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**

**все якщо**

**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 повторити**

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 повторити**

**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 повторити**

**biggestCommonDivisor = a+b**

**якщо biggestCommonDivisor == 0**

**то**

виведення **pDivisor**

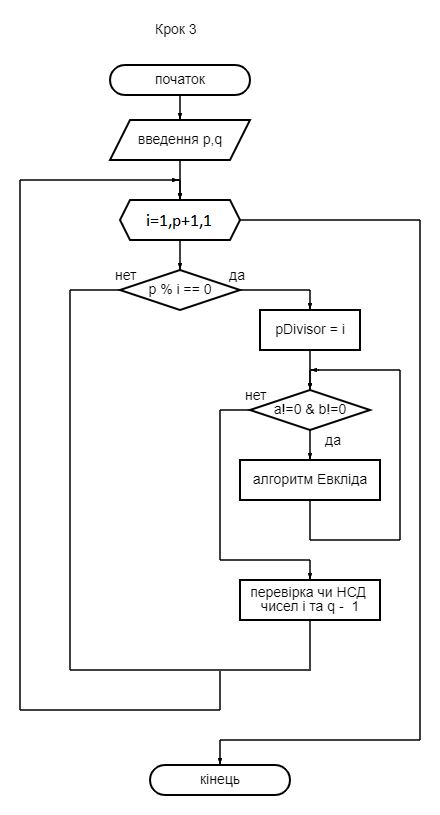
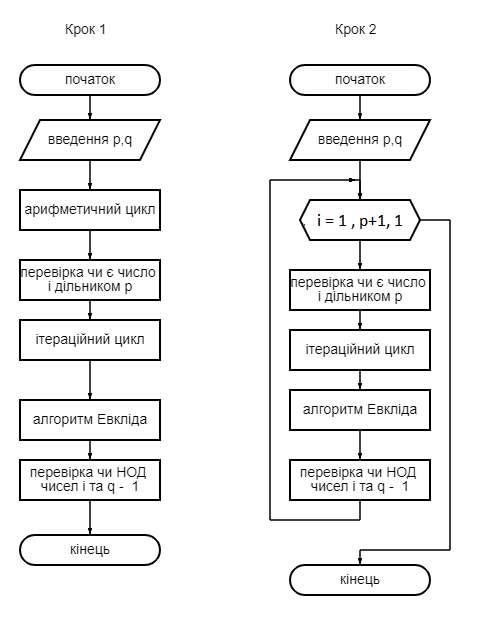
**все якщо**

