

Міністерство освіти і науки України
Національний технічний університет України «Київський
політехнічний
інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки
Кафедра автоматизованих систем обробки інформації
і управління

Звіт

з лабораторної роботи No3 з дисципліни
«Основи програмування»
«Дослідження ітераційних циклічних алгоритмів»

Виконав студент ІП-01 Адамчук Антон Іванович

Варіант 19

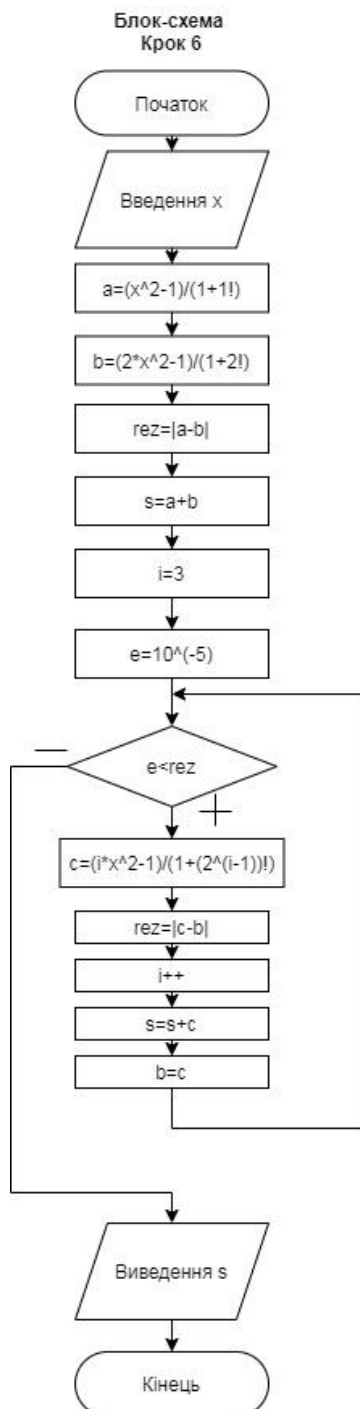
Задача.

19. Обчислити:

$$s = \frac{x^2 - 1}{1 + 1!} + \frac{2x^2 - 1}{1 + 2!} + \frac{3x^2 - 1}{1 + 4!} + \frac{4x^2 - 1}{1 + 8!} + \dots, \text{ для } 0 \leq x \leq 2$$

з точністю до члена ряду, що менше 10^{-5} .

Блок-схема:



Код програми на C++:

```
#include <iostream>

#include<cmath>           // Під'єднання бібліотеки математичних функцій

using namespace std;

int main()
{
    long double a,b;      // Перше та друге значення у виразі
    long double rez;      // Різниця двох послідовних значень у виразі
    long double fact,d;   // Змінні для знаходження факторіала та значення двійки в вказаному степені відповідно
    long double c;        // і-те значення у виразі
    long double x,        // Число на проміжку [0,2]
    e,                   // Змінна для точності обчислення
    s;                   // Змінна для суми всіх послідовних значень
    int i;               // Лічильник

    cout<<"Enter number 0<=x<=2: "; cin>> x;
    if((x<=2) && (x>=0))
    {
        a=(x*x-1)/(2);
        b=(2*x*x-1)/(1+2);
        rez=abs(a-b);
        s=a+b;
        i=3;
        fact=1;
        e=pow(10,-5);

        while(e<rez)      // Доки точність менша ніж різниця двох послідовних чисел виконуємо тіло циклу
        {
            d=pow(2,i-1);  // Піднесення двійки до степеня і-1
            for (int k=1; k<=d; k++)
            {
                fact=fact*k;    // Обчислення факторіалу числа d
            }

            c=(i*x*x-1)/(1+fact); // Обчислення і-того значення у виразі
            rez=abs(c-b);       // Різниця двох послідовних значень у виразі
```



```

    i++;

    s=s+c;           // Збільшення суми на i-тий елемент виразу

    b=c;           // Присвоєння значення змінної c змінній b


    fact=1;         // Присвоєння факторіалу значення 1 для коректності його визначення для наступного
елемента

}

cout << "The sum of all elements to the nearest 10^(-5): " << s << endl << endl; // Виведення суми всіх послідовних значень з точністю до 10^(-5)

}

else cout << "Don't correct data";

}

```

Копії екранних форм:

```

Enter number 0<=x<=2: 1
The sum of all elements to the nearest 10^(-5): 0.413408

Process returned 0 (0x0)   execution time : 1.190 s
Press any key to continue.

