

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 № 6 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження рекурсивних алгоритмів»

Варіант 35

Виконав студент ІП-15, Шабанов Метін Шаміль огли
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Лабораторна робота 6

Дослідження складних циклічних алгоритмів

Мета – дослідити особливості роботи рекурсивних алгоритмів та набути практичних навичок їх використання під час складання програмних специфікацій підпрограм.

Індивідуальне завдання

Варіант 35

Умова задачі

Сформувати послідовність з k чисел Фібоначчі: перші два значення дорівнюють 0 та 1, а кожне наступне значення – це сума двох попередніх.

Постановка задачі

Обчислити k (задані користувачем) елементів послідовності Фібоначчі, використовуючи рекурсивний алгоритм, з заданими першими членами: 0 та 1. Результатом виконання алгоритму є набір числових значень.

Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Число елементів послідовності, обмежувач циклу	Ціле додатне	k	Вхідні дані
Лічильник циклу	Ціле додатне	i	Проміжні дані
Поточне значення елемента послідовності	Ціле додатне	result	Результат
Параметр функції	Ціле додатне	n	Проміжні дані
Функція для обчислення поточного елемента	Підпрограма	fibonacci(n)	Функція

Для обчислення значення k елементів послідовності скористаємося арифметичним циклом з початковим значенням $i = 1$, умовою $i \leq k$ та кроком 1. У тілі циклу виклинемо функцію fibonacci, яка параметром буде приймати лічильник циклу. Підпрограма fibonacci містить умовний оператор, який при параметрі 1 повертає значення 0, при параметрі 2 повертає значення 1, а інакше значення виразу $(\text{fibonacci}(n-1) + \text{fibonacci}(n-2))$.

Розв'язання

Програмні специфікації запишемо у псевдокодi та графічній формi у вигляді блок-схеми.

Крок 1. Визначимо основні дії;

Крок 2. Деталізуємо роботу арифметичного циклу;

Крок 3. Деталізуємо присвоєння змінній result поточного значення функції та виклик підпрограми.

Крок 4. Деталізуємо роботу підпрограми.

Псевдокод

Крок 1

Початок

Робота арифметичного циклу

Присвоєння змінній result поточного значення функції та виклик підпрограми

Робота підпрограми

Кінець

Крок 2

Початок

повторити

для i від 1 до k включно

Присвоєння змінній result поточного значення функції та виклик підпрограми

все повторити

Робота підпрограми

Кінець

Крок 3

Початок

повторити

для i від 1 до k включно

result = fibo(i)

все повторити

Робота підпрограми

Кінець

Крок 4

Початок

Основна програма

повторити

для i від 1 до k включно

result = fibo(i)

все повторити

Підпрограма fibo(n)

якщо $n == 1$

то

повернути 0

інакше якщо $n == 2$

то

повернути 1

інакше

повернути fibo($n - 1$) + fibo($n - 2$)

