**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Algorytmy i struktury danych | **Zadanie** | 2.1 |
| **Autor** | Wojciech Matras | **Grupa** | WCY21KY1S1 |
| **Temat** | Sito Atkina | | |

1. Treść zadania

Zaimplementuj algorytm wyznaczania liczb pierwszych z zadanego przedziału [2,n]. (Użyj listy jednokierunkowej zaimplementowanej wskaźnikowo)

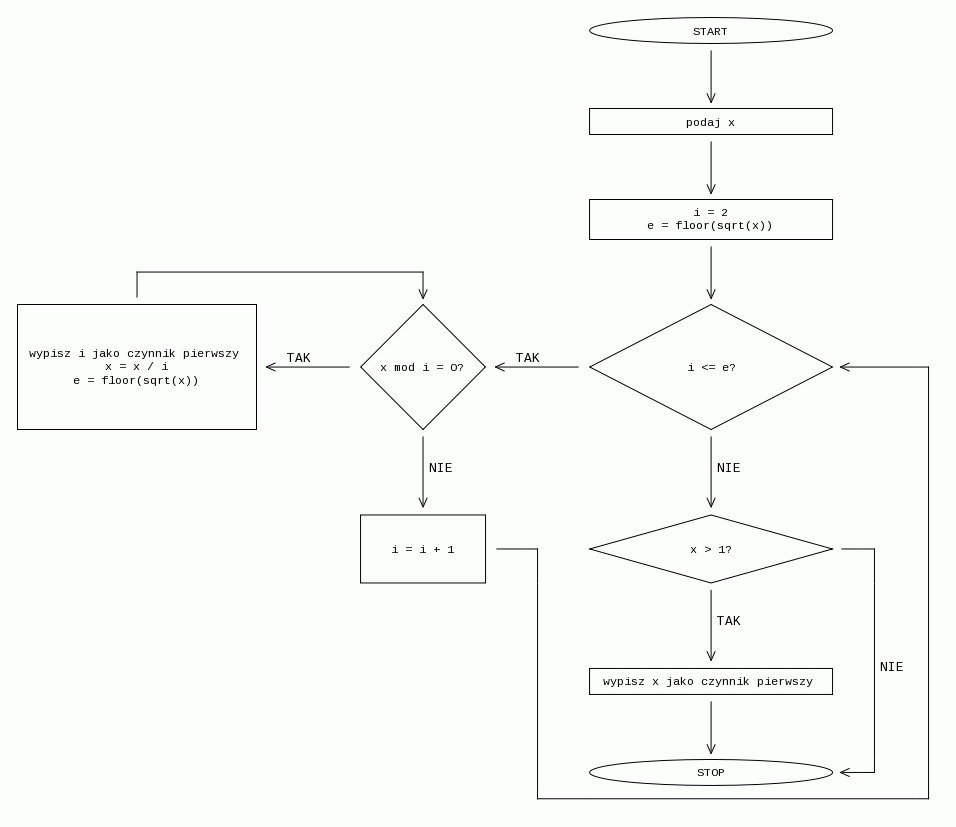
* 1. Metoda realizacji
  2. Założenia / ograniczenia dotyczące danych:
     1. Dane wejściowe:

n podawane z klawiatury

* + 1. Dane wyjściowe:

wy.TXT- plik tekstowy o nazwie wy.TXT, w którym każda liczba pierwsza zapisana jest w nowej linii.

1. Realizacja
   1. Algorytm *(lista kroków lub schemat blokowy)*



* 1. Kod źródłowy

//WCY21KY1S1 WOJCIECH MATRAS

#include<iostream>

#include<fstream> //bedziemy dzialac na plikach

using namespace std;

typedef struct Node //tworzymy liste jednokierunkowa z int'em

{

int data;

struct Node\* next;

} Node;

bool sitoAtkina(int limit, bool m[], Node\*\* pointerToHead)

{

for (int i = 0; i <= limit; i++) //zeruje cala tablice bool

m[i] = false;

// 2 i 3 sa znane jako liczby pierwsze

if (limit > 2)

m[2] = true;

if (limit > 3)

m[3] = true;

for (int x = 1; x \* x <= limit; x++) {

for (int y = 1; y \* y <= limit; y++) {

// warunek 1

int n = (4 \* x \* x) + (y \* y);

if (n <= limit

&& (n % 12 == 1 || n % 12 == 5))

m[n] ^= true;

