Algorytmy szkolne, część 2

dr Marcin Ziółkowski

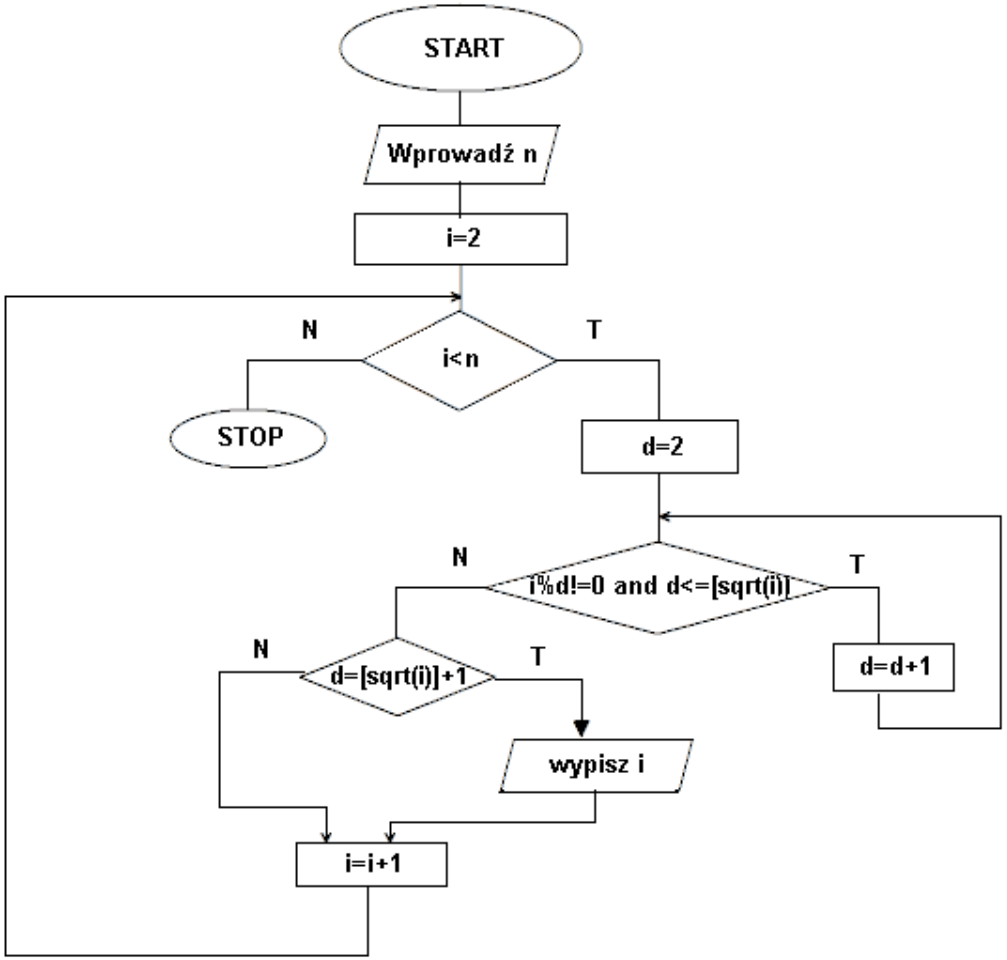
Instytut Matematyki i Informatyki
Akademia im. Jana Długosza w Częstochowie

25 kwietnia 2017 r.

WYPISYWANIE LISTY LICZB PIERWSZYCH

Zajmiemy się teraz algorytmami, za pomocą których wypisujemy liczby pierwsze. Na początek oprzemy się na znanym już nam algorytmie sprawdzania, czy liczba jest pierwsza i na jego bazie zbudujemy algorytm, który wypisuje liczby pierwsze mniejsze od wprowadzonej przez użytkownika liczby n. Algorytm wygląda następująco:

WYPISYWANIE LICZB PIERWSZYCH - ALGORYTM



WYPISYWANIE LISTY LICZB PIERWSZYCH

Program napisany w języku PYTHON realizujący ten algorytm wygląda następująco:

