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 25

Виконав ІП-15, Плугатирьов Дмитро Валерійович

студент (шифр, прізвище, ім’я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна

(прізвище, ім’я, по батькові)

Київ 2021

**Лабораторна робота № 6**

**Дослідження рекурсивних алгоритмів**

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

**Варіант 25**

**Завдання**

25.) Отримати всі піфагорові трійки натуральних чисел, кожне з яких не перевищує n, тобто всі такі трійки натуральних чисел a, b, c, що a2+ b2= c2

(a ≤, b ≤, c ≤ n).

1. **П о с т а н о в к а з а д а ч і**

Оскільки нашою остаточною ціллю є отримання значень змінних, які утворюють піфагорові трійки, значення яких не може перевищувати задане користувачем, то можна знайти кожну трійку за допомогою використання рекурсії. Числа є натуральними та обраховуються наступним чином: якщо під час одного із рекурсивних викликів функції вийшло так, що третя змінна (с) дорівнює введеному користувачем числу, то відбуватиметься присвоєння одиниці до с та збільшення значення змінної, яка розташовується лівіше у формулі Піфагора на одиницю.

Результатом є виведення усіх можливих піфагорових трійок з дублюванням (оскільки перша та друга змінні дають правильний результат, коли є взаємо обернені) та кількості трійок без урахування дублювання.

1. **П о б у д о в а м а т е м а т и ч н о ї м о д е л і**

|  |  |  |  |
| --- | --- | --- | --- |
| *Змінна* | *Тип* | *Ім’я* | *Призначення* |
| Дане число | Цілочисельний | n | Початкові дані |
| Перша змінна | Цілочисельний | a | Проміжні дані |
| Друга змінна | Цілочисельний | b | Проміжні дані |
| Третя змінна | Цілочисельний | c | Проміжні дані |
| Кількість трійок | Цілочисельний | count | Проміжні дані |
| Пошук піфагорових трійок | Процедура | pythagorean\_trio | Проміжні дані |
| Виведення трійок | Процедура | output | Результат |

* Для вирішення даної задачі треба скористатися теоремою Піфагора: сума квадратів катетів прямокутного трикутника дорівнює квадрату гіпотенузи.
* Пошук піфагорових трійок відбувається шляхом послідовного перебору значень правої змінної до значення, заданого користувачем, після проходження якого значення зліва збільшується на одиницю і т.д.

Дія pow(x, m) означає піднесення числа x до степені m.

Дії x += 1 та ++x означають x := x + 1.

Дія **вивести** x, v, t означає послідовне виведення x, v, t.

1. **Р о з в ' я з а н н я**

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

*Крок 1.* Визначити основні дії.

*Крок 2.* Введення обмежувальної змінної та перевірка її значення на натуральність.

*Крок 3.* Виклик підпрограми, реалізація рекурсій в ній, складання умови виведення потрібних даних.

1. **П с е в д о к о д**

**Основна програма:**

*Крок 1*

**початок**

введення обмежувальної змінної та перевірка її значення на натуральність

виклик підпрограми, реалізація рекурсій в ній, складання умови виведення потрібних даних

**кінець**

*Крок 2*

**початок**

a := 1

b := 1

c := 1

count := 0

**ввести** n

**повторити**

**поки** n <= 0

**ввести** n

**все повторити**

виклик підпрограми, реалізація рекурсій в ній, складання умови виведення потрібних даних

**кінець**

*Крок 3*

**початок**

a := 1

b := 1

c := 1

count := 0

**ввести** n

**повторити**

**поки** n <= 0

**ввести** n

**все повторити**

pythagorean\_trio(n, a, b, c, count)

**вивести** count

**кінець**

**Підпрограми:**

**output(a, b, c, count)**

**вивести** count, a, b, c

**кінець**

**pythagorean\_trio(n, a, b, c, count)**

**якщо** pow(a, 2) + pow(b, 2) == pow(c, 2)

**то**

**++**count

output(a, b, c, count)

**все якщо**

**якщо** c != n

**то**

pythagorean\_trio(n, a, b, ++c, count)

**інакше якщо** c == n **та** b != n

**то**

c := 1

pythagorean\_trio(n, a, ++b, c, count)

**інакше якщо** c == n **та** b == n **та** a != n

**то**

b := 1

c := 1

pythagorean\_trio(n, ++a, b, c, count)

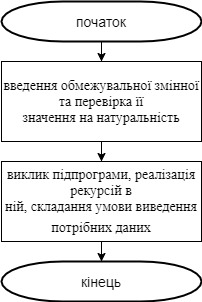
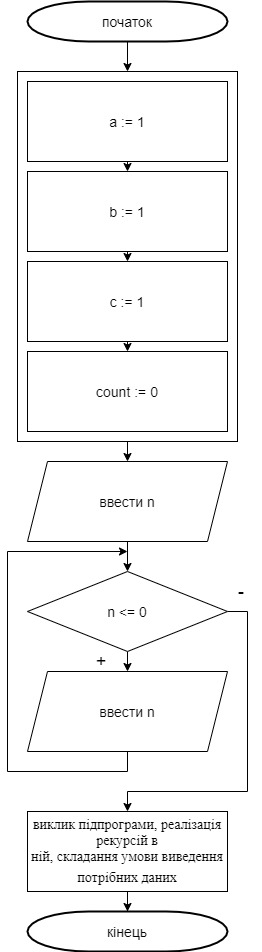
**все якщо**