// warunek 2

n = (3 \* x \* x) + (y \* y);

if (n <= limit && n % 12 == 7)

m[n] ^= true;

//warunek 3

n = (3 \* x \* x) - (y \* y);

if (x > y && n <= limit

&& n % 12 == 11)

m[n] ^= true;

}

}

// Zaznaczam wszystkie wielokrotnosci jako nie pierwsze

for (int r = 5; r \* r <= limit; r++) {

if (m[r]) {

for (int i = r \* r; i <= limit; i += r \* r)

m[i] = false;

}

}

// Wpisuje na liste wszystkie liczby pierwsze

cout << " \n";

for (int a = 1; a <= limit; a++)

{

if (m[a])

{

Node\* temp=(Node\*)malloc(sizeof(struct Node)); //ze w tym miejscu jest nowa liczba pierwsza

temp->data = a;

temp->next=\*pointerToHead;

if(\*pointerToHead != NULL) temp->next=\*pointerToHead;

\*pointerToHead = temp;

}

}

}

void drukuj(Node\* head) //funkcja drukujaca liste

{ int i =0;

printf("\n");

while(head != NULL) //dopoki head != NULL drukujemy wartosci listy

{

printf("%d ", head->data);

i++; // naliczamy i by program nie drukowal w jednej lini wszystkich wyrazow

if (i%7==0){ // co 7 wyraz przechodzimy na nastepna linie

printf("\n");

}

head = head->next;

}

}

void zapisdoPliku(Node \*head) { //funkcja zapisujaca liste do Pliku.txt

fstream new\_file; //tworzymy Plik

new\_file.open("new\_file\_write.txt",ios::out); //otwieramy Plik

if(!new\_file)

{

printf("\n");

cout<<"File creation failed";

}

else

{

while(head != NULL) //dopoki head != NULL wypisujemy kolejne elementy do Pliku

{

new\_file<<head->data<<" "; //wpisujemy do Pliku

head = head->next;

}

new\_file.close(); //Zamykamy nowy Plik

}

}

int main()

{ Node\* head ; //tworzymy wskaznik na glowe listy

head=NULL;

int n; //n to liczba do ktorej szukamy liczb pierwszych w zakresie od [2,n]

cout<<"Podaj zakres górny przedziału: ";

cin>>n; //wczytujemy n

bool m[n+1]; //tworzymy tablice n elementowa bool czyli z wartosciamy 0 lub 1

sitoAtkina(n, m, &head);

cout<<"Kolejne liczby pierwsze z przedziału [2.."<<n<<"]: ";

drukuj(head);

zapisdoPliku(head);

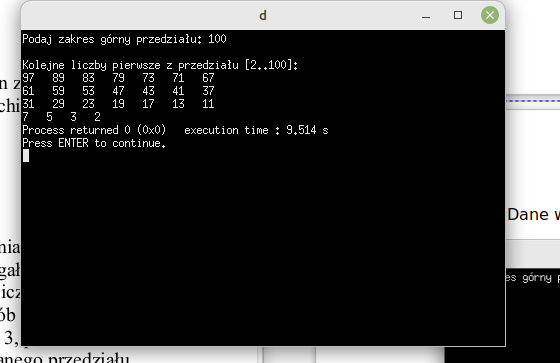
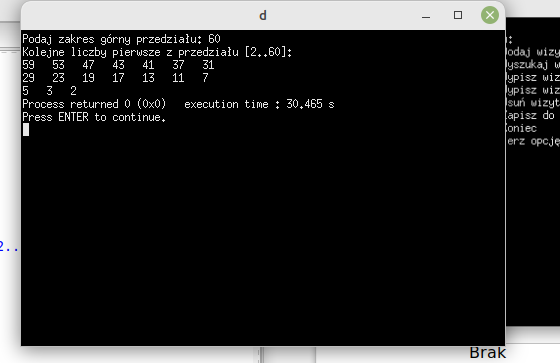
return 0;

}

* 1. Dane wejściowe *(ekran uruchomienia programu)*



* 1. Dane wyjściowe *(ekran uruchomienia programu)*



Wnioski *(złożoność asymptotyczna algorytmu przy użyciu notacji* ***O lub innej i inne wnioski)***

Sito Atkina-Bernsteina znajduje (wypisuje) wszystkie liczby pierwsze mniejsze niż N w czasie O(N) wykorzystując pamięć O(N1/2/log N).