Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження лінійних алгоритмів»

Варіант 12

Виконав студент ІП-13 Дойчев Костянтин Миколайович
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота №5

Тема: Дослідження складних циклічних алгоритмів

Мета – дослідити особливості роботи складних циклів та набути практичних навичок їх використання під час складання програмних специфікацій.

Варіант 12

1) Постановка задачі:

Дано натуральне число n. Визначити всі натуральні числа, менші за n і взаємно прості з ним.

Розв'язання

Крок 1. Визначимо основні дії
Крок 2. Перебір натуральних чисел number менших за n

Крок 3. Перебір натуральних чисел j менших за number

Крок 4. Знаходження чи являється число j дільником number

Крок 5. Знаходження чи являється число j дільником n

Крок 6. Вивід взаємно простого числа

2) Побудова математичної моделі:

Таблиця імен змінних

Змінна	Тип	Ім'я	Призначення

Натуральне число яке вводить користувач	Цілий	n	Вхідні дані
Натуральне число-лічильник	Цілий	number	Вихідні дані
Натуральне число-лічильник, яке являється дільником	Цілий	j	Проміжні дані
Чи є спільний дільник	Булевий	haveSameDivisor	Проміжні дані
Чи є не спільний дільник	Булевий	haveDifferentDivisor	Проміжні дані

Таким чином, математичне формулювання задачі зводиться до знаходження числа(number), яке має один спільний дільник з числом n, а отже вони взаємно прості. Для цього використаємо вкладені цикли та умовні оператори. Числа n, number та j мають бути > 1. Через те, що у структурному програмуванні не бажано користуватися операторами break та continue, будемо використовувати так звані змінні-прапори, які даватимуть інформацію про те, чи є у 2 чисел спільні дільники окрім 1.
Починати лічильники будемо з 2, через те, що 1 і так є дільником всіх чисел, тому починати з неї - не раціонально.

3) Псевдокод алгоритму

Крок 1:

Початок

Введення даних
Перебір натуральних чисел number менших за n
Перебір натуральних чисел j менших за number
Знаходження чи являється число j дільником number

Знаходження чи являється число j дільником n
Вивід взаємно простого числа
Кінець

Крок 2:

Початок
Введення даних
для number **від** 2 **до** n **з кроком** 1 **повторити**
Перебір натуральних чисел j менших за number
Знаходження чи являється число j дільником number
Знаходження чи являється число j дільником n
Вивід взаємно простого числа
все повторити
Кінець

Крок 3:

Початок
Введення даних
для number **від** 2 **до** n **з кроком** 1 **повторити**
haveSameDivisor:= false;
haveDifferentDivisor:= false;
для j **від** 2 **до** number **з кроком** 1 **повторити**
Знаходження чи являється число j дільником number
Знаходження чи являється число j дільником n
все повторити
Вивід взаємно простого числа
все повторити
Кінець

Крок 4:

Початок
Введення даних

для number **від** 2 **до** n **з кроком** 1 **повторити**
haveSameDivisor:= false;
haveDifferentDivisor:= false;
для j **від** 2 **до** number **з кроком** 1 **повторити**
якщо number % j == 0
то
Знаходження чи являється число j дільником n
все якщо
все повторити
Вивід взаємно простого числа
все повторити

Кінець

Крок 5:

Початок
Введення даних
для number **від** 2 **до** n **з кроком** 1 **повторити**
haveSameDivisor:= false;
haveDifferentDivisor:= false;
для j **від** 2 **до** number **з кроком** 1 **повторити**
якщо number % j == 0
то
якщо n % j == 0
то
haveSameDivisor:=true
інакше
haveDifferentDivisor:=true
все якщо
все якщо
все повторити
Вивід взаємно простого числа
все повторити
Кінець

Крок 6:

Початок

Введення даних

для number **від** 2 **до** n **з кроком** 1 **повторити**

haveSameDivisor:= false;

haveDifferentDivisor:= false;

