Алгоритми та структури данних. Основи алгоритмів розгалуження.

*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 № 5 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження складних циклічних алгоритмів»

Варіант 15

Виконав студент ІП-12, Кириченко Владислав Сергійович

(шифр, прізвище, ім'я, по батькові)

Перевірив

( прізвище, ім'я, по батькові)

Київ 202 1

**Лабораторна робота № 5**

**Назва роботи**: Дослідження складних циклічних алгоритмів

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

**Варіант 15**

**Умова задачі***:*

Дано цілі числа p і q. Визначити всі дільники числа p, взаємно прості з q.

**Постановка задачі**: Задано змінні ***p*** та ***q***, знайти всі дільники числа p, взаємно прості з q. Результатом розв’язку задачі є ряд чисел**.**

**Побудова математичної моделі:**

*Розіб’ємо задачу на два етапи:*

**1.**Знаходження дільників числа **p.**

**2.**Знаходження серед дільників числа **p** чисел взаємнопростих з **q.**

Перший етап реалізуємо за допомогою арифметичного циклу.

Другий етап реалізуємо за допомогою алгоритму Евкліда, описаного ітераційним циклом.

*Пояснення другого етапу:*

Для знаходження взаємнопростих з деяким числом чисел з деякого ряду потрібно перевірити на цю властівість кожний з членів ряду. Тобто потрібен алгоритм перевірки чи є два числа взаємнопростими.

Два числа є взаємнопротими якщо ії НОД (найбільший спільний дільник дорівнює 1). Тобто потрібно знайти НОД двух чисел і перевірити чи це число дорівнює одиниці.

Найпростіший у реалізації метод знаходження НОД - алгоритм Евкліда, його і використаємо.

*Псевдокод алгоритму Евкліда:*

**поки а**!=0 & **b**!=0

**якщо** a > b

**то a** %= **b**

**інакше b** %= **a**

**виведення** (a+b)

Щоб виконання алгоритму Евкліда не впливало на лобальні значення змінних, що перевіряються, скористаємося тимчасовими зміннми a=i та b=q.

Складемо таблицю змінних:

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Перша змінна | Цілочисельний | ***p*** | Початкові дані |
| Друга змінна | Цілочисельний | ***q*** | Початкові дані |
| Лічильник | Натуральний | ***i*** | Проміжкове значення |
| Тимчасова змінна для перевірки чи число з ряду дільників **p** є взаємнопростим із **q**.(i) | Цілочисельний | ***a*** | Проміжкове значення |
| Тимчасова змінна для перевірки чи число з ряду дільників **p** є взаємнопростим із **q**.(q) | Цілочисельний | ***b*** | Проміжкове значення |
| Значення НОД змінних | Цілочисельний | ***biggestCommonDivisor*** | Проміжкове значення |
| Дільник числа p | Цілочисельний | ***pDivisor*** | Проміжкове значення |

*3.*Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

**Крок 1.** Визначимо основні дії.

**Крок 2.** Деталізація арифметичного циклу, який перебирає всі натуральні значення, які менша за **p**.

**Крок 3.**Деталізація перевірки чи є число **i** дільником **p**.

**Крок 4.**Деталізація ітераційного циклу, який обумовлює реалізацію алгоритму Евкліда

**Крок 5.** Деталізація знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)

**Крок 6.** Деталізація перевірки чи НСД чисел **і** та **q** - 1 (чи є число **i** одним із шуканих значень).

*Псевдокод:*

*Крок 1.*

**початок**

введення ***p,q***

арифметичний цикл, який перебирає всі натуральні значення, які менші за **p**.

перевірка чи є число **i** дільником **p**.

ітераційний цикл, який обумовлює реалізацію алгоритму Евкліда

знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)

перевірка чи НСД чисел **і** та **q** - 1 (чи є число **i** одним із шуканих значень).

**кінець**

*Крок 2.*

**початок**

введення ***p,q***

**повторити**

**для і від** 1 **до p**+1

перевірка чи є число i дільником p.

ітераційний цикл, який обумовлює реалізацію алгоритму Евкліда

знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)

перевірка чи НСД чисел **і** та **q** - 1 (чи є число **i** одним із шуканих значень).

**все повторити**

**кінець**

*Крок 3.*

**початок**

введення ***p,q***

**повторити**

**для і від** 1 **до p**+1

**якщо p** % **i** == 0  
 **то**

**pDivisor = i**

ітераційний цикл, який обумовлює реалізацію алгоритму Евкліда

знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)

перевірка чи НСД чисел **і** та **q** - 1 (чи є число **i** одним із шуканих значень).

**все якщо**

**все повторити**

**кінець**

*Крок 4.*

**початок**

введення ***p,q***

**повторити**

**для і від** 1 **до p**+1

**якщо p** % **i** == 0  
 **то**

**pDivisor = i**

**a = pDivisor**

**b = q**

**поки (a!=0 & b!=0) повторити**

знаходження НСД кожного дільника числа p і числа q.(алгоритм Евкліда)

**все поки**

перевірка чи НСД чисел **і** та **q** - 1 (чи є число **i** одним із шуканих значень).