WYPISYWANIE LICZB PIERWSZYCH - PROGRAM 1

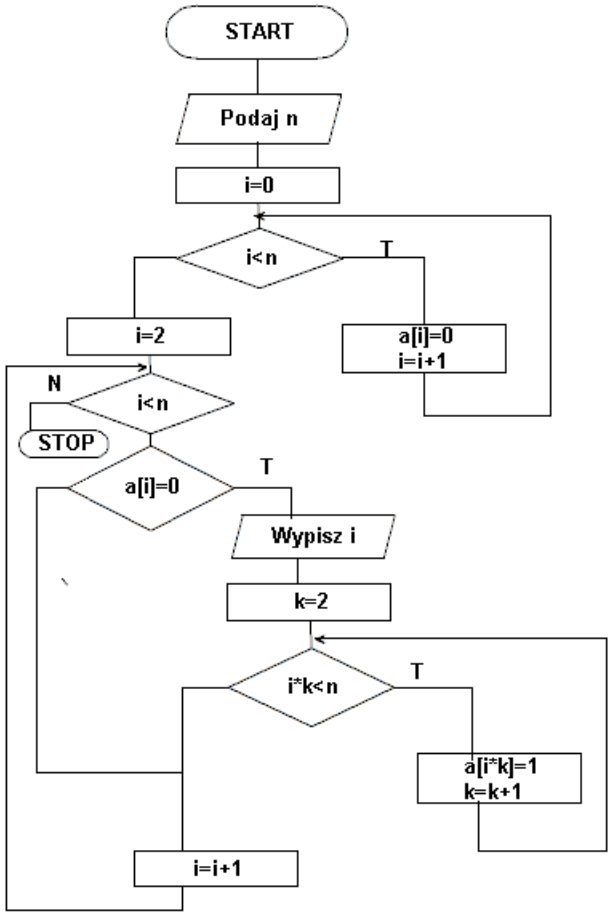
```
import math
print("Podaj n")
n=int(input())
for i in range (2,n):
    d=2
    while i%d!=0 and d<=math.floor(math.sqrt(i)):
        d=d+1
    if d==math.floor(math.sqrt(i))+1:
        print(i)
input()
```

WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

Sito Eratostenesa jest jednym z najprostszych algorytmów, za pomocą których znajdujemy liczby pierwsze. Polega on na przechodzeniu przez tablicę liczb naturalnych (od dwójki) i wykreślaniu wielokrotności liczb, które w poprzednich krokach nie były wykreślone. Liczby które pozostaną nieskreślone to liczby pierwsze. Do konstrukcji tego algorytmu najlepiej wykorzystać tablice (listy). Algorytm wygląda następująco:

WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

WYPISYWANIE LICZB PIERWSZYCH - ALGORYTM ERATOSTENESA



WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

A oto program w języku PYTHON:

WYPISYWANIE LICZB PIERWSZYCH - PROGRAM 2

```
print("Podaj n")
n=int(input())
a=[0 for i in range (n)]
for i in range (2,n):
    if a[i]==0:
        print(i)
        k=2
        while (i*k)<n:
            a[i*k]=1
            k=k+1
input()
```

WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

Program można zmodyfikować, ponieważ oczywiście wystarczy dojść do elementu $a[\lfloor\sqrt{n}\rfloor]$.

WYPISYWANIE LICZB PIERWSZYCH - PROGRAM 3

```
print("Podaj n")
n=int(input())
a=[0 for i in range (n)]
import math
for i in range (2,math.floor(math.sqrt(n))):
   if a[i]==0:
      k=2
      while (i*k)<n:
         a[i*k]=1
         k=k+1
for i in range (2,n):
   if a[i]==0:
      print(i)
input()
```

ALGORYTM HORNERA

W matematyce i innych naukach przyrodniczych mamy często do czynienia z obliczaniem wartości wielomianu w danym punkcie. Na przykład, gdy dany jest wielomian $W(x) = 4x^4 + 3x^3 + 2x^2 + 5x + 6$ i chcemy obliczyć jego wartość w punkcie $x = 2$ obliczamy:
$W(2) = 4 \cdot 2^4 + 3 \cdot 2^3 + 2 \cdot 2^2 + 5 \cdot 2 + 6 =$
$4 \cdot 16 + 3 \cdot 8 + 2 \cdot 4 + 10 + 6 = 64 + 24 + 8 + 16 = 112$. Jednak taki sposób obliczania wartości wielomianu nie jest najbardziej szybki i efektywny, gdyż musimy wykonać wiele mnożeń (w naszym przykładzie aż 10!, ponieważ obliczanie potęg również jest obliczaniem odpowiednich iloczynów). Dla wielomianów o dużych stopniach prowadzi to do długich i żmudnych obliczeń zawierających setki mnożeń, co nawet w przypadku obliczeń komputerowych może prowadzić do długiego działania programów, wykorzystujących obliczanie wartości wielomianów. Istnieje sposób bardziej efektywny, wykorzystujący algorytm Hornera, który omówimy teraz bardziej szczegółowo.

