**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

**Лабораторна робота № 2.1**

з дисципліни  
«Алгоритми і структури даних»

Виконав: Перевірила:

студент групи ІМ-22 Молчанова А. А.  
Кушнір Микола Миколайович  
номер у списку групи: 13

Київ 2023

***Постановка задачі***

Дане натуральне число ***n***. Знайти суму перших ***n*** членів ряду чисел,

заданого рекурентною формулою. Розв’язати задачу трьома способами

(написати три програми):

1. в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному спуску;
2. в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному поверненні;
3. в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

Програми повинні працювати коректно для довільного натурального ***n*** включно з ***n*** ***= 1***.

***Завдання для варіанту 13***

*F1 = x; Fi+1 = Fi · x2(2i - 1)2 / (4i2+2i); i > 0;*

*∑Fi = arcsin x, -1 < x < 1.*

***Текст програм***

***Алгоритм 1***

double alg1(  
 double x,  
 unsigned int n,  
 unsigned int i,  
 double currentElement,  
 double currentSum  
 )  
{  
 double item;  
 if (n == 0)  
 item = currentSum;  
 else  
 {  
 if (i == 0)  
 currentElement = x;  
 else  
 {  
 currentElement = currentElement \*  
 (x \* x) \*  
 ((2 \* i - 1) \* (2 \* i - 1)) /  
 (4 \* (i \* i) + 2 \* i);  
 }  
 currentSum += currentElement;  
 item = alg1(x, (n - 1), (i + 1), currentElement, currentSum);  
 }  
 return item;  
}

***Алгоритм 2***

typedef struct  
{  
 double element, sum;  
} structure;  
  
structure alg2(double x, unsigned int n)  
{  
 structure result;  
 if (n == 1)  
 result.sum = result.element = x;  
 else  
 {  
 result = alg2(x, (n - 1));  
 --n;  
 result.element = result.element \*  
 (x \* x) \*  
 ((2 \* n - 1) \* (2 \* n - 1)) /  
 (4 \* (n \* n) + 2 \* n);  
 result.sum += result.element;  
 }  
 return result;  
}

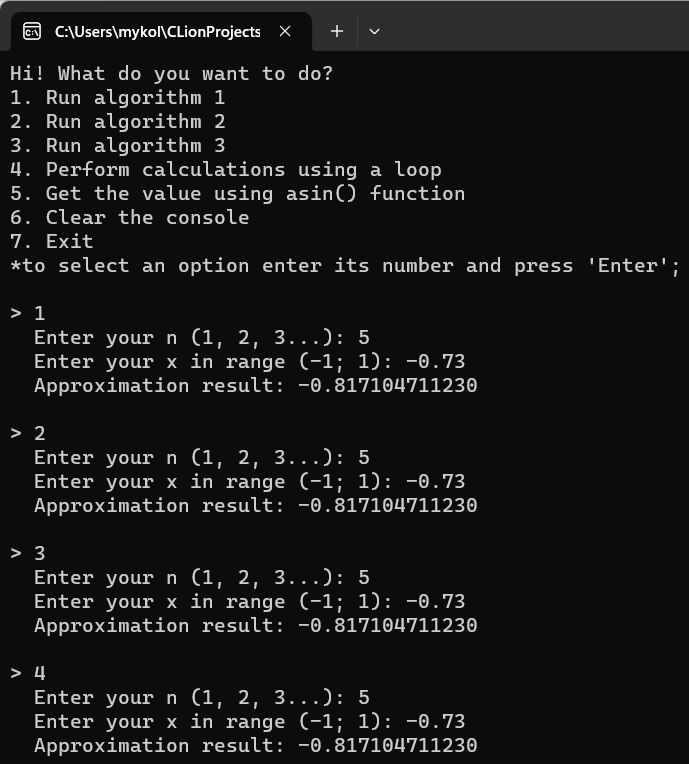
***Алгоритм 3***

double alg3(  
 double x,  
 unsigned n,  
 unsigned i,  
 double currentElement  
 )  
{  
 double sum;  
 if (n == 0)  
 sum = 0;  
 else  
 {  
 if (i == 0)  
 currentElement = x;  
 else  
 {  
 currentElement = currentElement \*  
 (x \* x) \*  
 ((2 \* i - 1) \* (2 \* i - 1)) /  
 (4 \* (i \* i) + 2 \* i);  
 }  
 sum = alg3(x, (n - 1), (i + 1), currentElement);  
 sum += currentElement;  
 }  
 return sum;  
}