для j **від** 2 **до** number **з кроком** 1 **повторити**

якщо number % j == 0

то

якщо n % j == 0

то

haveSameDivisor:=true

інакше

haveDifferentDivisor:=true

все якщо

все якщо

все повторити

якщо !haveSameDivisor && haveDifferentDivisor

то

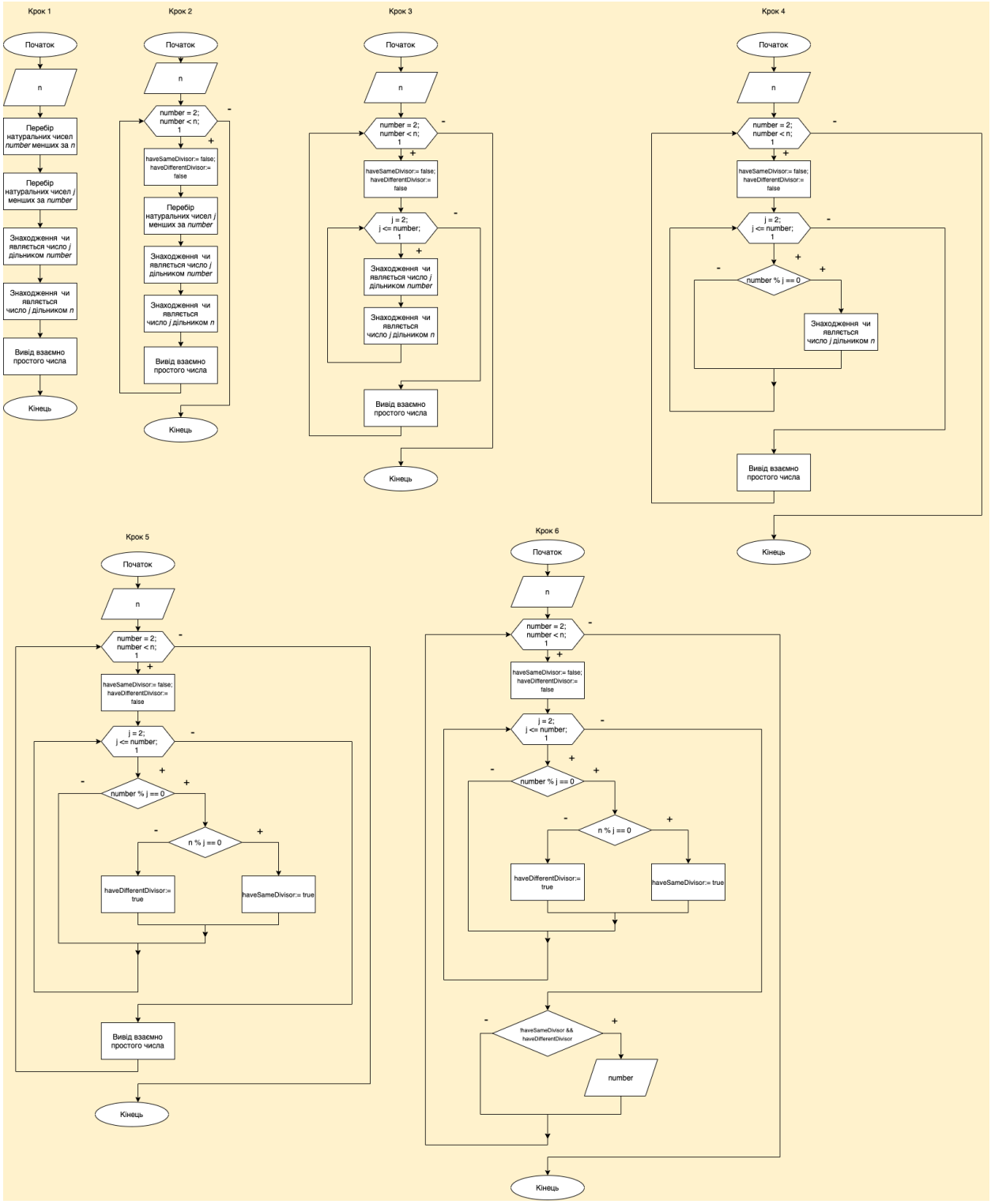
Виведення number

все якщо

все повторити

Кінець

4) Блок схема алгоритму


Крок 1
Початок
n
Перебір натуральних чисел number менших за n
Перебір натуральних чисел j менших за number
Знаходження чи являється число j дільником number
Знаходження чи являється число j дільником n
Вивід взаємно простого числа
Кінець
Крок 2
Початок
n
number = 2; number < n; 1
haveSameDivisor:= false; haveDifferentDivisor:= false
Перебір натуральних чисел j менших за number
Знаходження чи являється число j дільником number
Знаходження чи являється число j дільником n
Вивід взаємно простого числа
Кінець
Крок 3
Початок
n
number = 2; number < n; 1
haveSameDivisor:= false; haveDifferentDivisor:= false
j = 2; j <= number; 1
Знаходження чи являється число j дільником number
Знаходження чи являється число j дільником n
Вивід взаємно простого числа
Кінець
Крок 4
Початок
n
number = 2; number < n; 1
haveSameDivisor:= false; haveDifferentDivisor:= false
j = 2; j <= number; 1
number % j == 0
Знаходження чи являється число j дільником n
Вивід взаємно простого числа
Кінець
Крок 5
Початок
n
number = 2; number < n; 1
haveSameDivisor:= false; haveDifferentDivisor:= false
j = 2; j <= number; 1
number % j == 0
n % j == 0
haveDifferentDivisor:= true
haveSameDivisor:= true
Вивід взаємно простого числа
Кінець
Крок 6
Початок
n
number = 2; number < n; 1
haveSameDivisor:= false; haveDifferentDivisor:= false