```

Обчислимо вручну для перевірки правильності роботи програми:

Вхідні дані : $x=1$

Обробка даних:

- 1) $a = (1 * 1 - 1) / (1 + 1) = 0$ -значення першого виразу у ряді
- 2) $b = (2 * 1 * 1 - 1) / (1 + 2) = 1/3 = 0.33333333$ -значення другого виразу у ряді
- 3) $rez = |0 - 0.33333333| = 0.33333333$ -значення різниці по модулю першого і другого виразів у ряді
- 4) $s = a + b = 0.33333333$ -значення суми першого і другого виразів у ряді
- 5) $i = 3$
- $e = 10^{(-5)}$

Підходимо до циклу і перевіряємо умову:

- 6) $10^{(-5)} < 0.33333333$ true, отже, виконуються наступні дії:
 - $C = (3 * 1 * 1 - 1) / (1 + 4!) = 0.08$ -значення i-того виразу у ряді
 - $Rez = |0.08 - 0.33333333| = 0.253333$ -значення різниці по модулю двох послідовних виразів у ряді
 - $l = 3 + 1 = 4$
 - $S = 0.33333333 + 0.08 = 0.413333$ -значення суми після обчислення ще одного виразу
 - $B = 0.08$
- 7) $10^{(-5)} < 0.253333$ true, отже, виконуються наступні дії:
 - $C = (4 * 1 * 1 - 1) / (1 + 8!) = 7.4 * 10^{(-5)}$

Rez=|0.0000744-0.08|=0.0799256

l=4+1=5

S=0.413333+0.0000744=0.413408

B=0.0000744

8) $10^{-5} < 0.0799256$ true, отже, виконуються наступні дії:

$C = (5 * 1 * 1 - 1) / (1 + 16!) = 1.91179 * 10^{-13}$

Rez=|1.91179*10[^](-13)-0.0000744|=0.00007439

l=5+1=6

S=0.413408+1.91179*10[^](-13)=0.413408

B=1.91179*10[^](-13)

9) $10^{-5} < 0.00007439$ true, отже, виконуються наступні дії:

$C = (6 * 1 * 1 - 1) / (1 + 32!) = 1.9002 * 10^{-35}$

Rez=|1.9002*10[^](-35)-1.91179*10[^](-13)|=1.91179*10[^](-13)

l=6+1=7

S=0.413408+1.9002*10[^](-35)=0.413408

B=1.9002*10[^](-35)

10) $10^{-5} < 1.91179 * 10^{-13}$ false, отже, цикл зупиняється і виконуються дії, що йдуть після цикла:

11) Виведення суми S =0.413408

Вихідні дані: 0.413408

Код програми на Python:

```
import math                                     # Під'єднання бібліотеки математичних функцій

x=float(input("Enter number 0<=x<=2: "))      # Введення змінної для числа 0<=x<=2
if (x<=2) and (x>=0):

    a=(x*x-1)/2                                # Змінна першого значення у виразі
    b=(2*x*x-1)/3                              # Змінна другого значення у виразі
    rez=abs(a-b)                               # Змінна різниці двох послідовних значень у виразі
    s=a+b                                       # Змінна суми двох послідовних значень
    i=3                                         # Змінна лічильник для задання циклу
    e=pow(10,-5)                               # Змінна точності обчислення
    while e<rez:                               # Доки точність менша ніж різниця двох послідовних чисел виконуємо тіло циклу
```



```

d=pow(2,i-1)                # Змінна для збереження значення двійки піднесеної до степеня i-1
fact=math.factorial(d)       # Змінна для визначення факторіалу
c=(i*x*x-1)/(1+fact)         # i-те значення у виразі
rez=abs(c-b)
i=i+1
s=s+c                        # Збільшення суми на i-тий елемент
b=c

print("The sum of all elements to the nearest 10^(-5): %.6f" % (s)) # Виведення суми всіх послідовних значень з точністю до 10^(-5)
else:
    print("Don't correct data")

```

Копії екранних форм:

Підставимо раніше перевірене значення $x=1$

```

Enter number 0<=x<=2: 1
The sum of all elements to the nearest 10^(-5): 0.413408

```

Число дорівнює тому, що ми отримали при перевірці алгоритму вручну.

Висновок. Алгоритм працює при всіх можливих вхідних даних, що відповідають умові. Задача розв'язана та протестована на двох мовах програмування (C++ та Python).