ALGORYTM HORNERA

Przedstawmy nasz wielomian w trochę inny sposób (zaczynając od najniższych potęg):

$$W(x) = 6 + 5x + 2x^2 + 3x^3 + 4x^4 = 6 + x(5 + 2x + 3x^2 + 4x^3) =$$

$$= 6 + x(5 + x(2 + 3x + 4x^2)) = 6 + x(5 + x(2 + x(3 + 4x)))$$

Taki sposób przedstawienia wielomianu powoduje, że mamy do czynienia tylko z 4 mnożeniami (dodawań jest tyle samo). Każdy wielomian można doprowadzić do takiej postaci, a sposób obliczania wartości wielomianu w oparciu o taką jego postać nazywa się algorytmem Hornera.

ALGORYTM HORNERA

Pokażemy teraz jak łatwo w oparciu o ten algorytm obliczyć wartość wielomianu $W(x) = 4x^4 + 3x^3 + 2x^2 + 5x + 6$ np. dla $x = 2$. Obliczenia przedstawimy w tabeli:

krok	1	2	3	4
wartość	4*2+3=11	11*2+2=24	24*2+5=53	53*2+6=112

Mało tego, za pomocą tego algorytmu możemy wykonywać szybkie dzielenie wielomianu przez dwumian $x - a$. Tutaj mamy:

$$(4x^4 + 3x^3 + 2x^2 + 5x + 6) : (x - 2) = 4x^3 + 11x^2 + 24x + 53 \; R \; 112$$

Inaczej można zapisać to tak:

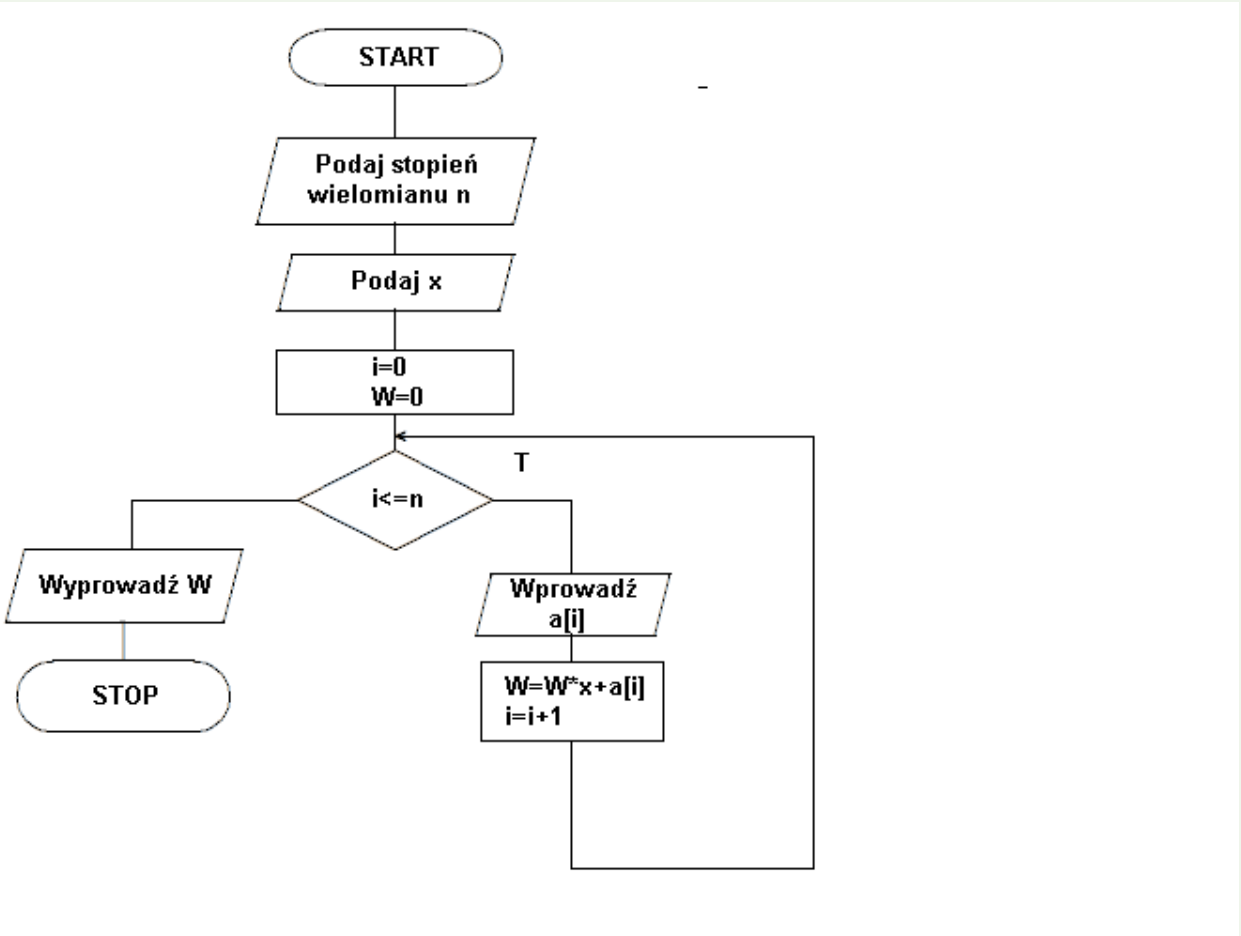
$$4x^4 + 3x^3 + 2x^2 + 5x + 6 = (4x^3 + 11x^2 + 24x + 53)(x - 2) + 112$$

