Міністерство освіти і науки України

Національний технічний університет України “Київський політехнічний

інститут імені Ігоря Сікорського ˮ

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни

“Алгоритми та структури даних-1.

Основи алгоритмізації ˮ

“ Дослідження лінійних алгоритмів ˮ

Варіант:12

Виконав студент: ІП-12 Єльчанінов Артем Юрійович

(шифр, прізвище, ім'я, по батькові)

Перевірив: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(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 12**

**Задача:** Обчислити суму елементів арифметичної прогресії, що убуває: початкове значення - 3π, кінцеве – -4π, крок – π/2.

**Постановка задачі**

Результатом розв’язку задачі є обчислення суми елементів арифметичної прогресії, що убуває.

Обчислення суми буде відбуватися за рекурсивним алгоритмом, де функція(sum\_of\_progression) буде сама себе викликати, цим самим обчислюючи наступні члени прогресії, поки початкове значення арифметичної прогресії не буде дорівнювати кінцевому значенню прогресії. Після цього функція буде зворотнім ходом до початкового значення арифметичної прогресії додавати попередньо обчисленні наступні значення членів прогресії, цим самим обчислить її суму, і задача буде виконана.

**Математична модель**

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Початкове значення арифметичної прогрессії | Дійсний | start | Вхідне дане |
| Кінцеве значення арифметичної прогресії | Дійсний | end | Вхідне дане |
| Крок зменшення значення членів ариметичної прогресії | Дійсний | step | Вхідне дане |
| Рекурсивна функція обчислення суми арифметичної прогресії | Дійсний | sum\_of\_progression | Обчислення арифметичної прогресії |
| Сума арифметичної прогресії | Дійсний | sum | Вихідне дане |

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

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію знаходження суми арифметичної прогресії з використанням функції.

**Псевдокод алгоритму**

**Крок 1.**

**Початок**

Ініціалізація змінних

Знаходження суми арифметичної прогресії з використанням функції

**Виведення:** sum

**Кінець**

**Крок 2.**

**Початок**

start:= 3π; end:= -4π; step:= π/2; sum:= 0;

Знаходження суми арифметичної прогресії з використанням функції

**Виведення:** sum

**Кінець**

**Крок 3.**

**Початок**

start:= 3π; end:= -4π; step:= π/2; sum:= 0;

sum:= sum\_of\_progression(start, end, step, sum)

**Виведення:** sum

**Кінець**

**sum\_of\_progression(start, end, step, sum)**

**якщо** start == end

**то**

sum:= end

**інакше**

sum:=start + sum\_of\_progression(start - step, end, step, sum)

**все якщо**

**Повернути** sum

**Блок-схема**

**Крок 1: Крок 2:**

 

