**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Algorytmy i struktury danych | **Zadanie** | 3\_02 |
| **Autor** | Kamil Borkowski | **Grupa** | WCY22IY1S1 |
| **Temat** | Sito Atkina Bernsteina | | |

1. Treść

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

Metoda realizacji

Wypełniamy wszystkie elementy listy liczbami od 2 do n. Usuwamy te elementy z listy, które nie są liczbami pierwszymi według sita Atkina Bernsteina.

* 1. Założenia / ograniczenia dotyczące danych:
     1. Dane wejściowe *(opis)*

Górna granica przedziału – n – wczytywana z klawiatury

* + 1. Dane wyjściowe *(opis)*

Liczby pierwsze w przedziale [2,n] – wyświetlone na ekranie

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

Deklaracja struktury listy

Uzupełnienie listy poprzez tworzenie nowych elementów listy

Usunięcie elementów nie będącymi liczbami pierwszymi

Wyświetlenie wszystkich elementów listy

* 1. Kod źródłowy

//================================================

//Zadanie 3\_02 jezyk C

//Sito Atkina Bernsteina

//WCY21IY1S1 Borkowski Kamil

//================================================

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>// biblioteka umozliwiajaca uzycie bool

#include <math.h>// biblioteka umozliwiajaca uzycie pierwiastka

int main()

{

int n,x,y,xx,yy,z,i,k=2;

printf("Podaj zakres liczb pierwszych: ");

scanf("%d",&n);

bool \*lista = (bool\*) malloc((n+1) \* sizeof(bool));//zarezerwownanie potrzebnej pamieci dla listy

for(i=5;i<=n;i++)

{

\*(lista+i) = false;//poczatkowe uznanie wszystkich elementow listy za liczby niepierwsze(true=liczba pierwsza,false=liczba niepierwsza)

}

for(x=1;x<=sqrt(n);x++)

{

xx = x\*x;

for(y=1;y<=sqrt(n);y++)

{

yy = y\*y;

z = (xx<<2)+yy; //XX<<2 - przesuniecie xx o 2 bity

if((z<=n) && ((z%12==1)||(z%12==5)))

{

lista[z] = !lista[z]; //zmiana true na false i vice versa

}

z -= xx;

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

{

lista[z] = !lista[z];//zmiana true na false i vice versa

}

if(x>y)

{

z -= yy<<1;

if((z<=n) && (z%12==11))

{

lista[z] = !lista[z];//zmiana true na false i vice versa

}

}

}

}

for(i=5;i<=sqrt(n);i++)

if(\*(lista+i)==true)

{

xx = i\*i;

z = xx;

while(z<=n)

{

lista[z] = false;//nadanie false liczbom niepierwszym

z += xx;

}

}

printf("Liczby pierwsze w przedziale [2,%d] to:\n", n);

printf("2 3 ");

for(i=5;i<=n;i++)

{

if(k==20)

{

printf("\n");//wypisujemy po 20 liczb w wierszu dla przejrzystosci

k=0;

}

if(\*(lista+i)==true)

{

printf("%d ",i);//wypisujemy wszystkie liczby pierwsze

k++;

}

}

free(lista);//zwalniamy pamiec

return 0;

}

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

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

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

Obraz zawierający tekst, elektronika, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

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

Złożoność obliczeniowa programu wynosi: O(n), ponieważ najbardziej złożone obliczeniowo jest wykonanie utworzenie wszystkich elementów listy.