Albo bardziej formalnie:

$$\frac{4x^4 + 3x^3 + 2x^2 + 5x + 6}{x - 2} = 4x^3 + 11x^2 + 24x + 53 + \frac{112}{x - 2}$$

Przejdziemy teraz do samego algorytmu Hornera i jego ilustracji z użyciem schematu blokowego. Użyjemy algorytmu iteracyjnego wykorzystującego tablice.

ALGORYTM HORNERA



ALGORYTM HORNERA - PROGRAM

```
print("Podaj stopien wielomianu")
n=int(input())
print("Podaj x")
x=float(input())
i=0
W=0
a=[0 for i in range (n+1)]
for i in range(n+1):
    a[i]=float(input("Wprowadz wsp. wielomianu:"))
    W=W*x+a[i]
print("Wartosc wielomianu wynosi", W)
input()
```

ZASTOSOWANIE ALGORYTMU HORNERA - ZAMIANA LICZBY ZAPISANEJ W INNYM SYSTEMIE LICZENIA NA SYSTEM DZIESIĘTNY

Algorytm Hornera można wykorzystać do odczytywania liczby zapisanej w innym systemie liczenia. Rozpatrzmy przykład:

$$1234_{(5)} = 1 \cdot 5^3 + 2 \cdot 5^2 + 3 \cdot 5 + 4 = 125 + 50 + 15 + 4 = 194$$

Zauważmy, że jest to również wartość wielomianu $1x^3 + 2x^2 + 3x + 4$ w punkcie $x = 5$. A oto ciekawy program (przy podstawie systemu $p \leqslant 10$):

```
print("Wprowadz liczbe")
s=input()
print("Podaj podstawe systemu")
p,W=int(input()),0
for i in range (0,len(s)):
   W=W*p+int(s[i])
print("Liczba w systemie dziesietnym to:", W)
input()
```

ZAMIANA LICZBY ZAPISANEJ W SYSTEMIE DZIESIĘTNYM NA INNY SYSTEM LICZENIA

Ogólnie znany jest algorytm zamiany liczby w systemie dziesiętnym na inny system liczenia. Wykonujemy wtedy dzielenia z resztą przez podstawę systemu, dopóki uzyskany iloraz będzie różny od zera i zapisujemy uzyskane reszty w odwrotnej kolejności. Na przykład, gdy chcemy zapisać liczbę 25 w systemie trójkowym, dzielimy:

$$25 : 3 = 8 \ r \ 1$$

$$8 : 3 = 2 \ r \ 2$$

$$2 : 3 = \mathbf{0} \ r \ 2$$

Zatem $25 = 221_{(3)}$. Na kolejnym slajdzie znajduje się program (wykorzystujący tablice (listy)), który wykonuje takie zadanie dla podstawy systemu $p \leqslant 10$. Przy "scalaniu" elementów listy, będących kolejnymi resztami, czytanymi "od końca" również został wykorzystany (w sposób celowy) algorytm Hornera. Tu sztucznie reszty traktujemy jak cyfry w systemie dziesiętnym, aby zapisać liczbę za pomocą tych cyfr.

ZAMIANA LICZBY ZAPISANEJ W SYSTEMIE DZIESIĘTNYM NA INNY SYSTEM LICZENIA

```
a=list()
print("Podaj liczbe")
n=int(input())
print("Podaj podstawe systemu")
p=int(input())
while(n!=0):
    x=n%p
    n=n//p
    a.append(x)
W=0
for i in range (len(a)-1,-1,-1):
    W=W*10+a[i]
print("Liczba zapisana w nowym systemie to:",W)
input()
```