**все якщо**

**все повторити**

**кінець**

*Крок 5.*

**початок**

введення ***p,q***

**повторити**

**для і від** 1 **до p**+1

**якщо p** % **i** == 0  
 **то**

**pDivisor = i**

**a = pDivisor**

**b = q**

**поки (a!=0 & b!=0) повторити**

**якщо a>b**

**то**

**а%=b**

**інакше**

**b %= a**

**все поки**

**biggestCommonDivisor = a+b**

перевірка чи НСД чисел і та **q** - 1 (чи є число i одним із шуканих значень).

**все якщо**

**все повторити**

**кінець**

*Крок 6.*

**початок**

введення ***p,q***

**повторити**

**для і від** 1 **до p**+1

**якщо p** % **i** == 0  
 **то**

**pDivisor = i**

**a = pDivisor**

**b = q**

**поки (a!=0 & b!=0) повторити**

**якщо a>b**

**то**

**а%=b**

**інакше**

**b %= a**

**все поки**

**biggestCommonDivisor = a+b**

**якщо biggestCommonDivisor == 0**

**то**

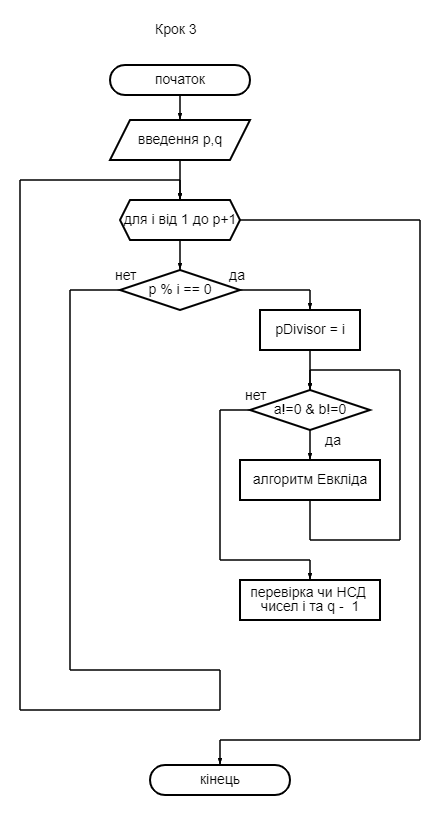
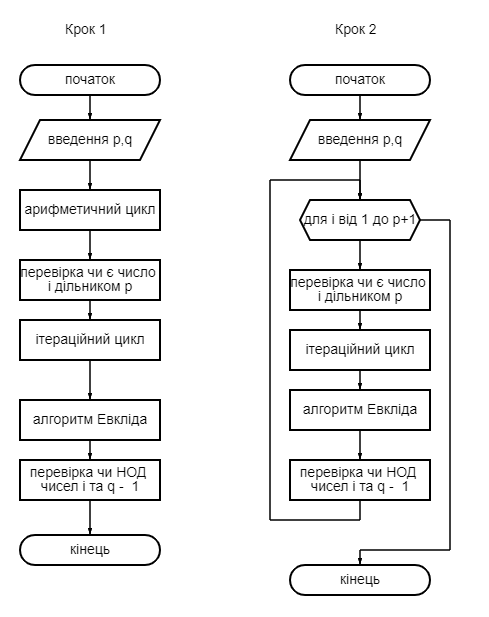
виведення **pDivisor**