***Результати тестування програм***

***Циклічний варіант рішення задачі***

double loop(double x, unsigned n)  
{  
 double element, sum;  
 sum = element = x;  
 for (int i = 1; i < n; ++i)  
 {  
 element = element \*  
 (x \* x) \*  
 (2 \* i - 1) \* (2 \* i - 1) /  
 (4 \* (i \* i) + 2 \* i);  
 sum += element;  
 }  
 return sum;  
}

**

***Обчислення за***

***допомогою програм***

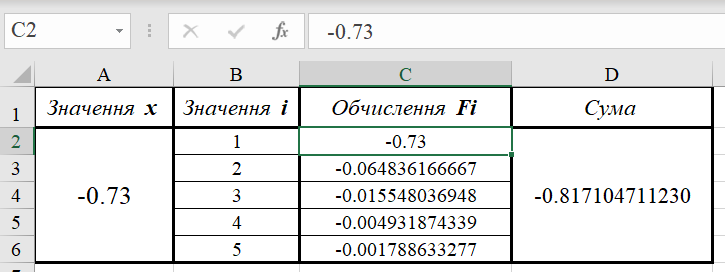
*Результати обчислення за допомогою* ***алгоритму 1***

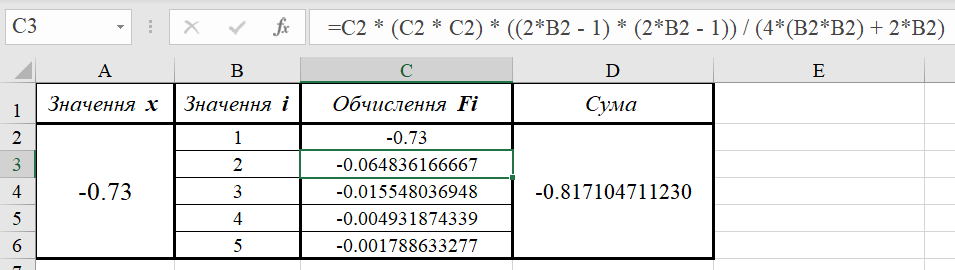
*Результати обчислення за допомогою* ***алгоритму 2***

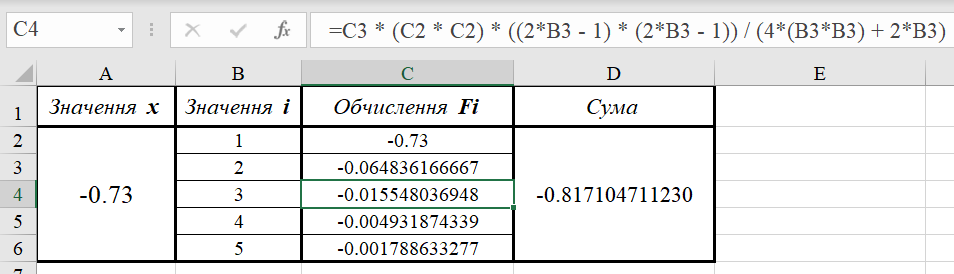
*Результати обчислення за допомогою* ***алгоритму 3***

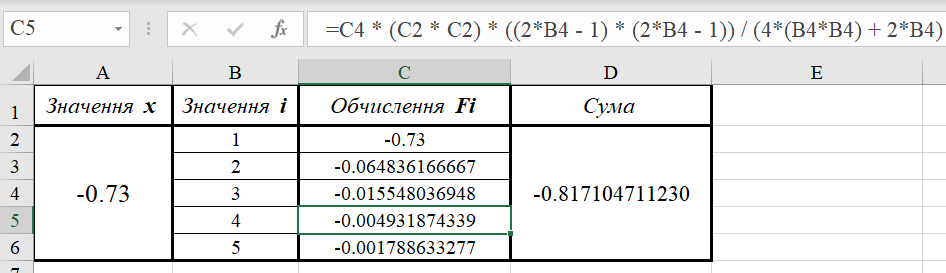
*Результати обчислення* ***циклічною програмою***

