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 29

Виконав студент ІП-11 Тарасьонок Дмитро Євгенович

(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна

( 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 29**

Знайти всі чотирьохзначні паліндроми

1. **Постановка задачі:**

Постановка задачі зводиться до того, що нам необхідно взяти всі чотирьохзначні числа та подивитися, чи буде це число читатися в обох напрямках однаково

# Побудова математичної моделі. Складемо таблицю імен змінних:

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Довжина числа | Цілий | numlen | Проміжні дані |
| Число | Цілий | num | Проміжні дані, лічильник |
| Дубль числа | Цілий | tempnum | Проміжні дані |
| Старша цифра | Цілий | first | Проміжні дані |
| Молодша цифра | Цілий | last | Проміжні дані |
| Паліндромність числа | Булевий | palindrome | Результат |

Для вирішення даної задачі можна використати два підходи. Перший буде підходити виключно для чотирьохзначних чисел. Він передбачає знаходження всіх цифр числа окремо і порівняння першої з четвертою, другої з третьою. Другий же підхід є універсальним для числа для будь-якої довжини. Пропонується використати саме другий підхід. У ньому необхідно спочатку знайти довжину числа шляхом поступового цілочисельного ділення на 10, поки воно не стане рівним нулю, при цьому збільшуючи змінну довжини числа. Далі необхідно поступово знаходити старші та молодші цифри, порівнюючи їх між собою. Якщо вони рівні, виводити число.

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію перевірки числа на паліндромність

Крок 3. Деталізуємо дію обчислення довжини числа

Крок 4. Деталізуємо дію порівняння цифр числа

Крок 5. Деталізуємо дію виведення інформації щодо паліндромності

1. **Псевдокод:**

*Крок 1*

# Початок

Перевірка числа на паліндромність

**Кінець**

*Крок 2*

# Початок

**для num від 1000 до 1000**

**повторити**

palindrome := true

numlen := 0

tempnum := num

Обчислення довжини числа

Порівняння цифр числа

Виведення, чи є це число паліндромом

**все повторити**

**Кінець**

*Крок 3*

# Початок

**для num від 1000 до 1000**

**повторити**

palindrome := true

numlen := 0

tempnum := num

поки tempnum != 0

**повторити**

numlen := numlen + 1

tempnum := tempnum div 10

**все повторити**

Порівняння цифр числа

Виведення, чи є це число паліндромом

**все повторити**

**Кінець**

*Крок 4*

# Початок

**для num від 1000 до 1000**

**повторити**

palindrome := true

numlen := 0

tempnum := num

поки tempnum != 0

**повторити**

numlen := numlen + 1

tempnum := tempnum div 10

**все повторити**

**для i від 0 поки i < numlen div 2**

**повторити**

first := num div 10 ^ (numlen - i – 1) mod 10

last := 0

tempnum := num

**для o від 0 поки o <= i**

**повторити**

last := tempnum mod 10

tempnum := tempnum div 10

**все повторити**

**якщо first := last**

**то**

palindrome := false

**все якщо**

Виведення, чи є це число паліндромом

**все повторити**

**Кінець**

*Крок 5*

# Початок

**для num від 1000 до 1000**

**повторити**

palindrome := true

numlen := 0

tempnum := num

поки tempnum != 0

**повторити**

numlen := numlen + 1

tempnum := tempnum div 10

**все повторити**

**для i від 0 поки i < numlen div 2**

**повторити**

first := num div 10 ^ (numlen - i – 1) mod 10

last := 0

tempnum := num

**для o від 0 поки o <= i**

**повторити**

last := tempnum mod 10

tempnum := tempnum div 10

**все повторити**

**якщо first := last**

**то**

palindrome := false

**все якщо**

**якщо palindrome == true**

**то**

Виведення «num – число паліндром»

**все якщо**

**все повторити**

**Кінець**

**4. Блок-схема:**



# 5. Випробування алгоритму:

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | Початок |
| **1** | num = 1000 |
| **2** | numlen = 4 |
| **3.1** | first = 1  last = 0 |
| **3.2** | palindrome = false |
| **3.3** | first = 0  last = 0 |
| **3.4** | palindrome = false |
|  | Кінець |

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | Початок |
| **1** | num = 1001 |
| **2** | numlen = 4 |
| **3.1** | first = 1  last = 1 |
| **3.2** | palindrome = true |
| **3.3** | first = 0  last = 0 |
| **3.4** | palindrome = true |
| **4** | Виведення «1001 – число-паліндром» |
|  | Кінець |

# 6. Висновки:

За допомогою математичної моделі та циклів можна запрограмувати знаходження всіх чотирьохзначних паліндромів одразу двома способами: як простим, так і більш складним. Більш складний у свою чергу є універсальним, коли простий підходить виключно для чотирьохзначних чисел. Виконується цей алгоритм за допомогою складних циклів з використанням одного зовнішнього арифметичного циклу, та трьох вкладених: одного ітераційного з передумовою та двох арифметичних, один із яких вкладено в другий