Ćwiczenia 7

Algorytm Euklidesa

- Pierwsza wersja:

  ```
  function Euclid(a,b)
    if b = 0 then
      return a
    else if a < b then
      return Euclid(b,a)
    else
      return Euclid(a − b,b)
  ```

  Złożoność: fatalnie — $O(n)$, np. dla $Euclid(2^k, 1)$ algorytm wykona 2^k operacji.

- Wersja z operacją modulo:

  ```
  function Euclid(a,b)
    if b = 0 then
      return a
    else if a < b then
      return Euclid(b,a)
    else
      return Euclid(b,a mod b)
  ```

  Złożoność: $O(\log^2 n)$.

- Wersja z dzieleniem i odejmowaniem:

  ```
  function Euclid(a,b)
    if b = 0 then
      return a
    else if a < b then
      return Euclid(b,a)
    else if a mod 2 = 0 and b mod 2 = 0 then
      return 2 · Euclid(a/2,b/2)
    else if a mod 2 = 0 then
      return Euclid(a/2,b)
    else if b mod 2 = 0 then
      return Euclid(a,b/2)
    else
      return Euclid(a − b,b)
  ```

  Złożoność: taka jak poprzednio $O(\log^2 n)$, ale znacznie prostszy w implementacji.

- Rozszerzony algorytm Euklidesa, rozwiązujący równania typu:

  $$nwd(a, b) = ax + bx$$

  ```
  function ExtEuclid(a,b)
  ```

```
if b = 0 then
  return (a, 1, 0)
(d', x', y')=ExtEuclid(b,a mod b);
return (d', y', x' - ⌊a/b⌋y')
```

algorytm zakłada, że $a \geq b$.

Rozwiązania równania $ax \equiv b \pmod{n}$

Dane: a, b i n
Wynik: x

- jeśli $b \not\equiv 0 \pmod{nwd(a,n)}$, to równanie nie ma rozwiązań,
- wpp. niech $ax' + n'y = nwd(a,n)$, to $x = \frac{x'b+in}{nwd(a,n)}$ dla $i \in Z$.

Udowodnij

$$(a+b)^p \equiv a^p + b^p \equiv a + b \pmod{p}$$

dla $a, b \in Z_p^*$, p — liczba pierwsza

- dla $0 < k < p$, p dzieli $\binom{p}{k}$
- $(a+b)^p = \binom{p}{0}a^p + \binom{p}{1}a^{p-1}b + \binom{p}{2}a^{p-2}b^2 + \ldots + \binom{p}{p-1}a^b p - 1 + \binom{p}{p}b^p$
- stąd $(a+b)^p \equiv a^p + b^p \pmod{p}$,
- z małego Tw. Fermata $a^{p-1} \equiv 1 (\mathrm{mod}\ p)$, stąd $a^p \equiv a \pmod{p}$, stąd $a^p + b^p \equiv a + b \pmod{p}$

Szybkie potęgowanie

Dane: $a, n \geq 1$
Wynik: a^n mod N

```
function exp(a,n) :  int;
begin
  if n = 0 return 1;
  if n mod 2 = 0 return sqr(exp(a,n/2)) mod N;
  return a*exp(a,n-1) mod N;
end
```

Złożoność: $O(\log n)$ (przy założeniu że operacje arytmetyczne wykonywane są w czasie $O(1)$.

Liczby Fibonacciego

Dane: $n > 0$
Wynik: $fib(n) \bmod N$

Niech: $fib(0) = 0$, $fib(1) = 1$, $fib(n) = fib(n-1) + fib(n-2)$ dla $n > 1$.
Można skorzystać z obserwacji:

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}^n = \begin{pmatrix} fib(n) & fib(n+1) \\ fib(n-1) & fib(n) \end{pmatrix}$$

Przy czym do obliczania n–tej potęgi macierzy można zastosować szybki algorytm potęgowania z poprzedniego punktu.

Złożoność: $O(\log n)$ (przy założeniu że operacje arytmetyczne wykonywane są w czasie $O(1)$.