**Крок 3:**





**Випробування алгоритму:** Перевіримо правильність алгоритму на довільних конкретних значеннях початкових даних.

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | **Початок** |
| 1 | start:= 3π; end:= -4π; step:= π/2; sum:= 0 |
| 2 | sum:= sum\_of\_progression(3π, -4π, π/2, 0) |
| 3 | (3π == -4π) == false |
| 4 | sum:=3π + sum\_of\_progression(3π - π/2, -4π, π/2, 0) |
| 5 | (2.5π == -4π) == false |
| 6 | sum:=2.5π + sum\_of\_progression(2.5π - π/2, -4π, π/2, 0) |
| 7 | (2π == -4π) == false |
| 8 | sum:=2π + sum\_of\_progression(2π - π/2, -4π, π/2, 0) |
| 9 | (1.5π == -4π) == false |
| 10 | sum:=1.5π + sum\_of\_progression(1.5π - π/2, -4π, π/2, 0) |
| 11 | (π == -4π) == false |
| 12 | sum:=π + sum\_of\_progression(π - π/2, -4π, π/2, 0) |
| 13 | (0.5π == -4π) == false |
| 14 | sum:=0.5π + sum\_of\_progression(0.5π - π/2, -4π, π/2, 0) |
| 15 | (0 == -4π) == false |
| 16 | sum:=0 + sum\_of\_progression(0 - π/2, -4π, π/2, 0) |
| 17 | (-0.5π == -4π) == false |
| 18 | sum:=-0.5π + sum\_of\_progression(-0.5π - π/2, -4π, π/2, 0) |
| 19 | (-π == -4π) == false |
| 20 | sum:= **-**π + sum\_of\_progression(-π - π/2, -4π, π/2, 0) |
| 21 | (-1.5π == -4π) == false |
| 22 | sum:= **-**1.5π + sum\_of\_progression(-1.5π - π/2, -4π, π/2, 0) |
| 23 | (-2π == -4π) == false |
| 24 | sum:=-2π + sum\_of\_progression(-2π - π/2, -4π, π/2, 0) |
| 25 | (-2.5π == -4π) == false |
| 26 | sum:=-2.5π + sum\_of\_progression(-2.5π - π/2, -4π, π/2, 0) |
| 27 | (-3π == -4π) == false |
| 28 | sum:= **-**3π + sum\_of\_progression(-3π - π/2, -4π, π/2, 0) |
| 29 | (-3.5π == -4π) == false |
| 30 | sum:= **-**3.5π + sum\_of\_progression(-3.5π - π/2, -4π, π/2, 0) |
| 31 | (-4π == -4π) == true |
| 32 | sum = -4π |
| 33 | return -4π |
| 34 | sum = -3.5π + (-4π) = -7.5π |
| 35 | return -7.5π |
| 36 | sum = -3π + (-7.5π) = -10.5π |
| 37 | return -10.5π |
| 38 | sum = -2.5π + (-10.5π) = -13π |
| 39 | return -13π |
| 40 | sum = -2π + (-13π) = -15π |
| 41 | return -15π |
| 42 | sum = -1.5π + (-15π) = -16.5π |
| 43 | return -16.5π |
| 44 | sum = -π + (-16.5π) = -17.5π |
| 45 | return -17.5π |
| 46 | sum = -0.5π + (-17.5π) = -18π |
| 47 | return -18π |
| 48 | sum = 0 + (-18π) = -18π |
| 49 | return -18π |
| 50 | sum = 0.5π + (-18π) = -17.5π |
| 51 | return -17.5 π |
| 52 | sum = π + (-17.5π) = -16.5π |
| 53 | return -16.5π |
| 54 | sum = 1.5π + (-16.5π) = -15π |
| 55 | return -15π |
| 56 | sum = 2π + (-15π) = -13π |
| 57 | return -13π |
| 58 | sum = 2.5π + (-13π) = -10.5π |
| 59 | return -10.5π |
| 60 | sum = 3π + (-10.5π) = -7.5π |
| 61 | return -7.5π |
| 62 | *Виведення:* -23.5619 |
|  | **Кінець** |

**Код програми на мові С++ :**

#include <iostream>

#define \_USE\_MATH\_DEFINES

#include <math.h>

using namespace std;

double sum\_of\_progression(double start, double end, double step, double sum);

int main() {

double start, end, step, sum = 0;

start = 3 \* M\_PI;

end = -4 \* M\_PI;

step = M\_PI\_2;

sum = sum\_of\_progression(start, end, step, sum);

cout << "Sum of progression: " << sum << endl;

system("pause");

return 0;

}

double sum\_of\_progression(double start, double end, double step, double sum) {

if (start == end) {

sum = end;

}

else {

sum = start + sum\_of\_progression(start - step, end, step, sum);

}

return sum;

}

**Тестування програми:**



**Висновок.**

У результаті лабораторної роботи було розроблено математичну модель, що відповідає постановці задачі; псевдокод та блок-схеми, які пояснюють логіку алгоритму. Було набуто практичного новичок у складанні складних рекурсивних алгоритмів та їх інтерпретації у блок-схеми і псевдокод.

Алгоритм був випробуваний з використанням значень, заданих умовою задачі:

start = 3π, end = -4π, step = π/2; у підсумку було отримано, що sum = -23.5619. Таким чином, було доведено вірність складеного алгоритму. Отже, його можна застосовувати для обчислення суми елементів арифметичної прогресії, що убуває.