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 властивість кожний з членів ряду. Тобто потрібен алгоритм перевірки чи є два числа взаємнопростими.

Два числа є взаємнопротими якщо її 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 a!=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 змінними 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 **i** 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 **і** дільником **p**.
  ітераційний цикл, який обумовлює реалізацію алгоритму Евкліда
  знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)
  перевірка чи НСД чисел **i** 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 і дільником р.
    ітераційний цикл, який обумовлює реалізацію алгоритму Евкліда
    знаходження НСД кожного дільника числа **p** і числа **q**.(алгоритм Евкліда)
    перевірка чи НСД чисел **i** 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 i та q -  1 (чи є число i одним із
                    шуканих значень).
          все якщо
   все повторити
кінець
```

Крок 4.

```
початок
   введення p,q
   повторити
          для i 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 i та q -  1 (чи є число i одним із
                    шуканих значень).
          все якщо
   все повторити
кінець
```

Крок 5.

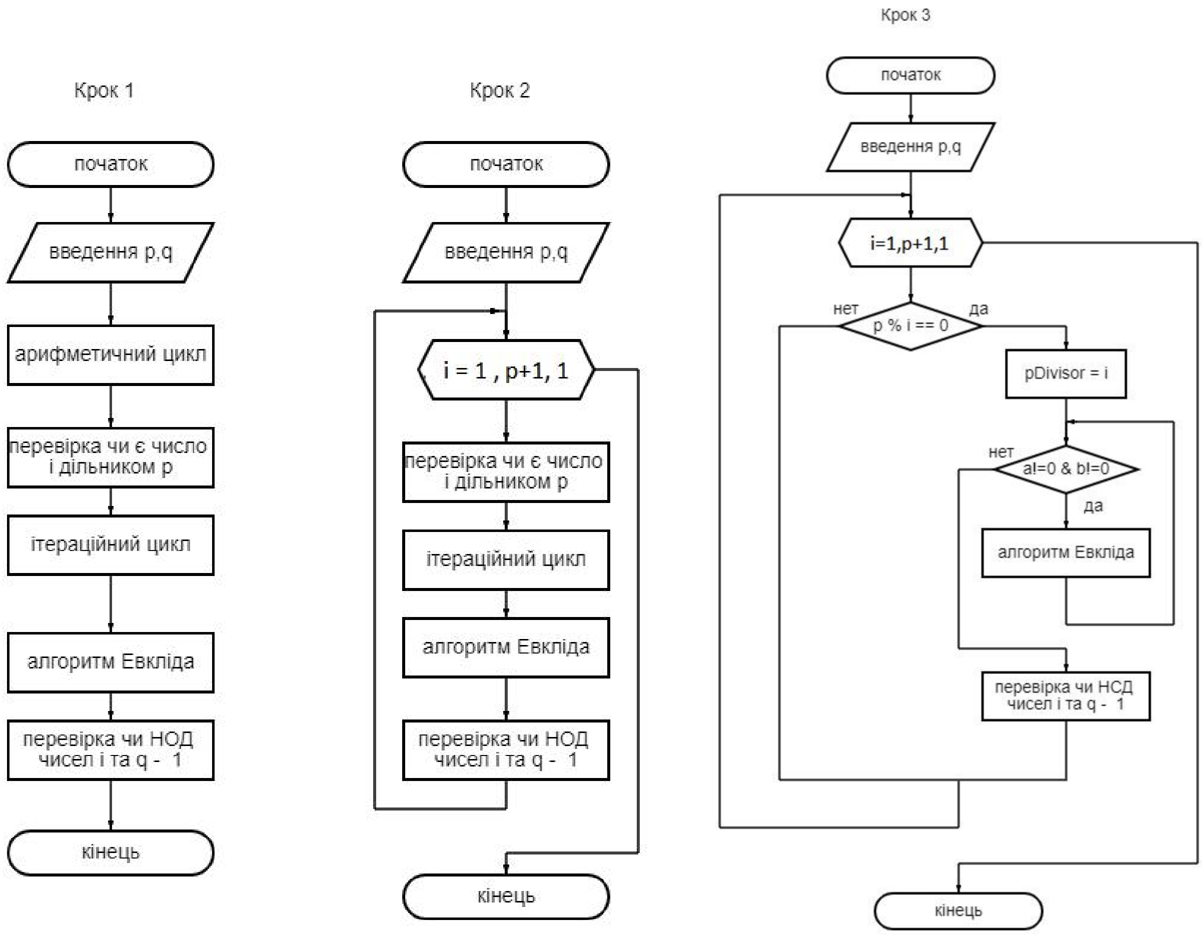
```
початок
   введення p,q
   повторити
          для i від 1 до p+1
          якщо p % i == 0
              то
                    pDivisor = i
                    a = pDivisor
                    b = q
                    поки (a!=0 & b!=0) повторити
                           якщо a>b
                                  то
```

```
                                        a%=b
                              інакше
                                        b %= a
                         все поки
                         biggestCommonDivisor = a+b
                         перевірка чи НСД чисел і та q -  1 (чи є число і одним із
                         шуканих значень).
