|  |  |
| --- | --- |
| Изображение выглядит как текст, небо, внешний  Автоматически созданное описание | |
| Національний технічний університет України  «Київський політехнічний інститут ім. Ігоря Сікорського»    Інститут прикладного системного аналізу  Кафедра системного проектування | |
| **Звіт** | |
| **про виконання лабораторної роботи №3** | |
| **з дисципліни «Алгоритмізація та програмування»** | |
| Виконав:  Студент гр. ДА-12  Панасенко О. А.  Прийняв:  к.т.н., доцент Безносик О. Ю. |  |
| Київ - 2021 | |

**Мета роботи**

Закріплення навичок програмної реалізації алгоритмів розгалужених та циклічних структур, оволодіння навичками реалізації найпростіших алгоритмів, включаючи алгоритм ітераційної циклічної структури: обчислення суми кінцевого числа доданків, обчислення добутку кінцевого числа множників, визначення найбільшего (найменшого) значення з усіх даних, що вводяться, обчислення суми збіжної послідовності із заданою точністю.

**Завдання 21**

Скласти схему алгоритму і програму обчислень суми ряду з точністю eps1 = 0.01 і eps2 = 0.0001:

Изображение выглядит как текст, часы

Автоматически созданное описание

Вивести в двох випадках на друк значення s, n (число ітерацій), еps, а також лістинг програми. Зробити висновки.

**1. Аналіз умови задачі.**

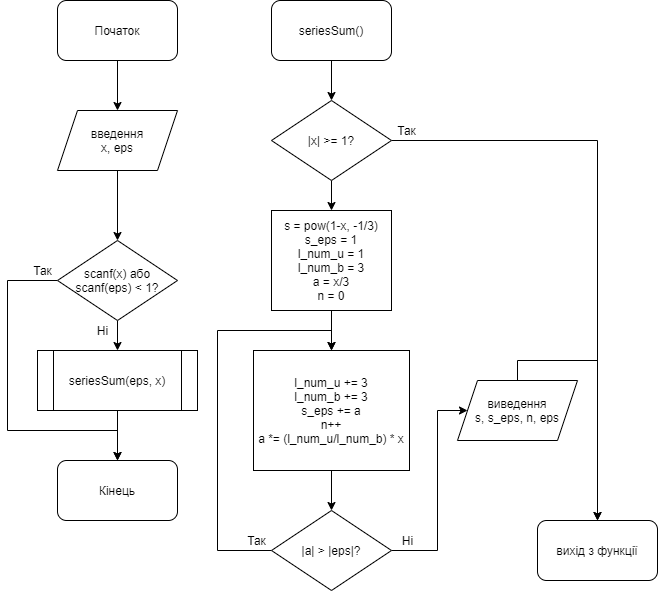
Завдання потребує обчислення суми ряду, залежного від деякого значення x з точністю до деякого епсілон, а тому найбільш зручним способом для подальшого використання програми буде створення окремої функції, що буде приймати значення x та eps.



Оскільки сума ряду обчислюється за формулою вище, то аби знайти його значення з деякою заданою точністю достатньо лише порівняти деякий елемент та eps.

Якщо , то слід зупинити обчислення суми ряду. Останнє виконуватиметься у циклі do-while задля зручності та читабельності коду, але його також виконати за допомогою звичайного циклу while.

Блок-схему алгоритму можна побачити на рисунку 1.



*Рисунок 1. Блок-схема програми*

**2. Код програми та результат виконання програми:**

|  |
| --- |
| #include <stdio.h> #include <windows.h> #include <math.h>  void seriesSum(double eps, double x) {  // часть 1  if (fabs(x) >= 1) {  printf("|x| >= 1. Завершение 5 задания невозможно\n");  return;  }  double s = pow(1 - x, -1.0/3.0);   // часть 2  double s\_eps = 1.0, l\_num\_u = 1.0, l\_num\_b = 3.0, a = (1.0 / 3.0) \* x;  int n = 0;  do {  l\_num\_u += 3.0, l\_num\_b += 3.0, s\_eps += a;  n++;  } while (fabs((a \*= (l\_num\_u / l\_num\_b) \* x)) > fabs(eps));  printf("S = %15.11f\nS\_eps = %15.11f\nn = %d\neps = %15.11f\n======\n", s, s\_eps, n, eps); }  int main() {  SetConsoleOutputCP(CP\_UTF8);  double x, eps;  printf("X Eps через пробел.\n");  if (scanf("%lf", &x) < 1 || scanf("%lf", &eps) < 1) {  printf("Неверный тип данных для x или eps (ожидался double)\n");  return -1;  }  seriesSum(eps, x); } |

На рисунку нижче зображено результат виконання програми.

Изображение выглядит как текст, внешний

Автоматически созданное описание

*Рисунок 2. Результат виконання програми*

**Висновки**

У процесі виконання *лабораторної роботи №3* я розробив алгоритм для обчислення суми ряду з деякою точністю в залежності від наданих значень і реалізував цей алгоритм на *мові програмування C*. Я створив окрему функцію, яка обчислює суму ряду, аби користуватися програмою було зручніше. Алгоритм реалізовано за допомогою цикла do-while, адже, в даному випадку, він є найбільш зручним і допомогає зменшити кількість коду. Всі використані змінні мають тип – *double*, оскільки, обрахування результатів допускає дробові числа. Щоб переконатися що користувач ввів число, порівнюю функцію *scanf* і 1. Якщо функція повернула число менше 1 - тип даних не підійшов для *double.* Виведення данних реалізовано за допомогою форматированного виведення тексту з бібліотеки *iostream*