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 22

Виконав студент ІП-12 Мельник Михайло Олександрович

Перевірив \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( 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.

**Варіант 22**

**Задача:** Натуральне число називається паліндромом, якщо його запис читається однаково з початку та з кінця (наприклад, 575, 9). Знайти всі паліндроми з інтервалу [106,109].

**Розв’язання:**

1. **Постановка задачі**

Результатом розв’язку задачі є послідовність чисел з інтервалу [106,109], які відповідають даній умові. Для визначення результату не потрібно задавати жодних початкових значень.

1. **Побудова математичної моделі**

***Складемо таблицю імен змінних***

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Поточне число | Порядковий | num | Проміжне дане |
| Число | Цілий | num\_\_ | Проміжне дане |
| Обернене число | Цілий | revNum | Проміжне дане, результат |
| Остання цифра (остача) | Цілий | rem | Проміжне дане |

Таким чином, математичне формулювання завдання зводиться до арифметичного циклу з лічильником num ∈ [1000000, 1000000000], в якому відбувається знаходження оберненого числа таким чином: num\_\_ = num; поки num\_\_ > 0, знаходимо останню цифру rem := num\_\_ % 10; до значення оберненого числа «прикріпляємо» в кінець цю останню цифру таким чином: revNum := revNum \* 10 + rem; відкидаємо останню цифру від num\_\_: num\_\_ := num\_\_ // 10. Відповідно, якщо обернене число дорівнює поточному, це число є поліндромом і ми його виводимо.

1. **Розроблення алгоритму**

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

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізуємо дію перебору чисел з інтервалу

*Крок 3.* Деталізуємо дію знаходження оберненого числа

*Крок 4.* Деталізуємо дію перевірки числа на умову та його вивід

***Псевдокод***

*крок 1*

**початок**

перебір чисел з інтервалу

знаходження оберненого числа

перевірка числа на умову, його вивід

**кінець**

*крок 2*

**початок**

**повторити**

**для** num **від** 1000000 **до** 1000000000

знаходження оберненого числа

перевірка числа на умову, його вивід

**все повторити**

**кінець**

*крок 3*

**початок**

**повторити**

**для** num **від** 1000000 **до** 1000000000

num\_\_ := num

revNum := 0

**повторити**

**поки** num\_\_ > 0

rem := num\_\_ % 10

revNum := revNum\*10 + rem

num\_\_ := num\_\_ // 10

**все повторити**

перевірка числа на умову, вивід

**все повторити**

**кінець**

*крок 4*

**початок**

**повторити**

**для** num **від** 1000000 **до** 1000000000

num\_\_ := num

revNum := 0

**повторити**

**поки** num\_\_ > 0

rem := num\_\_ % 10

revNum := revNum\*10 + rem

num\_\_ := num\_\_ // 10

**все повторити**

**якщо** revNum == num

**то**

**вивід** revNum

**все якщо**

**все повторити**

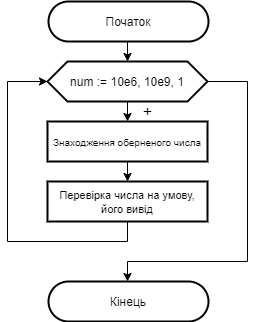
**кінець**

***Блок-схема***

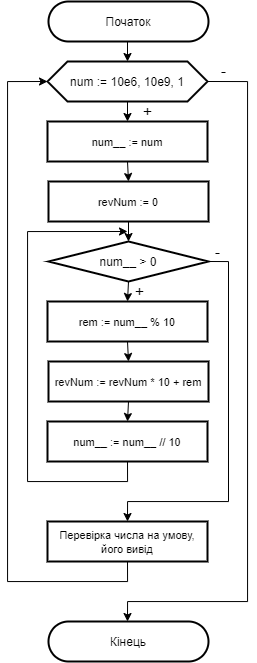
*Крок 1*

**

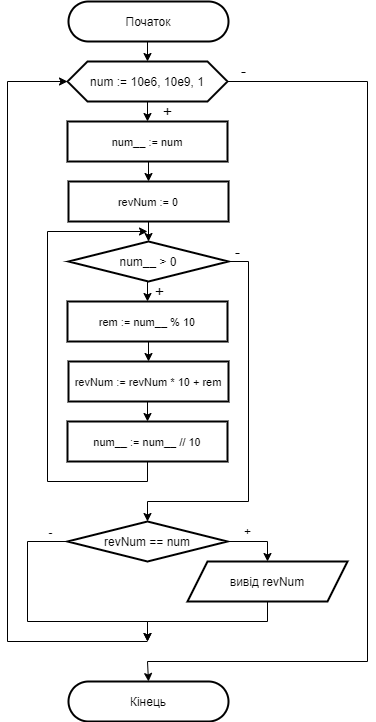
*Крок 2*

**

*Крок 3*

**

*Крок 4*

**

1. **Випробування алгоритму**

Перевіримо правильність алгоритму для перших двох ітерацій арифметичного циклу.

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | Початок |
| 1 | num := 1000000, num\_\_ := 1000000, revNum := 0 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 0 + 0 = 0, num\_\_ := 100000 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 0, num\_\_ = 10000 |
| 2 | … |
| 2 | num\_\_ := 10, num\_\_ > 0 істина, rem := 0, revNum := 0, num\_\_ = 1 |
| 2 | num\_\_ > 0 істина, rem := 1, revNum := 1, num\_\_ = 0 |
| 2 | num\_\_ > 0 хибне |
| 3 | revNum == num хибне |
| 1 | num := 1000001, num\_\_ := 1000001, revNum := 0 |
| 2 | num\_\_ > 0 істина, rem := 1, revNum := 0 + 1 = 1, num\_\_ := 100000 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 10 + 0 = 10, num\_\_ := 10000 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 100 + 0 = 100, num\_\_ := 1000 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 1000 + 0 = 1000, num\_\_ := 1000 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 10000, num\_\_ := 100 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 100000, num\_\_ := 10 |
| 2 | num\_\_ > 0 істина, rem := 0, revNum := 1000000, num\_\_ := 1 |
| 2 | num\_\_ > 0 істина, rem := 1, revNum := 1000000 + 1 = 1000001,  num\_\_ := 0 |
| 2 | num\_\_ > 0 хибне |
| 3 | revNum == num істина, вивід 1000001 |
|  | ... |
|  | ... |
|  | Кінець |

Алгоритм відповідає поставленій задачі, помилок не виявлено.

1. **Висновки**

Під час виконання даної лабораторної роботи було розроблено й випробувано алгоритм перевірки чисел із заданого проміжку на те, чи є вони поліндромами, досліджено особливості роботи складних циклів та набуто практичних навичок їх використання під час складання програмних специфікацій.