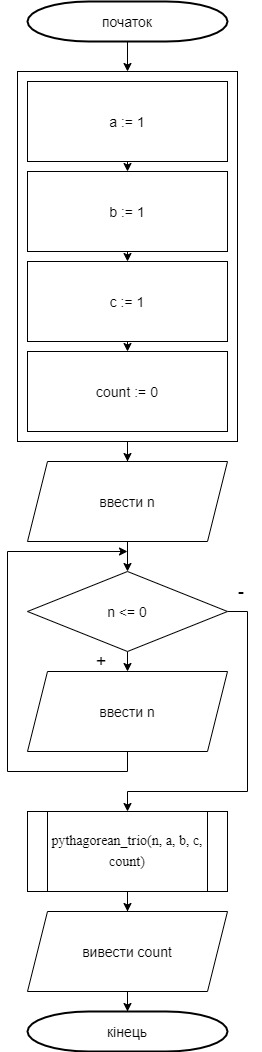
**кінець**

*Блок-схема*

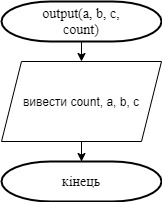
**Основна програма:**

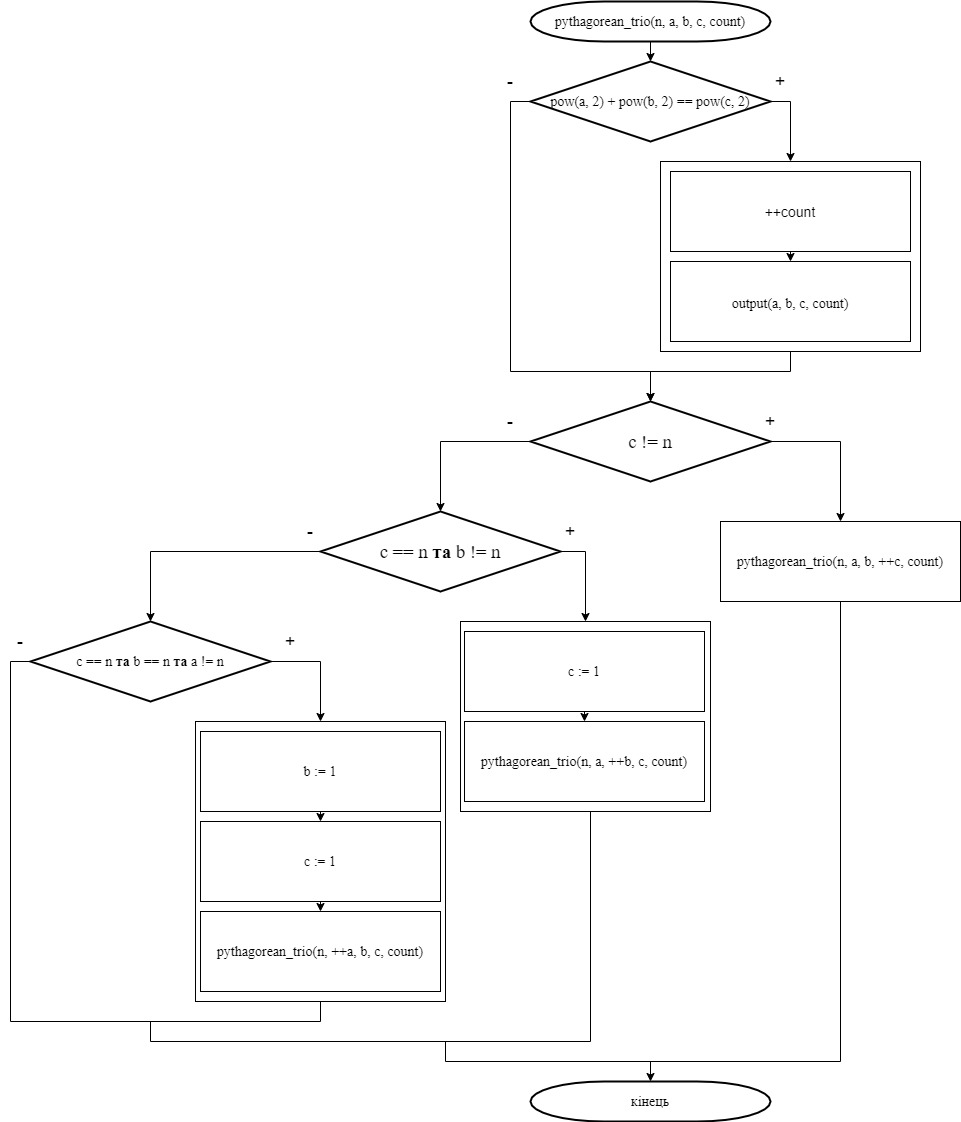
*Крок 1 Крок 2*

**

*Крок 3*

**Підпрограми:**

****

****

**Код програми**

#include <iostream>

#include <cmath>

void pythagorean\_trio(int n, int a, int b, int c, int& count);

void output(int a, int b, int c, int count);