               все якщо
   все повторити
кінець
```

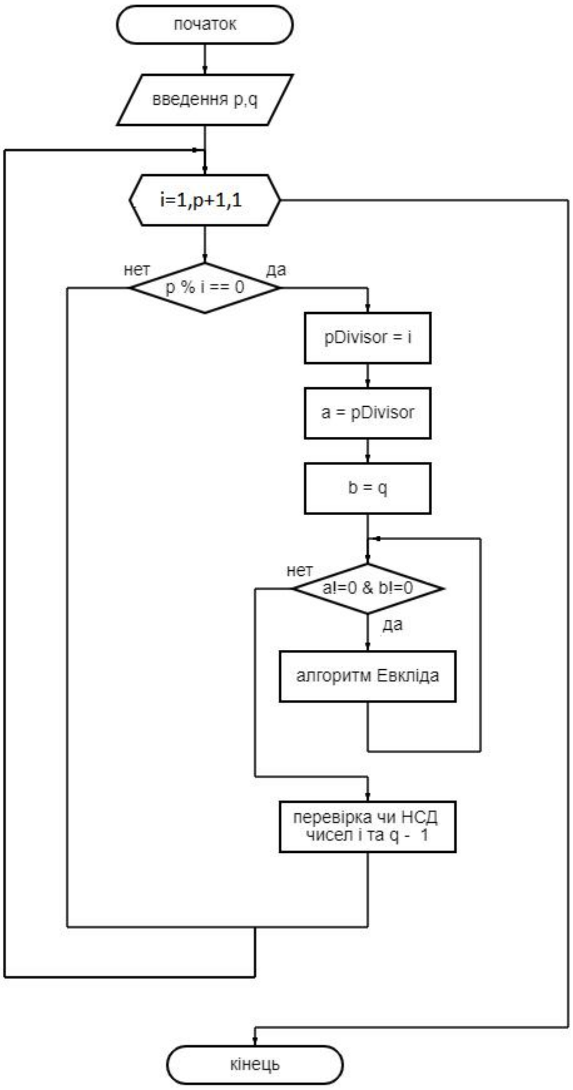
Крок 6.

```
початок
   введення p,q
   повторити
               для і від 1 до p+1
               якщо p % і == 0
                    то
                         pDivisor = i
                         a = pDivisor
                         b = q
                         поки (a!=0 & b!=0) повторити
                              якщо a>b
                                   то
                                        a%=b
                              інакше
                                        b %= a
                         все поки
                         biggestCommonDivisor = a+b
                         якщо biggestCommonDivisor == 0
                              то
                                 виведення pDivisor
               все якщо
   все повторити
кінець
```

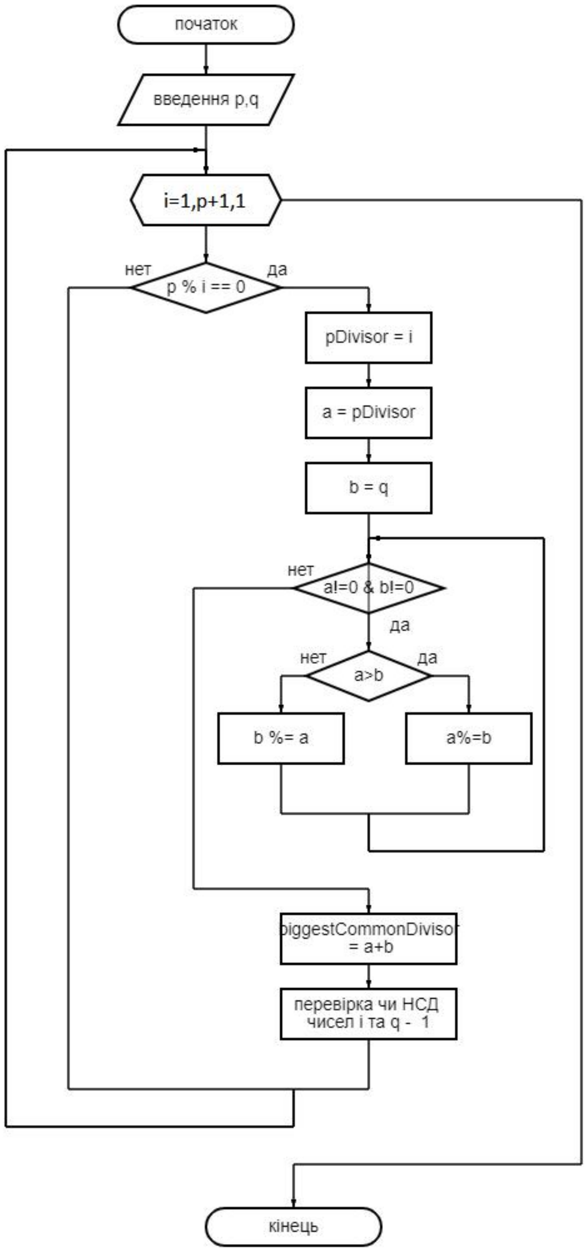
Блок схема:


Крок 1
початок
введення p,q
арифметичний цикл
перевірка чи є число
і дільником p
ітераційний цикл
алгоритм Евкліда
перевірка чи НОД
чисел і та q - 1
кінець
Крок 2
початок
введення p,q
i = 1 , p+1, 1
перевірка чи є число
і дільником p
ітераційний цикл
алгоритм Евкліда
перевірка чи НОД
чисел і та q - 1
кінець
Крок 3