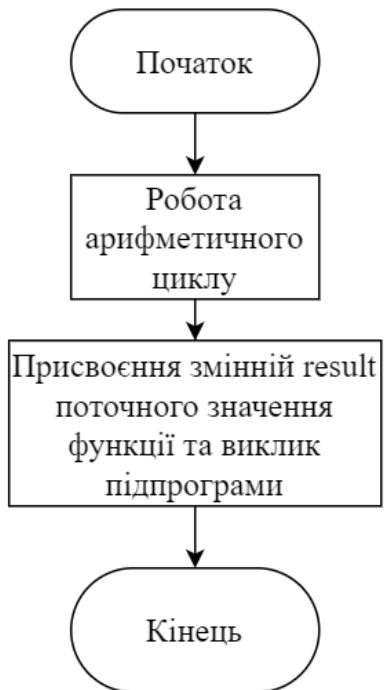
все якщо

все якщо

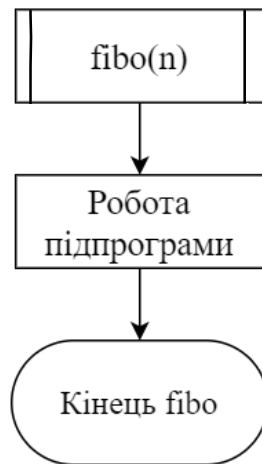
Кінець

Блок схема

Основна програма



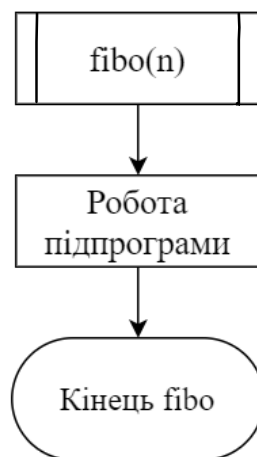
Підпрограма



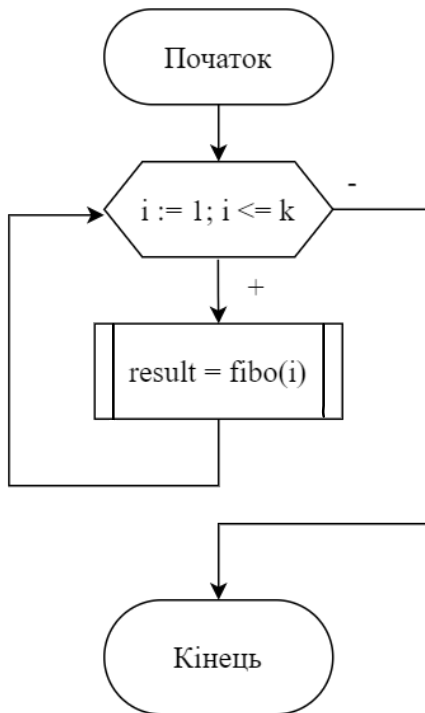
Основна програма



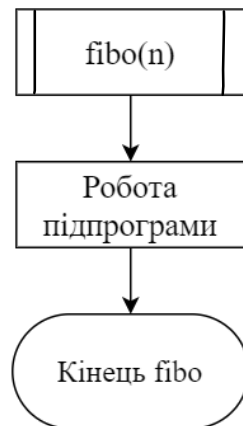
Підпрограма



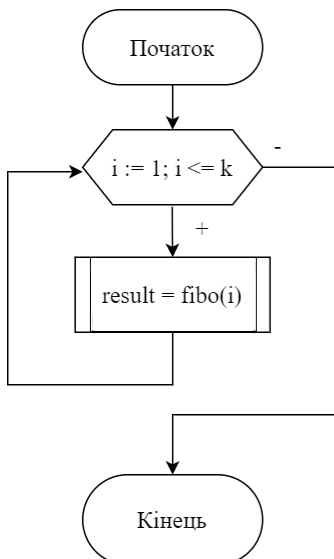
Основна програма



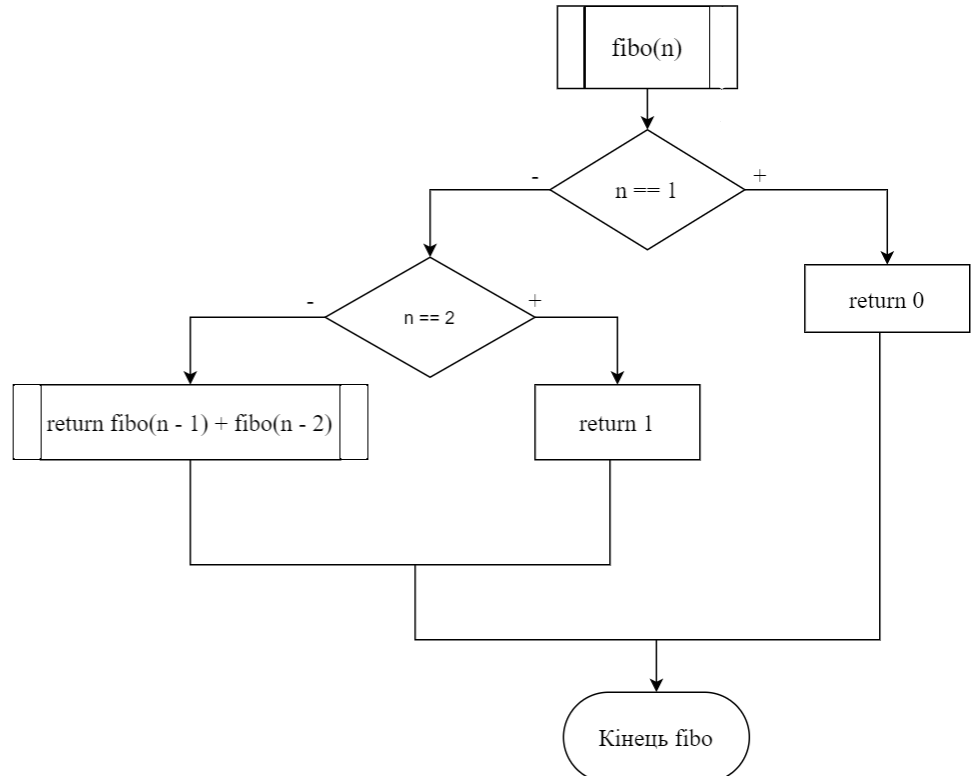
Підпрограма



Основна програма



Підпрограма



Код програми

```
#include <iostream>
using namespace std;
#include <Windows.h>

int fibo(int);

int main()
{
    SetConsoleOutputCP(1251);
    cout << "Введіть кількість елементів послідовності: ";
    int k;
    cin >> k;
    int result;
    for (int i = 1; i <= k; i++)
    {
        result = fibo(i);
        cout << result << endl;
    }
    return 0;
}

int fibo(int n)
{
    if (n == 1)
    {
        return 0;
    }
    else if (n == 2)
    {
        return 1;
    }
    else
    {
        return fibo(n - 1) + fibo(n - 2);
    }
}
```

Висновки

Ми дослідили особливості роботи рекурсивних алгоритмів та набули практичних навичок їх використання під час складання програмних специфікацій підпрограм. Як результат, ми отримали алгоритм обчислення k елементів послідовності Фібоначчі, розділивши задачу на чотири кроки: визначення основних дій, деталізація роботи арифметичного циклу, деталізація присвоєння змінній `result` поточного значення функції та виклик підпрограми, деталізація роботи підпрограми. В процесі випробовування при $k = 9$ ми отримали результати 0; 1; 1; 2; 3; 5; 8; 13; 21.