// TASK 25

int main()

{

int n, a = 1, b = 1, c = 1, count = 0;

// Input of the numbers

std::cout << "Please, enter the natural n: ";

std::cin >> n;

while (n <= 0) // The codition of a natural number

{

std::cout << "n is less or equal to 0. Please, enter a natural one: ";

std::cin >> n;

}

pythagorean\_trio(n, a, b, c, count); // Calculation of pythagorean trios

std::cout << "The quantity of pythagorean trios is: " << count / 2 << std::endl; // Output of the count of trios

}

void pythagorean\_trio(int n, int a, int b, int c, int& count)

{

if (a <= n && b <= n && c <= n)

{

if (pow(a, 2) + pow(b, 2) == pow(c, 2)) // Completed pythagorean trio

{

count++;

output(a, b, c, count);

}

if (c != n)

{

pythagorean\_trio(n, a, b, ++c, count);

}

else if (c == n && b != n)

{

c = 1;

pythagorean\_trio(n, a, ++b, c, count);

}

else if (c == n && b == n && a != n)

{

b = 1;

c = 1;

pythagorean\_trio(n, ++a, b, c, count);

}

}

}

void output(int a, int b, int c, int count) // Output of a, b and c

{

std::cout << "There is the " << count << " pythagorean trio! It's consisted of: " << std::endl;

std::cout << "a, which equals to: " << a << std::endl;

std::cout << "b, which equals to: " << b << std::endl;

std::cout << "c, which equals to: " << c << std::endl;

}

**5.) Т е с т у в а н н я**

|  |  |  |  |
| --- | --- | --- | --- |
| **Блок** | **Дія 1** |  | **Дія 2** |
|  | **Початок** |  | **Початок** |
| **1** | **a := 1, b := 1, c := 1,**  **count := 0** | **1** | **a := 1, b := 1, c := 1, count := 0** |
| **2** | **n := 5** | **2** | **n := 10** |
| **3** | **pythagorean\_trio(n, a, b, c, count)** | **3** | **pythagorean\_trio(n, a, b, c, count)** |
| **4** | **pythagorean\_trio(n, a, b, ++c, count)** | **4** | **pythagorean\_trio(n, a, b, ++c, count)** |
| **5** | **pythagorean\_trio(n, a, b, ++c, count)** | **5** | **pythagorean\_trio(n, a, b, ++c, count)** |
| **…** | **…** | **…** | **…** |
| **22** | **c := 1** | **22** | **c := 1** |
| **23** | **pythagorean\_trio(n, a, ++b, c, count)** | **23** | **pythagorean\_trio(n, a, ++b, c, count)** |
| **24** | **pythagorean\_trio(n, a, b, ++c, count)** | **24** | **pythagorean\_trio(n, a, b, ++c, count)** |
| **…** | **…** | **…** | **…** |
| **44** | **c := 1** | **44** | **c := 1** |
| **45** | **pythagorean\_trio(n, a, ++b, c, count)** | **45** | **pythagorean\_trio(n, a, ++b, c, count)** |
| **…** | **…** | **…** | **…** |
| **111** | **b := 1** | **111** | **b := 1** |
| **112** | **c := 1** | **112** | **c := 1** |
| **113** | **pythagorean\_trio(n, ++a, b, c, count)** | **113** | **pythagorean\_trio(n, ++a, b, c, count)** |
| **…** | **…** | **…** | **…** |
| **310** | **count += 1** | **310** | **count += 1** |
| **311** | **a := 3, b := 4, c := 5, count := 1,**  **output(a, b, c, count)** | **311** | **a := 3, b := 4, c := 5, count := 1,**  **output(a, b, c, count)** |
| **…** | **…** | **…** | **…** |
| **409** | **count += 1** | **409** | **count += 1** |
| **410** | **a := 4, b := 3, c := 5, count := 2,**  **output(a, b, c, count)** |
| **…** | **…** |
| **620** | **count += 1** |
| **621** | **a := 6, b := 8, c := 10, count := 3,**  **output(a, b, c, count)** |
| **…** | **…** |
| **820** | **count += 1** |
| **821** | **a := 8, b := 6, c := 10, count := 4,**  **output(a, b, c, count)** |
|  | **Кінець** |

|  |  |
| --- | --- |
|  | **Кінець** |

**6.) В и с н о в о к**

В цій лабораторній роботі мені довелося зайнятися дослідженням особливостей роботи рекурсивних алгоритмів та закріпити навички на практиці, складаючи програмні специфікації підпрограм. А саме, реалізація виводу трьох чисел, які утворюють піфагорову трійку за допомогою рекурсії і певних умов виводу даних та виходу з неї.