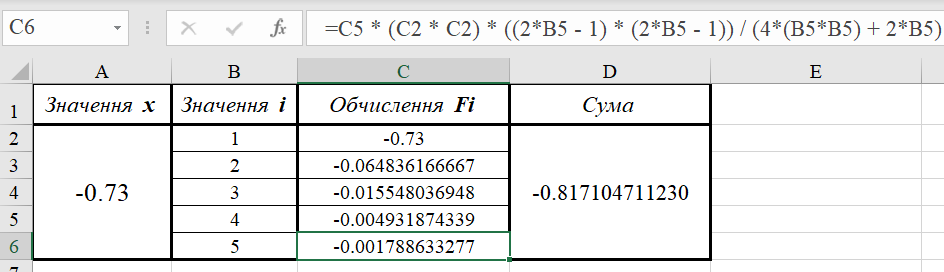
***Результати обчислення на калькуляторі***

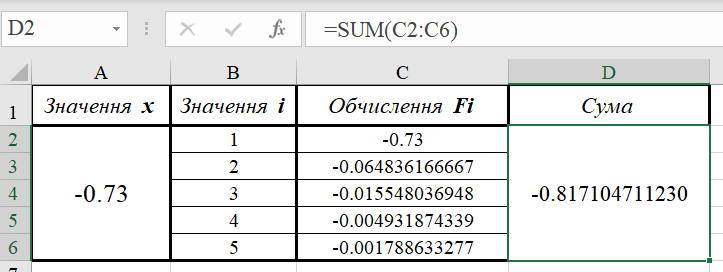












***Аналіз похибок (для n = 5)***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Значення* ***x*** | *Результат апроксимації* | *Значення* ***arcsin(x)*** | *Похибка* | *Абсолютна похибка* |
| -0.99 | -1.292405244179 | -1.429256853470 | -0.136851609291 | 0.136851609291 |
| -0.9 | -1.098909877952 | -1.119769514999 | -0.020859637047 | 0.020859637047 |
| -0.8 | -0.923349414603 | -0.927295218002 | -0.003945803399 | 0.003945803399 |
| -0.7 | -0.774674468963 | -0.775397496611 | -0.000723027648 | 0.000723027648 |
| -0.6 | -0.643387894286 | -0.643501108793 | -0.000113214507 | 0.000113214507 |
| -0.5 | -0.523585195390 | -0.523598775598 | -0.000013580208 | 0.000013580208 |
| -0.4 | -0.411515773968 | -0.411516846067 | -0.000001072099 | 0.000001072099 |
| -0.3 | -0.304692611401 | -0.304692654015 | -0.000000042614 | 0.000000042614 |
| -0.2 | -0.201357920317 | -0.201357920790 | -0.000000000473 | 0.000000000473 |
| -0.1 | -0.100167421161 | -0.100167421162 | -0.000000000001 | 0.000000000001 |
| 0 | 0 | 0 | 0 | 0 |
| 0.1 | 0.100167421161 | 0.100167421162 | 0.000000000001 | 0.000000000001 |
| 0.2 | 0.201357920317 | 0.201357920790 | 0.000000000473 | 0.000000000473 |
| 0.3 | 0.304692611401 | 0.304692654015 | 0.000000042614 | 0.000000042614 |
| 0.4 | 0.411515773968 | 0.411516846067 | 0.000001072099 | 0.000001072099 |
| 0.5 | 0.523585195390 | 0.523598775598 | 0.000013580208 | 0.000013580208 |
| 0.6 | 0.643387894286 | 0.643501108793 | 0.000113214507 | 0.000113214507 |
| 0.7 | 0.774674468963 | 0.775397496611 | 0.000723027648 | 0.000723027648 |
| 0.8 | 0.923349414603 | 0.927295218002 | 0.003945803399 | 0.003945803399 |
| 0.9 | 1.098909877952 | 1.119769514999 | 0.020859637047 | 0.020859637047 |
| 0.99 | 1.292405244179 | 1.429256853470 | 0.136851609291 | 0.136851609291 |