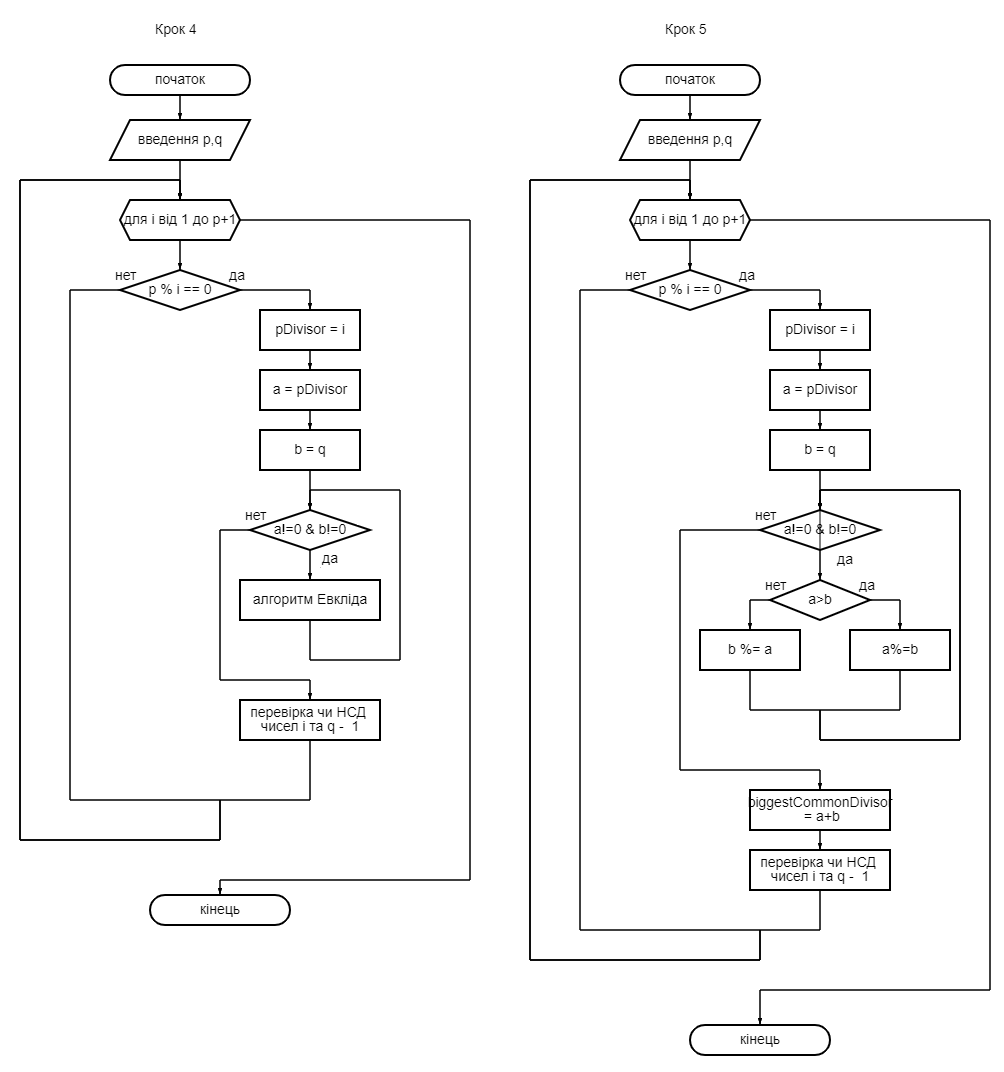
**все якщо**

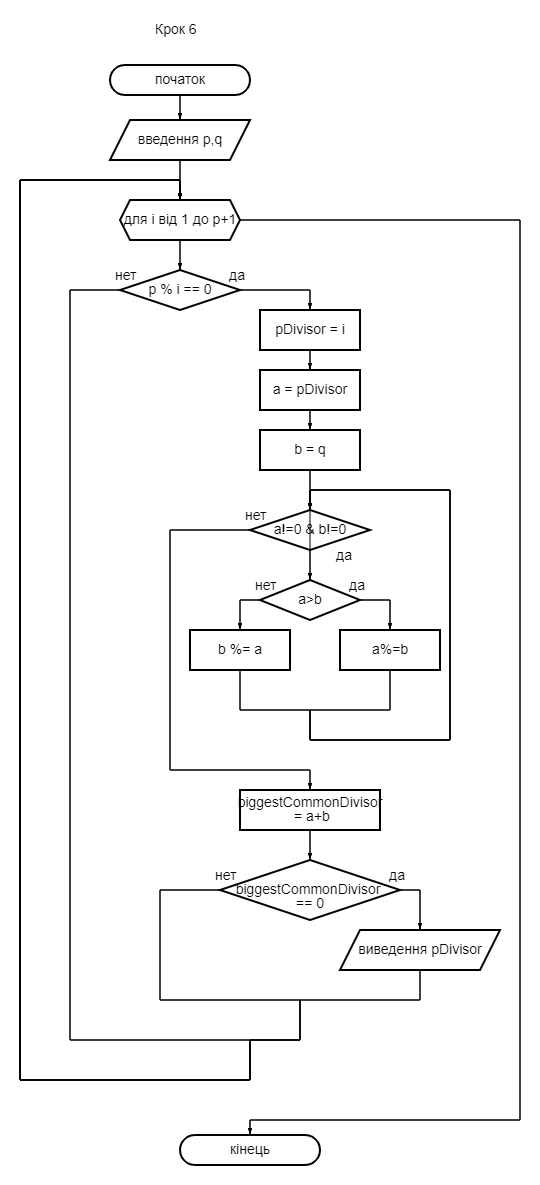
**все повторити**

**кінець**

Блок схема:







1. Перевірка алгоритму

|  |  |  |
| --- | --- | --- |
| Блок | Дія | Дія |
|  | **Початок** | **Початок** |
| 1 | Введення  ***x***=36, ***n***=13, | Введення  ***x***=67,***n*=**12, |
| 2 | виведення:1,2,3,4,6,9,12,18,36 | виведення:67,12 |
| 3 | **Кінець** | **Кінець** |

**Висновок -** Було досліджено особливості роботи складних циклів та набуто практичних навичок їх використання під час складання програмних специфікацій.