j = 2; j <= number; 1
number % j == 0
n % j == 0
haveDifferentDivisor:= true
haveSameDivisor:= true
!haveSameDivisor && haveDifferentDivisor
number
Кінець

5) Випробування алгоритму:

Блок	Дія
	Початок
1	n:= 5;
2	number:= 2; number < n == true;
3	haveSameDivisor:= false; haveDifferentDivisor:= false;
4	j = 2; j <= number == true;
5	(number % j == 0) == true
6	((n % j) == 0) == false
7	haveDifferentDivisor:= true;
8	!haveSameDivisor && haveDifferentDivisor == true
9	j = 3; j <= number == false;
10	Вивід 2
11	number:= 3; number < n == true;
12	haveSameDivisor:= false; haveDifferentDivisor:= false;
13	j = 2; j <= number == true;
14	(number % j == 0) == false;
15	j = 3; j <= number == true;
16	(number % j == 0) == true
17	((n % j) == 0) == false
18	haveDifferentDivisor:= true;
19	!haveSameDivisor && haveDifferentDivisor == true

20	j = 4; j <= number == false;
21	Вивід 3
22	number:= 4; number < n == true;
23	haveSameDivisor:= false; haveDifferentDivisor:= false;
24	j = 2; j <= number == true;
25	(number % j == 0) == true;
26	((n % j) == 0) == false
27	haveDifferentDivisor:= true;
28	j = 3; j <= number == true;
29	(number % j == 0) == false;
30	j = 4; j <= number == true;
31	(number % j == 0) == true;
32	((n % j) == 0) == false
33	haveDifferentDivisor:= true;
34	j = 5; j <= number == false;
35	!haveSameDivisor && haveDifferentDivisor == true
36	Вивід 4
37	number:= 5; number < n == false;
	Кінець

6) Виновки:

Дослідив особливості роботи складних циклів та набув практичних навичок їх використання під час складання програмних специфікацій. Побудував

математичну модель, псевдокод та блок схему для вирішення задачі.
Протестував алгоритм.