

**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 №1
з дисципліни
«Алгоритми і структури даних»

Виконав:

студент групи ІМ-43
Костеніч Степан Станіславович
номер у списку групи: 17

Перевірив:

Сергієнко А. М.

Київ 2025

Постановка задачі

Дане натуральне число n . Знайти суму перших n членів ряду чисел, заданого рекурентною формулою. Розв'язати задачу **трьома способами**:

- 1) у програмі використати рекурсивну функцію, яка виконує обчислення i членів ряду, і суми на рекурсивному спуску;
- 2) у програмі використати рекурсивну функцію, яка виконує обчислення i членів ряду, і суми на рекурсивному поверненні;
- 3) у програмі використати рекурсивну функцію, яка виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

При проектуванні програм **слід врахувати наступне**:

- 1) програми повинні працювати коректно для довільного цілого додатного n включно з $n = 1$;
- 2) видимість змінних має обмежуватися тими ділянками, де вони потрібні;
- 3) функції повинні мати властивість модульності;
- 4) у кожному з трьох способів рекурсивна функція має бути одна (за потреби, можна також використати додаткову функцію-обгортку (wrapper function));
- 5) у другому способі можна використати запис (struct) з двома полями (але в інших способах у цьому немає потреби і це вважатиметься надлишковим);
- 6) програми мають бути написані мовою програмування C.

Варіант №17

$$F_1 = x; \quad F_{i+1} = F_i \cdot x^2 / (4i^2 + 2i), \quad i > 0;$$

$$\sum_{i=1}^n F_i = \operatorname{sh} x, \quad |x| < 10^6.$$

Текст програми №1

```
#include <stdio.h>
```

```
double sumSinhV1Series(double x, unsigned int n, unsigned int i, double
fi, double sum) {
    double result = -1;

    if (i >= n) {
        result = sum;
    } else {
        fi *= x * x / (4.0 * i * i + 2.0 * i);
        sum += fi;
        result = sumSinhV1Series(x, n, i + 1, fi, sum);
    }

    return result;
}
```

```
double sinhV1(double x, unsigned int n) {
    const unsigned int i = 1;
    const double fi = x;
    const double sum = 0.0;

    return x + sumSinhV1Series(x, n, i, fi, sum);
}
```

```
int main() {
    double x;
    unsigned int n;

    printf("Enter x ( $|x| < 10^6$ ):");
    scanf("%lf", &x);
    printf("Enter n:");
    scanf("%u", &n);

    printf("\nThe result is %.15lf\n", sinhV1(x, n));

    return 0;
}
```

Текст програми №2

```
#include <stdio.h>

typedef struct {
    double fi;
    double sum;
} Result;

Result sumSinhV2Series(double x, unsigned int n) {
    Result result;

    if (n <= 1) {
        result = (Result) {x, x};
    } else {
        Result prev = sumSinhV2Series(x, --n);
        result.fi = prev.fi * (x * x / (4.0 * n * n + 2.0 * n));
        result.sum = prev.sum + result.fi;
    }

    return result;
}

double sinhV2(double x, unsigned int n) {
    return sumSinhV2Series(x, n).sum;
}

int main() {
    double x;
    unsigned int n;

    printf("Enter x (|x| < 10^6):");
    scanf("%lf", &x);
    printf("Enter n:");
    scanf("%u", &n);

    printf("\nThe result is %.15lf\n", sinhV2(x, n));

    return 0;
}
```

Текст програми №3

```
#include <stdio.h>
```

```
double sumSinhV3Series(double x, unsigned int n, unsigned int i, double
fi) {
    double result = -1;

    if (i >= n) {
        result = 0.0;
    } else {
        fi *= x * x / (4.0 * i * i + 2.0 * i);
        result = sumSinhV3Series(x, n, i + 1, fi);
        result += fi;
    }

    return result;
}
```

```
double sinhV3(double x, unsigned int n) {
    const unsigned int i = 1;
    const double fi = x;

    return x + sumSinhV3Series(x, n, i, fi);
}
```

```
int main() {
    double x;
    unsigned int n;

    printf("Enter x ( $|x| < 10^6$ ):");
    scanf("%lf", &x);
    printf("Enter n:");
    scanf("%u", &n);

    printf("\nThe result is %.15lf\n", sinhV3(x, n));

    return 0;
}
```

