Sito Eratostenesa

Eratostenes (Eratostenes z Cyreny) urodził się w 276 roku p.n.e, zmarł w 194 p.n.e. Był greckim uczonym, filozofem matematykiem, , astronomem, geografem oraz poetą. Jego osiągnięcia to oszacowanie średnicy Ziemi raz odległości od Słońca i Księżyca, pomiar kąta nachylenia ekliptyki do równika niebieskiego, propozycja wprowadzenia roku przestępnego, metoda znajdowania liczb pierwszych nazwana na jego cześć sitem Eratostenesa. Kierował biblioteką w Aleksandrii.

Opisaliśmy znajdowanie liczb pierwszych na podstawie ich definicji, tj. sprawdzając podzielność przez liczby mniejsze. Sposób ten nie jest najefektywniejszą metodą wyszukiwania liczb pierwszych - musimy wykonywać określoną ilość czasochłonnych dzieleń, tym większą, im większą wartość ma badana liczba.

W czasach starożytnych znano lepszą metodę opisaną przez greckiego uczonego Eratostenesa z Cyreny. Podszedł on do rozwiązania od drugiej strony - zamiast sprawdzać podzielność kolejnych liczb naturalnych przez znalezione liczby pierwsze, zaproponował wyrzucanie ze zbioru liczb naturalnych wielokrotności kolejnych liczb, które nie zostały wcześniej wyrzucone. To, co zostanie, będzie zbiorem liczb pierwszych, które nie posiadają innych podzielników jak 1 i same siebie. Metoda ta została nazwana **sitem Eratostenesa** i jest najszybszą metodą wyszukiwania liczb pierwszych w ograniczonym zbiorze.

Zobaczmy jak działa **sito Eratostenesa**. Spróbujmy wg tej metody odszukać wszystkie liczby pierwsze w zbiorze 20 początkowych liczb naturalnych.

Generacja liczb pierwszych przy pomocy sita Eratostenesa
{ 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 } Oto początkowy zbiór liczb. Najpierw usuniemy z niego liczbę 1 - nie jest to liczba pierwsza, ponieważ nie posiada dokładnie dwóch różnych podzielników.
{ 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 } Bierzemy pierwszą liczbę 2 i usuwamy ze zbioru wszystkie jej wielokrotności. W ten sposób pozbyliśmy się liczb parzystych. Zauważ iż obliczanie wielokrotności nie wymaga mnożenia - wystarczy dodawać daną liczbę.
{ 2 3 5 7 9 11 13 15 17 19 } Następną wolną liczbą jest 3. Usuwamy ze zbioru wszystkie wielokrotności liczby 3. Pozostaną więc liczby niepodzielne przez 2 i przez 3.
{ 2 3 5 7 11 13 17 19 } Wykonane - w zbiorze pozostały same liczby pierwsze.

Przed opisem algorytmu musimy zdecydować, w jaki sposób będziemy reprezentować w pamięci komputera zbiór liczb. Najprostszym rozwiązaniem wydaje się być tablica wartości logicznych. Liczbę określa indeks elementu tablicy. Wartość elementu określa z kolei, czy dana liczba jest w zbiorze czy też jej tam nie ma. Na przykład:

***t*[5] - element ten odnosi się do liczby 5.**

***t*[5] = *true* - liczba 5 jest w zbiorze.**

***t*[5] = *false* - liczba 5 została ze zbioru wyrzucona.**

Na początku wpiszemy do każdego elementu wartość true - wszystkie liczby będą w zbiorze. Następnie w elementach, których indeks jest równy wartości wyrzucanych liczb, umieścimy false. Na koniec przeglądniemy całą tablicę i wypiszemy te indeksy, dla których wskazywane elementy zawierają wciąż wartość true.

Specyfikacja problemu

Dane wejściowe

g - górny kres przedziału, w którym poszukujemy liczb pierwszych,
g należy do **N**

Dane wyjściowe

Kolejne liczby pierwsze zawarte w przedziale od 2 do g.

