1. Wprowadzenie

Sito Erastotenesa to algorytm wyznaczający liczby pierwsze z przedziału [2;n], opracowany przez greckiego filozofa, astronoma I matematyka – Erastotenesa. Nazwa sito pochodzi od metody eliminowania liczb, przypominającą przesiewanie sitem.

2. Opis działania

Zaproponowana implementacja algorytmu zakłada stworzenie tablicy (n+1)-elementowej, zawierającej liczby z przedziału [0;n] (idealistyczna wersja tegoż algorytmu posiada tablicę (n-1)-elementową z liczbami z przedziału [2;n], lecz ze względu na przejrzystość rozwiązania, aby index w tablicy był równy wartości pola tablicy, zastosowano tablicę dłuższą o dwa pola). Następnie iterując pętlą po kolejnych liczbach całkowitych, wyszukujemy kolejnych liczb niebędących wielokrotnością liczb różnych od 1 (tj. pierwszych). Jako że pierwszą liczbą pierwszą jest 2, to od niej zaczynamy iterację. Następnie za pomocą pętli zamieniamy wszystkie wielokrotności liczby 2 w tablicy na liczbę 0. Następnie przechodzimy do następnej liczby w tablicy niebędącej zerem i analogicznie zamieniamy jej wielokrotności w tablicy na 0. Pętlę pierwszą kończymy, gdy iterator dojdzie do wartości zaokrąglonego wzwyż pierwiastka kwadratowego z n. Następnie funkcja zwraca wszystkie liczby z tablicy, niebędące zerem i większe lub równe dwa. Algorytm jest o złożoności liniowej.

3. Wizualizacja

https://www.youtube.com/watch?v=dhfhu9Q5g8U

Wizualizacja przedstawiająca działanie algorytmu sita Erastotenesa

4. Przykładowy kod w języku Python3

```
def sito(n):
        # Stworzenie tablicy zawierającej liczby z przedziału [2;n]
        arr = []
        for i in range(n+1):
            arr.append(i)


    # Główna pętla algorytmu
        for i in range(2, int(n ** 0.5) + 1):
            if arr[i] != 0:
                for y in range(i*i, n+1, i): # Pętla zaczyna się od i^2, ponieważ wszystkie mniejsze wielokrotności liczby i zostały już wykluczone
                    arr[y] = 0
        # Tworzenie tablicy zawierającej liczby != 0
        ret = []
        for i in range(2,n+1):
            if arr[i] != 0:
                ret.append(arr[i])
        return ret # zwracanie tablicy z wynikiem działania algorytmu
```

5. Inna implementacja algorytmu

Jest rozwiązania mniej czytelnie i nieoptymalne, stanowi ciekawostkę, dla zaawansowanych czytelników. Zawiera ono funkcję enumerate, o której można preczytać na stronie: https://docs.python.org/3/library/functions.html#enumerate. Oraz przekształcenie listy (list comprehension), o którym można przeczytać na stronie: https://docs.python.org/3/tutorial/datastructures.html#tut-listcomps

```
def sito_1(n):
    ret = [i for i in range(2, n+1)] # Tworzenie listy zawierającej liczby ze zbioru [2;n]
    for index, war in enumerate(ret): # Użycie funkcji enumerate i iterowanie po index'ie  i wartości z tablicy
        if war != 0:
            for y in range(index+war, len(ret), war): # Zamiana wielokrotności liczby na zera
                ret[y] = 0
    return [a for a in ret if a != 0] # Zwracanie tablicy liczb != 0
```