початок
введення p,q
i=1,p+1,1
нет
да
p % i == 0
pDivisor = i
нет
a!=0 & b!=0
да
алгоритм Евкліда
перевірка чи НСД
чисел і та q - 1
кінець

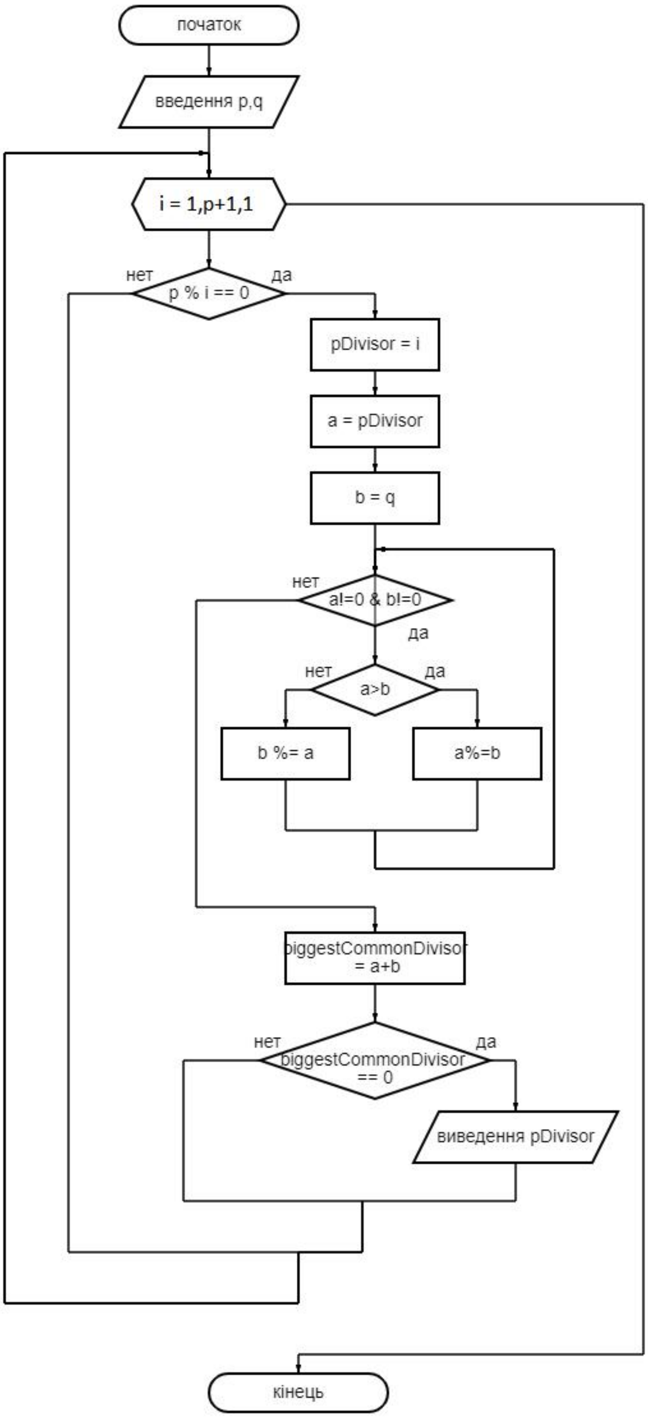
Крок 4


початок
введення p,q
i=1,p+1,1
нет
да
p % i == 0
pDivisor = i
a = pDivisor
b = q
нет
a!=0 & b!=0
да
алгоритм Евкліда
перевірка чи НСД чисел і та q - 1
кінець

Крок 5


початок
введення p,q
i=1,p+1,1
нет
да
p % i == 0
pDivisor = i
a = pDivisor
b = q
нет
a!=0 & b!=0
да
нет
да
a>b
b %= a
a%=b
biggestCommonDivisor = a+b
перевірка чи НСД чисел і та q - 1
кінець

Крок 6



4. Перевірка алгоритму

Блок	Дія
	Початок
1	Введення ***x***=16, ***n***=7,
2	**iteration: 1**
3	**(p % i == 0) = true**

	pDivisor = 1
	a = 1
	b = 7
	Euclid algorithm
	(a != 0 && b != 0) = true
	a<b = false
	a = 1
	b = 0
	biggestCommonDivisor = 1
	(biggestCommonDivisor == 1) = true
	виведення 1
	iteration: 2
	(p % i == 0) = true
	pDivisor = 2
	a = 2
	b = 7
	Euclid algorithm
	(a != 0 && b != 0) = true
	a<b = false
	a = 2
	b = 1
	(a != 0 && b != 0) = true
	a<b = true
	a = 0
	b = 1
	biggestCommonDivisor = 1
	(biggestCommonDivisor == 1) = true

	виведення 2
	iteration: 3
	(p % i == 0) = false
	iteration: 4
	(p % i == 0) = true
	pDivisor = 4
	a = 4
	b = 7
	Euclid algorithm
