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 13

Виконав студент ІП-15 Конденко Іван Ігорович
(шифр, прізвище, ім’я, по батькові)

Перевірив(-ла) Вечерковська Анастасія Сергіївна
(прізвище, ім’я, по батькові)

Київ 2021

Лабораторна робота 5

Дослідження складних циклічних алгоритмів

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

Індивідуальне завдання

Варіант 13

Постановка задачі

13. Натуральне число називається паліндромом, якщо його запис читається однаково з початку та з кінця (наприклад, 575, 9). Знайти всі числа-паліндроми, що не перевищують 100, та при піднесенні до квадрату також дають паліндроми.

Математична модель

Змінна	Тип	Ім'я	Призначення
Число	Цілочисельний	n	Вхідні дані
Число (копія)	Цілочисельний	n_copy	Проміжні дані
Квадрат числа	Цілочисельний	n_s	Проміжні дані
К-сть цифр числа	Цілочисельний	digits_count	Проміжні дані
Ітератор	Цілочисельний	i	Проміжні дані
Ітератор #2	Цілочисельний	j	Проміжні дані
Змінна для результату перевірки на поліномність	Логічний	is_polynome	Проміжні дані
Функція для знаходження конкретної цифри числа	Функція, вихід цілочисельний	get_digit(number, digit_position)	Проміжні дані

// - цілочисельне ділення. % - знаходження остачі від ділення.

Визначимо n, як ітератор арифметичного циклу від 1 до 100.

Спочатку визначимо к-сть цифр (digits_count) цього числа за допомогою циклу, ділячи копію числа (n_copy) на 10. Потім перевіримо на поліномність за допомогою циклу, що перевіряє рівність відповідних цифр, самі ж цифри знайдемо за допомогою функції get_digit(). Після цього знайдемо квадрат числа (n_s) і так само перевіримо його на поліномність. Якщо поліномність не порушена (is_polynome == true), то вивести число.

Функція get_digit приймає дві змінні, що відповідають числу та номеру цифри з правого боку, а повертає значення цієї цифри. Для цього число number цілочисельно ділиться на 10 (digit_position – 1) разів, а потім знаходиться остання цифра результату за допомогою остачі від ділення на 10, ця ж цифра і повертається.

Розв'язання

Крок 1. Визначаємо основні дії

Крок 2. Деталізуємо крок знаходження digits_count

Крок 3. Деталізуємо крок перевірки на поліномність число та його квадрату

Псевдокод

Крок 1

Початок

Введення n

Знаходження digits_count

Перевірка на поліномність число та його квадрату

Кінець

Крок 2

Початок

Введення n

Повторити для n від 1 до n=<100

digits_count = 0

n_copy = n

поки n_copy >=1

n_copy /=10

digits_count +=1

все поки

Перевірка на поліномність число та його квадрату

все повторити

Кінець

Крок 3

Початок

Введення n

Повторити для n від 1 до n=<100

digits_count = 0

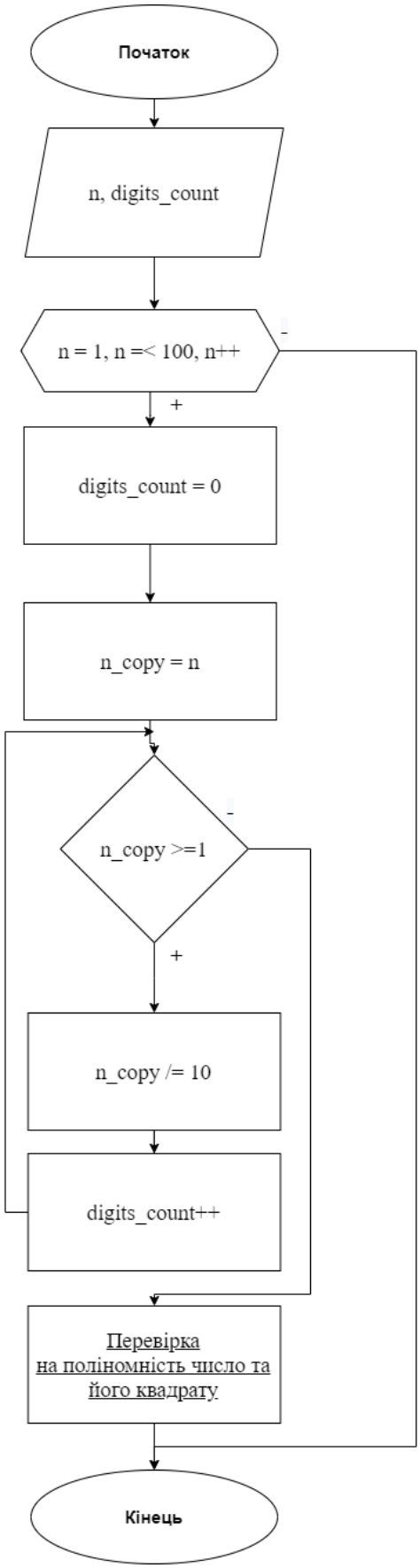
n_copy = n

```
        поки n_copy >=1
            n_copy /=10
            digits_count +=1
        все поки
        повторити для i від 1 до digits_count/2
            якщо get_digit(n, (digits_count - i)) != get_digit(n, (1 + i))
                is_polynome = false
                break
            все якщо
        все повторити
        is_polynome == true
        n_s = n*n
        n_copy = n_s
        digits_count = 0
        n_copy = n
        поки n_copy >=1
            n_copy /=10
            digits_count +=1
        все поки
        повторити для i від 1 до digits_count/2
            якщо get_digit(n, (digits_count - i)) != get_digit(n, (1 + i))
                is_polynome = false
                break
        все повторити
        is_polynome == true
        Вивести n
        все повторити
Кінець
Функція get_digit(number, digit_position)
Введення number, digit_position
    повторити для j від 1 до digit_position - 1
            number //=10
    все повторити
Повернути number %= 10
```