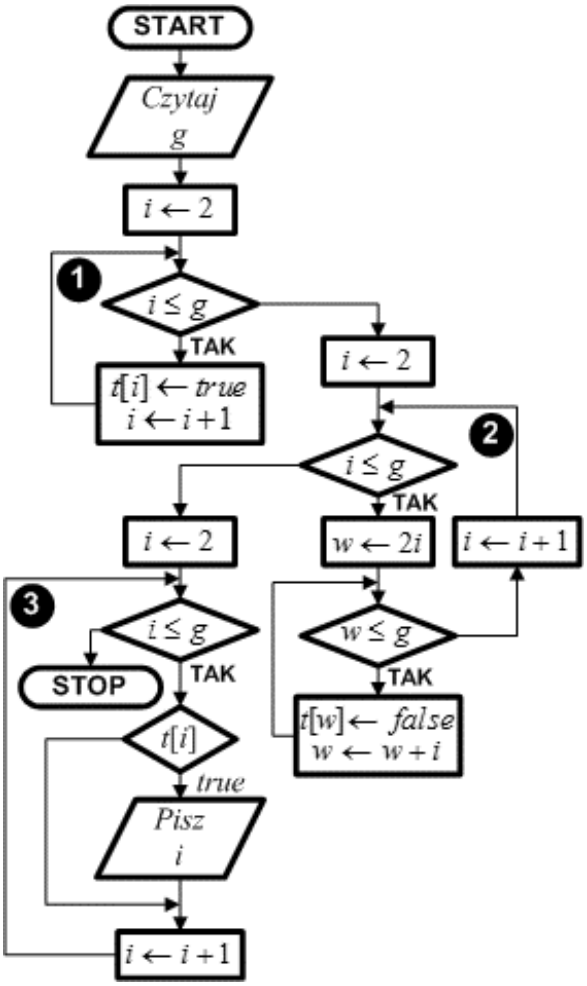
Zmienne pomocnicze

i - służy do sterowania pętlami iteracyjnymi, **i** należy do **N**
t[ ] - tablica logiczna odwzorowująca zbiór liczbowy. Indeksy elementów przebiegają wartości od 2 do **g**
w - służy do tworzenia kolejnych wielokrotności, **w** należy do **N**

Lista kroków

K01: Czytaj g
K02: Dla i = 2,3,...,g: wykonuj t[i] ← true
K03: Dla i = 2,3,...,g : wykonuj kroki K04...K07
K04: w ← 2i
K05: Dopóki w ≤ g: wykonuj kroki K06...K07.
K06: t[w] ← false
K07: w ← w + i
K08: Dla i = 2,3,...,g: jeżeli t[i] = true, to pisz i
K09: Zakończ algorytm

Schemat blokowy



Prezentowany algorytm **Sita Eratostenesa** jest maksymalnie uproszczony. Na początku odczytujemy górny kres przedziału, w którym będziemy wyznaczać liczby pierwsze umieszczając go w zmiennej *g*.

Następnie wykonywane są kolejno trzy pętle iteracyjne:

Pętla nr 1: - inicjalizacja

ustawia wszystkie elementy tablicy *t*[ ] na *true*. Zwróć uwagę, iż tablica jest ustawiana od elementu o indeksie 2. Zgodnie z tym, co powiedzieliśmy wyżej, tablica *t*[ ] odzwierciedla zbiór liczb naturalnych, w którym wartościami kolejnych liczb są indeksy elementów, natomiast zawartość tych elementów określa, czy reprezentowane przez nie liczby są (true) lub nie są (false) obecne w zbiorze.

Pętla nr 2: - eliminacja

usuwa ze zbioru wszystkie wielokrotności kolejnych liczb. Dokonujemy tego bez żadnych sprawdzeń, co oczywiście nie jest zbytnio efektywne, ale za to bardzo proste. W zmiennej *w* tworzymy pierwszą wielokrotność *i*-tej liczby. Następnie wszystkie elementy o indeksach równych kolejnym wielokrotnościom ustawiamy na *false*. Po zakończeniu pętli 2 w tablicy *t*[] wartość *true* pozostaje jedynie w elementach, których indeksy są liczbami pierwszymi.

Pętla nr 3: - prezentacja

przegląda kolejne elementy tablicy *t*[ ] i jeśli mają one wartość *true*, wyświetla ich indeks. Po wykonaniu tej pętli w oknie konsoli otrzymamy wszystkie liczby pierwsze zawarte w przedziale od 2 do *g*.