	(a != 0 && b != 0) = true
	a<b = false
	a = 4
	b = 3
	(a != 0 && b != 0) = true
	a<b = true
	a = 1
	b = 3
	(a != 0 && b != 0) = true
	a<b = false
	a = 1
	b = 0
	biggestCommonDivisor = 1
	(biggestCommonDivisor == 1) = true
	виведення 4
	iteration: 5
	(p % i == 0) = false
	iteration: 6
	(p % i == 0) = false

	iteration: 7
	(p % i == 0) = false
	iteration: 8
	(p % i == 0) = true
	pDivisor = 8
	a = 8
	b = 7
	Euclid algorithm
	(a != 0 && b != 0) = true
	a<b = true
	a = 1
	b = 7
	(a != 0 && b != 0) = true
	a<b = false
	a = 1
	b = 0
	biggestCommonDivisor = 1
	(biggestCommonDivisor == 1) = true
	виведення 8
	iteration: 9
	(p % i == 0) = false
	iteration: 10
	(p % i == 0) = false
	iteration: 11
	(p % i == 0) = false
	iteration: 12
	(p % i == 0) = false
	iteration: 13

	(p % i == 0) = false
	iteration: 14
	(p % i == 0) = false
	iteration: 15
	(p % i == 0) = false
	iteration: 16
	(p % i == 0) = true
	pDivisor = 16
	a = 16
	b = 7
	Euclid algorithm
	(a != 0 && b != 0) = true
	a<b = false
	a = 2
	b = 7
	(a != 0 && b != 0) = true
	a<b = false
	a = 2
	b = 1
	(a != 0 && b != 0) = true
	a<b = true
	a = 0
	b = 1
	biggestCommonDivisor = 1
	(biggestCommonDivisor == 1) = true
	виведення 16

(1,2,4,8,16)

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