Lista Nr 03, laboratorium

(wyk. 21,28 X 10 r.)

1. Algorytm square-and-multiply: Dla danej liczby rzeczywistej $x \in \mathbb{R}$, $x > 0$ i liczby naturalnej $n \in \mathbb{N}$ mamy obliczyć x^n. Dwie propozycje w jaki sposób można to zrobić:

```
Dane: n, x
Wynik: w = x^n
p ← x;
w ← 1.0;
while true do
    if 2 ∤ n then // sprawdzam czy n jest nieparzyste
        w ← w · p;
    end
    // obliczam część całkowitą z dzielenia n przez 2
    n ← n div 2;
    if n = 0 then break; // koniec obliczeń
    p ← p · p;
end
```

Algorytm 1: „Square-and-multiply”

```
Dane: n, x
Wynik: w = x^n
w ← x;
for i ← 2 to n do
    w ← w · x;
end
```

Algorytm 2: „Zwykłe” potęgowanie

Zadania:

(a) Czy w wyniku działnia obu algorytmów otrzymamy tę samą liczbę?
(b) Narysować *schemat blokowy* obu algorytmów.
(c) Napisać (*„źródło” w C*) i uruchomić programy realizujące oba algorytmy.
(d) Dla którego[1] z algorytmów należy wykonać mniej mnożeń[2] i dzieleń aby uzyskać wynik.

2. Rozszerzony algorytm Euklidesa: Dla danych liczb naturalnych $a, b \in \mathbb{N}$ wyznaczyć $p, q \in \mathbb{C}$ – całkowite o własności:

$$a \cdot p + b \cdot q = \mathrm{NWD}(a, b).$$

3. Napisać program, który będzie sprawdzał czy podana liczba naturalna jest *liczbą pierwszą*.

4. Dla danej liczby naturalnej $n \in \mathbb{N}$ napisać program, który wyznaczy jej *rozkład na czynniki pierwsze*.

5. Sito Eratostenesa Zadanie: *Dla danego $n \in \mathbb{N}$, $1 < n$ wyznaczyć wszystkie liczby pierwsze* $p \in \mathbb{N}$, $p \leq n$. Opis algorytmu:

Krok i: *począwszy od dwójki ustawiamy wszystkie liczby w ciąg rosnący* $\{2, 3, 4, \ldots, n\}$*;*

Krok ii: *bierzemy najmniejszą liczbę z ciągu (za pierwszym razem będzie to* 2*, za drugim* 3 *itd.) i wykreślamy te pozostałe wyrazy ciągu, które są jej wielokrotnością;*

Krok iii: *powtarzamy* **Krok ii**.

Po zakończeniu procesu skreślania pozostaną tylko liczby pierwsze nie większe od n.

[1] Poprzestać na $n \in \{1, 2, \ldots, 25\}$.
[2] Złożoność obliczeniowa: http://www.home.umk.pl/ abak/wdimat/s/Complex.html

6. Hipoteza Goldbacha: *każda liczba naturalna parzysta*[3] *większa od* 2 *jest sumą dwóch liczb pierwszych.* Napisać program sprawdzający *hipotezę Goldbacha* dla liczb parzystych nie większych od 1000.

7. Ciąg Fibonacciego, jest określony wzorem ($n \in \mathbb{N} \cup \{0\}$):

$$F_n = \begin{cases} 0, & n = 0 \\ 1, & n = 1 \\ F_{n-1} + F_{n-2}, & n > 1 \end{cases}$$

Napisać program wyznaczający ciąg F_n dla $n \leq 25$.

8. W *pliku tekstowym* mamy zapisanych n liczb rzeczywistych. Napisać program, który po ich wczytaniu obliczy i wyświetli na ekranie *sumę* oraz ich *średnią arytmetyczną.* Nazwę pliku należy podać jako parametr przy uruchomieniu programu.

Wrocław, dnia 16/11/2010

[3] np. $4 = 2 + 2$