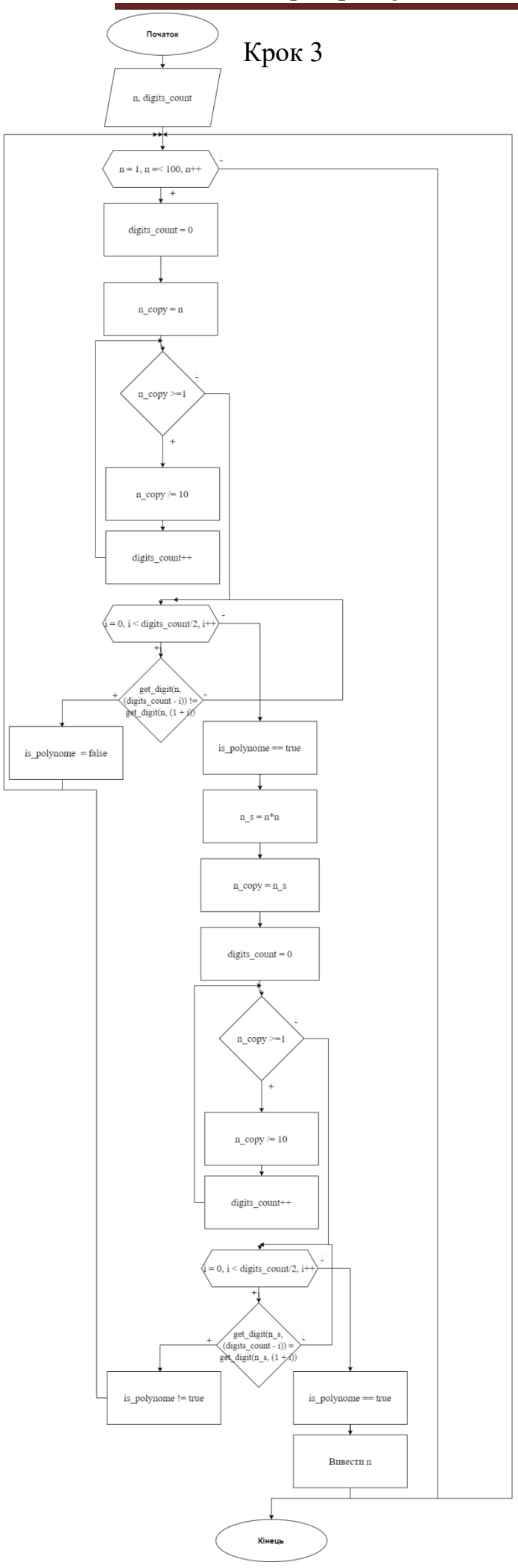
Блок схема

Крок 1

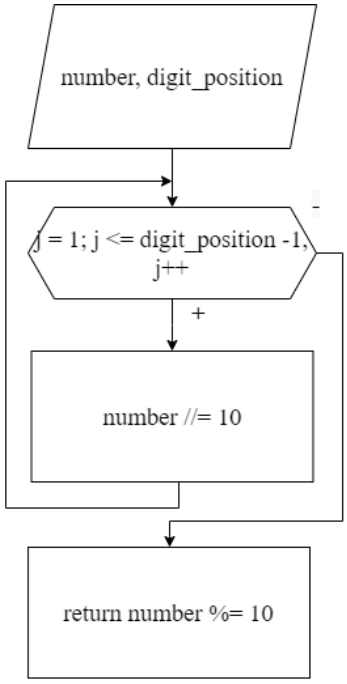
Крок 2



Крок 3


Початок
n, digits_count
n = 1, n =< 100, n++
digits_count = 0
n_copy = n
n_copy >=1
n_copy /= 10
digits_count++
i = 0, i < digits_count/2, i++
get_digit(n, (digits_count - i)) != get_digit(n, (1 + i))
is_polynome = false
is_polynome == true
n_s = n*n
n_copy = n_s
digits_count = 0
n_copy >=1
n_copy /= 10
digits_count++
i = 0, i < digits_count/2, i++
get_digit(n_s, (digits_count - i)) = get_digit(n_s, (1 + i))
is_polynome != true
is_polynome == true
Вивести n
Кінець

Блок-схема функції get_digit(number, digit_position)



Випробування

Блок	Дія
	Початок
1.	n = 1
2.	n_copy = 1
3.	digits_count = 1
4.	n_copy < 1
5.	digits_count/2=0,5; 0,5>0
6.	number = 1, digit_position = 1
7.	number % 10 = 1
8.	number = 1, digit_position = 1
9.	number % 10 = 1
10.	get_digit(1, (1)) = get_digit(1, (1))
11.	digits_count/2<1
12.	is_polynome == true
13.	n_s = n*n
14.	n_copy = n_s
15.	digits_count = 0
16.	n_copy >=1
17.	digits_count = 1
18.	n_copy < 1
19.	digits_count/2=0,5; 0,5>0
20.	number = 1, digit_position = 1

21.	number % 10 = 1
22.	number = 1, digit_position = 1
23.	number % 10 = 1
24.	get_digit(1, (1)) = get_digit(1, (1))
25.	digits_count/2<1
26.	is_polynome == true
27.	n = 1
…	…
…	n = 1, 2, 3, 11, 22
	Кінець

Висновки

Ми дослідили особливості роботи складних циклів та набули практичних навичок їх використання під час складання програмних специфікацій. В результаті виконання завдання ми виявили те, що існує лише 5 чисел та їх квадратів від 1 до 100, які є паліндромами.