3.2.10 Pokazać, że pierwiastki wielomianu $X^q - X$ dla $q = p^k$ stanowią ciało, podać (inną) praktyczną metodę generowania ciała skończonego

Jest delikatny blef w pytaniu do tego zadania bo ten wielomian musi być nad ciałem skończonym.

3.2.10.1 Twierdzenie

Dane jest ciało $\mathbb{F}$ charakterystyki p, oraz liczba $q = p^k$ dla pewnego k. Jeśli wielomian $f(X) = X^q - X$ ma q pierwiastków, to stanowią one ciało (q-elementowe podciało $\mathbb{F}$).

3.2.10.2 Dowód

Niech $A = \{a \in \mathbb{F} : a^q = a\}$ (czyli poprostu nasze pierwiastki z definicji. Pokażemy teraz zamknięcie na operacje, gdyż przemienność, łączność, etc. mamy z operacji na ciele $\mathbb{F}$.

- $0 \in A$, gdyż $0^q = 0$.
- $1 \in A$, gdyż $1^q = 1$.
- $-1 \in A$, gdyż $(-1)^q = -1$ (dla $p = 2$ zachodzi $1 = -1$, gdyż charakterystyka wynosi 2, czyli $1 + 1 = 0 = 1 + -1$).
- Jeżeli $a, b \in A$, to $a \cdot b \in A$, gdyż $(ab)^q = a^q b^q = ab$ czyli jest w A. W szczególności mamy stąd, że jeżeli $a \in A$ to $-a \in A$, gdyż $-a = (-1)a/$
- Pokażemy, że jeżeli $a, b \in A$, to $a + b \in A$. Ale najpierw udowidnimy sobie coś pomocniczego, czyli że dla elementów z ciała $\mathbb{F}$ zachodzi: $(a + b)^p = a^p + b^p$ dla dowolnych $a, b \in \mathbb{F}$. Rozbijamy z dwumianu Newtona $(a+b)^p = a^p + \binom{n}{1} a^{p-1}b + ... \binom{n}{n-1} ab^{p-1} + b^p$. Zauważamy, że każdy z dwumianów na środku jest podzielny przez p (gdyż w liczniku jak rozpiszemy dwumian mamy czynnik p, a w mianowniku nie mamy żadnego). Czyli z faktu, że p jest charakterystyką te wyrazy się zerują czyli mamy tezę. Rozpiszmy więc nasze dodawanie i skorzystajmy z faktu, że $q = p^k$. $(a + b)^q = (a + b)^{p^k} = (a + b)^{p \cdot p^{k-1}} = (a^p + b^p)^{p^{k-1}} = (a^{p^2} + b^{p^2})^{p^{k-2}} = ... = a^{p^k} + b^{p^k} = a^q + b^q = a + b$.

Generacja innych grup o liczności p^k, to poprostu strzelamy w wielomian stopnia k i sprawdzamy czy jest nierozkładalny i mamy na to bardzo dużą szansę.

3.2.11 Opisać algorytm Tonellego-Shanksa

Zacznijmy od kilku twierdzeń:

Twierdzenie 1.
Grupa cykliczna G, taka, że $|G| = n = 2m$, ma dokładnie m kwadratów (czyli połowe swojego rozmiaru). Oraz każdy kwadrat ma dokładnie dwa pierwiastki. (co więcej pokażemy, że parzyste potęgi generatora to kwadraty, a nieparzyste nie)

Dowód:
Weźmy sobie g generator grupy G, zauważmy, że każdy element grupy G należy do zbioru $\{g^0, g^1, g^2, ..., g^{2m-1}\}$, czyli potęgi generatora wynosą są modulo $2m$. A rozważmy dwa przypadki:

- 1. $a = g^k = g^{2j}$, k jest parzyste (czyli jest w parzystą potęgą generatora):
  Wtedy możemy zauważyć, że piewiastkami są g^j, g^{j+m}, gdyż $(g^j)^2 = g^{2j} = a$ oraz $(g^{j+m})^2 = g^{2j+2m} = g^{2j} = a$ oraz nie ma żadnego innego.
- 2. $a = g^k$, k jest nieparzyste:
  Załóżmy teraz nie wprost, że istnieje b, ktore jest pierwiastkiem i jest w postaci $b = g^j$, wynika z tego, że $b^2 = g^{2j} = a = g^k$, co oznacza, że $2j = k \mod 2m$, co prowadzi do sprzeczności, gdyż k jest nie parzyste a reszta z dzielenia jak i współczynnik przez który bierzemy modulo jest parzysty.

Czyli udowodniliśmy sobie pierwsze twierdzenie.

Twierdzenie 2.
Mamy grupę cykliczna G, taka, że $|G| = n = 2m$. Element równanie $x^2 = a$, takie, że $x, a \in G$ ma rozwiązanie (co oznacze, że a jest kwadratem) wtedy i tylko wtedy, gdy $a^m = 1$ (dla nie kwadratów $a^m = -1$ ($1, -1$ są to tylko symbole, gdyż grupa G to nie muszą być liczby).

Dowód:
Mamy przypadki (pokaże dwa reszta idzie analogicznie):

- 1. a jest kwadratem:
  Z poprzedniego twierdzenia wynika, że jeżeli a jest kwadratem to jest w postaci $a = g^{2j}$. Wiec z tego wynika, że $a^m = g^{2jm} = g^0$, gdyż potęgi generatora bierzemy modulo $2m$.
- 2. a nie jest kwadratem:
  Z poprzedniego Tw wiemy, że a jest w postaci $a = g^j$, gdzie j jest nie parzyste, a więc $g^{mj} = g^m = -1$, gdyż bierzemy potęgi generatora modulo $2m$, a j jest nie parzyste.