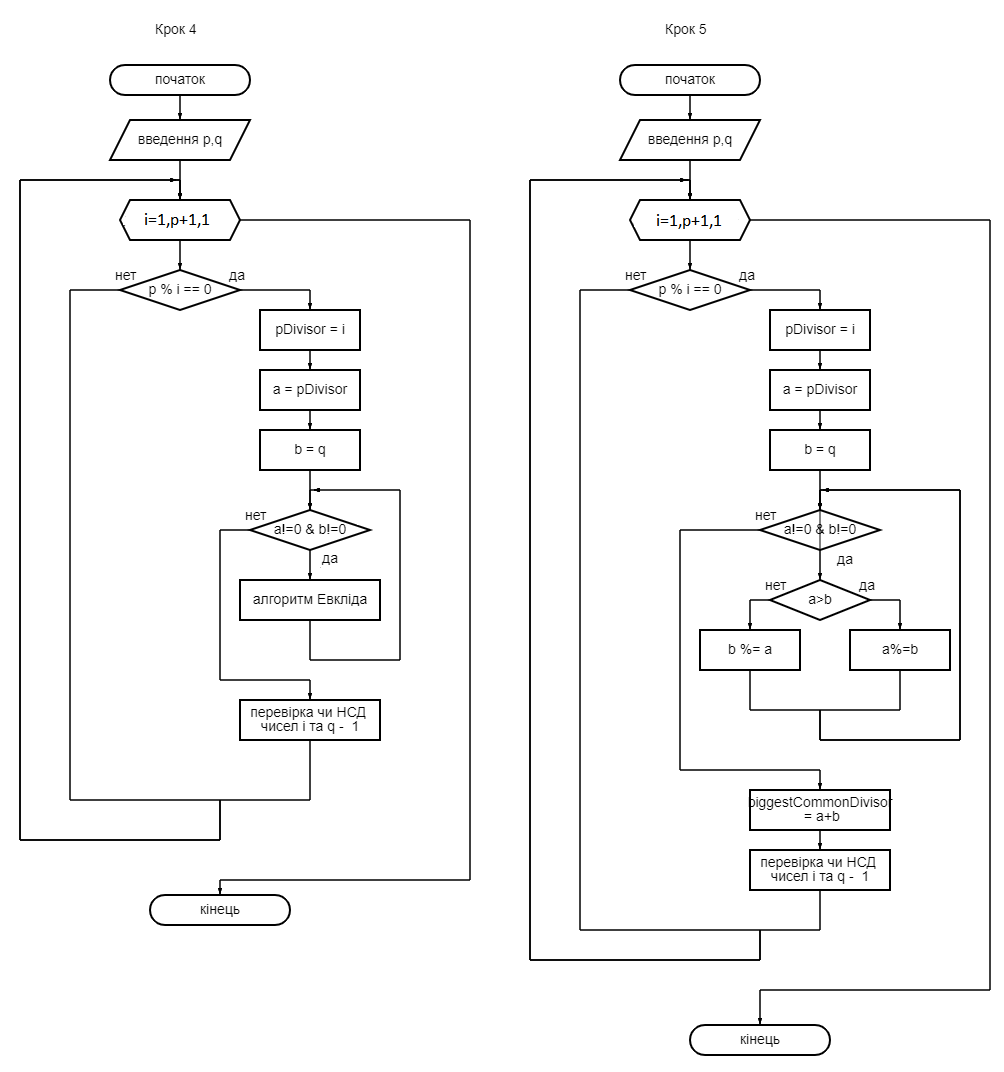
**все якщо**

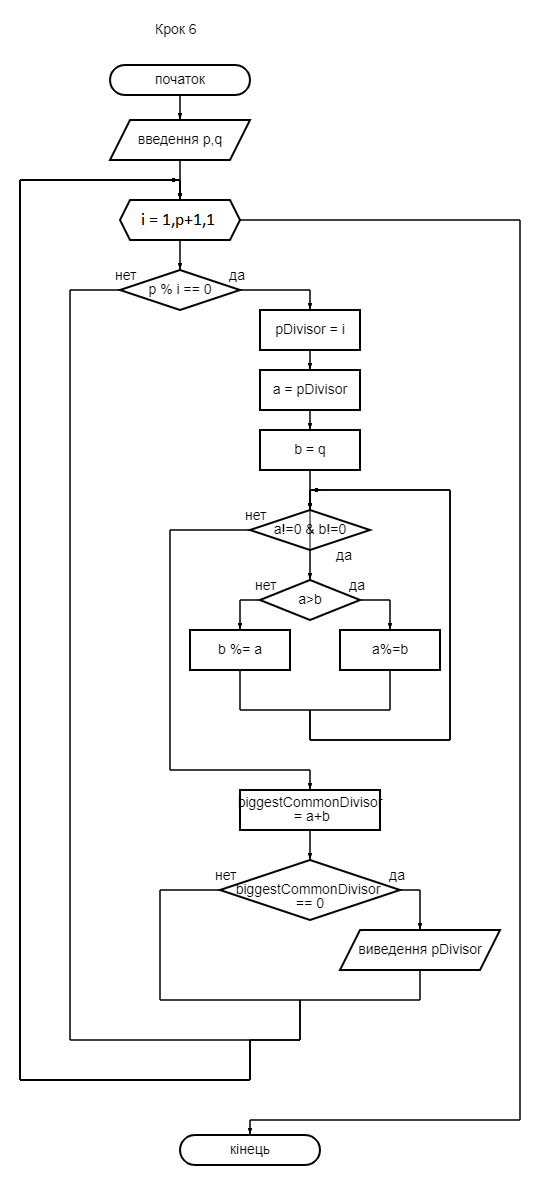
**все повторити**

**кінець**

Блок схема:







1. Перевірка алгоритму

|  |  |
| --- | --- |
| Блок | Дія |
|  | **Початок** |
| 1 | Введення  ***x***=16, ***n***=7, |
| 2 | **iteration: 1** |
| 3 | **(p % i == 0) = true** |
| 4 | **pDivisor = 1** |
| 5 | **a = 1** |
| 6 | **b = 7** |
| 7 | **Euclid algorithm** |
| 8 | **(a != 0 && b != 0) = true** |
| 9 | **a<b = false** |
| 10 | **a = 1** |
| 11 | **b = 0** |
| 12 | **biggestCommonDivisor = 1** |
| 13 | **( biggestCommonDivisor == 1) = true** |
| 14 | **виведення 1** |
| 15 | **iteration: 2** |
| 16 | **(p % i == 0) = true** |
| 17 | **pDivisor = 2** |
| 18 | **a = 2** |
| 19 | **b = 7** |
| 20 | **Euclid algorithm** |
| 21 | **(a != 0 && b != 0) = true** |
| 22 | **a<b = false** |
| 23 | **a = 2** |
| 24 | **b = 1** |
| 25 | **(a != 0 && b != 0) = true** |
| 26 | **a<b = true** |
| 27 | **a = 0** |
| 28 | **b = 1** |
| 29 | **biggestCommonDivisor = 1** |
| 30 | **( biggestCommonDivisor == 1) = true** |
| 31 | **виведення 2** |
| 32 | **iteration: 3** |
| 33 | **(p % i == 0) = false** |
| 34 | **iteration: 4** |
| 35 | **(p % i == 0) = true** |
| 36 | **pDivisor = 4** |
| 37 | **a = 4** |
| 38 | **b = 7** |
| 39 | **Euclid algorithm** |
| 40 | **(a != 0 && b != 0) = true** |
| 41 | **a<b = false** |
| 42 | **a = 4** |
| 43 | **b = 3** |
| 44 | **(a != 0 && b != 0) = true** |
| 45 | **a<b = true** |
| 46 | **a = 1** |
| 47 | **b = 3** |
| 48 | **(a != 0 && b != 0) = true** |
| 49 | **a<b = false** |
| 50 | **a = 1** |
| 51 | **b = 0** |
| 52 | **biggestCommonDivisor = 1** |
| 53 | **( biggestCommonDivisor == 1) = true** |
| 54 | **виведення 4** |
| 55 | **iteration: 5** |
| 56 | **(p % i == 0) = false** |
| 57 | **iteration: 6** |
| 58 | **(p % i == 0) = false** |
| 59 | **iteration: 7** |
| 60 | **(p % i == 0) = false** |
| 61 | **iteration: 8** |
| 62 | **(p % i == 0) = true** |
| 63 | **pDivisor = 8** |
| 64 | **a = 8** |
| 65 | **b = 7** |
| 66 | **Euclid algorithm** |
| 67 | **(a != 0 && b != 0) = true** |
| 68 | **a<b = true** |
| 69 | **a = 1** |
| 70 | **b = 7** |
| 71 | **(a != 0 && b != 0) = true** |
| 72 | **a<b = false** |
| 73 | **a = 1** |
| 74 | **b = 0** |
| 75 | **biggestCommonDivisor = 1** |
| 76 | **( biggestCommonDivisor == 1) = true** |
| 77 | **виведення 8** |
| 78 | **iteration: 9** |
| 79 | **(p % i == 0) = false** |
| 80 | **iteration: 10** |
| 81 | **(p % i == 0) = false** |
| 82 | **iteration: 11** |
| 83 | **(p % i == 0) = false** |
| 84 | **iteration: 12** |
| 85 | **(p % i == 0) = false** |
| 86 | **iteration: 13** |
| 87 | **(p % i == 0) = false** |
| 88 | **iteration: 14** |
| 89 | **(p % i == 0) = false** |
| 90 | **iteration: 15** |
| 91 | **(p % i == 0) = false** |
| 92 | **iteration: 16** |
| 93 | **(p % i == 0) = true** |
| 94 | **pDivisor = 16** |
| 95 | **a = 16** |
| 96 | **b = 7** |
| 97 | **Euclid algorithm** |
| 98 | **(a != 0 && b != 0) = true** |
| 99 | **a<b = false** |
| 100 | **a = 2** |
| 101 | **b = 7** |
| 102 | **(a != 0 && b != 0) = true** |
| 103 | **a<b = false** |
| 104 | **a = 2** |
| 105 | **b = 1** |
| 106 | **(a != 0 && b != 0) = true** |
| 107 | **a<b = true** |
| 108 | **a = 0** |
| 109 | **b = 1** |
| 110 | **biggestCommonDivisor = 1** |
| 111 | **( biggestCommonDivisor == 1) = true** |
| 112 | **виведення 16** |
|  | **кінець** |

(1,2,4,8,16)

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