Idea algorytmu: Mamy grupę G, $|G| = n = 2m$.

- 1. $q = m$, $t = n$
- 2. wylosuj z, które nie jest kwadratem, czyli gdy zajdzie $z^m \neq 1$ (powtarzaj ten krok dopóki dobrze nie wylosujesz)
- 3. Dopóki $2 \mid q$ wykonaj $q := \frac{q}{2}$, $t := \frac{t}{2}$. Jeżeli dla nowego q, t $a^q z^t \neq 1$ to $t := t + m$. I powtórz ten krok.
- 4. Zwróć $a^{\frac{q+1}{2}} z^{\frac{t}{2}}$

Zauważ, że w każdym kroku algorytmu trzymamy niezmiennik, że $a^q z^t = 1$. A więc to co zwróciliśmy $a^{\frac{q+1}{2}} z^{\frac{t}{2}} = x$, i zobaczmy że jest to poprawny wynik. $x^2 = a^{q+1} z^t = a \cdot a^q z^t = a$

czyli działa.
Niezmienniki jakie utrzymujemy:

- $a^q z^t = 1$
- Jeżeli $2^r \mid q$ to $2^{r+1} \mid t$

Początkowo oczywiście jest to spełnione. Krok w niezmiennikach:

- Jeżeli $a^{\frac{q}{2}} z^{\frac{t}{2}} = 1$, to trywialnie niezmienniki są spełnione.
- W przeciwnym wypadku $a^{\frac{q}{2}} z^{\frac{t}{2}} = -1$ (gdyż jakby się przyjrzeć w rozpisanie tego to zmniejszamy potęgę generatora dwukrotnie wiec może to być tylko -1). A z definicji z i Tw 2 wiemy,że $z^m = -1$, więc po operacji $t := \frac{t}{2} + m$ mamy $a^{\frac{q}{2}} z^{\frac{t}{2}} z^m = -1 \cdot -1 = 1$, czyli jest ok oraz wiemy, że m jest wielkrotnością $2q$ więc drugi niezmiennik dalej zachodzi.

Zauważmy jeszcze, że z Tw 1 połowa elementów nie jest kwadratami więc w punkcie 2. losujemy z prawdopodobieństwem $\frac{1}{2}$ (czyli w oczekiwaniu po stałej liczbie kroków mamy dobre z).

Zauważmy również, że cały algorytm ma $\mathcal{O}(\log n)$ iteracji (a w czasie iteracji mamy mnożenia, dzielenia i podnoszenie do potęgi) więc jest on wielomianowy.

3.3 Grupa C

Czyli grupa zadań, których lepiej nie robić

3.3.1 Opisać algorytm Millera-Rabina

Jak wszyscy wiedzą liczby pierwsze spełniają twierdzenie Fermata, tj. $a^{p-1} \equiv_p 1$. No i fajnie by było gdybyśmy mogli sobie odpalić ten test na naszej liczbie i wtedy mówimy czy ona jest pierwsza czy nie. Problem w tym że są liczby złożone, które i tak spełniają to twierdzenie. Dlatego trzeba zmodyfikować Fermata, by lepiej działał.
Najpierw ważny fakt - w grupie Z_p istnieją tylko dwa takie elementy x, że $x^2 \equiv_p 1$ (oczywiście są to 1 i $p-1$), ponieważ $(x-1)(x+1) \equiv_p x^2 - 1 \equiv_p 0$. A dla grup Z_n chyba jest ich więcej, bo wiem że dla 15 się psuje.
Przechodząc do algorytmu samego w sobie, pytamy czy n jest pierwsza:

1. Sprawdzamy czy Fermat działa, jeśli nie, to zwracamy NIE
2. liczymy sobie maksymalną potęge 2 w $n-1$, czyli szukamy takiego s, że $n - 1 = 2^s k$
3. Losujemy jakieś $a \in [1, 2, ..., n-1]$
4. Liczymy po kolei każde $a^{2^i k}$
5. Jeśli wystąpiła sytuacja, że dla $a^{2^i k} = 1$ zaszło $a^{2^{i-1} k} \neq 1$ i $a^{2^{i-1} k} \neq -1$, to zwracamy NIE
6. zwracamy PRAWDOPODOBNIE TAK

Podany algorytm zwraca NIE z 100% poprawnością, a tak to może się mylić z prawdopodobieniestwem, jakimś $\frac{1}{4}$.
Można udowodnić, że dla liczb $n < 2^{64}$ wystarczy sprawdzić a ze zbioru pierwszych 12 liczb pierwszych. BARK DOWODÓW OBECNIE

3.3.2 Pokazać, że grupa multiplikatywna ciała skończonego jest grupą cykliczną

3.3.3 Opisać algorytm "ro" Pollarda na faktoryzację

3.3.4 Opisać algorytm sita kwadratowego

3.3.5 Opisać algorytm "ro" Pollarda na logarytm dyskretny

3.3.6 Opisać algorytm Pohliga-Hellmana

3.3.7 Opisać algorytm rachunku indeksów

3.3.8 Opisać ideę algorytmu Schonhage-Strassena

3.3.9 Opisać algorytm Schreiera-Simsa

3.3.10 Opisać algorytm Grovera

3.3.11 Opisać ideę (bez dowodów) algorytmu Shora