Текст програми №4

```
#include <stdio.h>

double sinhLoop(double x, unsigned int n) {
    double result = 0.0;
    double fi = x;

    for (int i = 1; i < n; i++) {
        fi *= x * x / (4.0 * i * i + 2.0 * i);
        result += fi;
    }

    return result + x;
}

int main() {
    double x;
    unsigned int n;

    printf("Enter x (|x| < 10^6):");
    scanf("%lf", &x);
    printf("Enter n:");
    scanf("%u", &n);

    printf("\nThe result is %.15lf\n", sinhLoop(x, n));

    return 0;
}
```

Результати тестування програм

n is set to 5. Enter x ($|x| < 10^6$):0.5

sinh v1: 0.521095

sinh v2: 0.521095

sinh v3: 0.521095

sinh loop: 0.521095

n is set to 5. Enter x ($|x| < 10^6$):10

sinh v1: 5749.858907

sinh v2: 5749.858907

sinh v3: 5749.858907

sinh loop: 5749.858907

n is set to 5. Enter x ($|x| < 10^6$):100

sinh v1: 2775656692339.859375

sinh v2: 2775656692339.859375

sinh v3: 2775656692339.859375

sinh loop: 2775656692339.859375

n is set to 5. Enter x ($|x| < 10^6$):150

sinh v1: 106279232705507.140625

sinh v2: 106279232705507.140625

sinh v3: 106279232705507.140625

sinh loop: 106279232705507.140625

n is set to 5. Enter x ($|x| < 10^6$):200

sinh v1: 1413477094807960.250000

sinh v2: 1413477094807960.250000

sinh v3: 1413477094807960.250000

sinh loop: 1413477094807960.250000

n is set to 5. Enter x ($|x| < 10^6$):9000000

sinh v1: 1067627008929520447790522005150837856630736689055985041408.000000

sinh v2: 1067627008929520447790522005150837856630736689055985041408.000000

sinh v3: 1067627008929520447790522005150837856630736689055985041408.000000

sinh loop: 1067627008929520447790522005150837856630736689055985041408.000000

n is set to 5. Enter x ($|x| < 10^6$):-0.5

sinh v1: -0.521095

sinh v2: -0.521095

sinh v3: -0.521095

sinh loop: -0.521095

n is set to 5. Enter x ($|x| < 10^6$):-10

sinh v1: -5749.858907

sinh v2: -5749.858907

sinh v3: -5749.858907

sinh loop: -5749.858907

n is set to 5. Enter x ($|x| < 10^6$): -10

sinh v1: -5749.858907

sinh v2: -5749.858907

sinh v3: -5749.858907

sinh loop: -5749.858907

n is set to 5. Enter x ($|x| < 10^6$): -100

sinh v1: -2775656692339.859375

sinh v2: -2775656692339.859375

sinh v3: -2775656692339.859375

sinh loop: -2775656692339.859375

n is set to 5. Enter x ($|x| < 10^6$): -150

sinh v1: -106279232705507.140625

sinh v2: -106279232705507.140625

sinh v3: -106279232705507.140625

sinh loop: -106279232705507.140625

n is set to 5. Enter x ($|x| < 10^6$): -9000000

sinh v1: -1067627008929520447790522005150837856630736689055985041408.000000


sinh v2: -1067627008929520447790522005150837856630736689055985041408.000000

sinh v3: -1067627008929520447790522005150837856630736689055985041408.000000

sinh loop: -1067627008929520447790522005150837856630736689055985041408.000000


Результати тестування за допомогою калькулятора

Налаштування:

$F(i,x) = \{ i = 1:x, F(i-1,x) \cdot M(i-1,x) \}$	✕
$M(i,x) = \frac{x^2}{4i^2 + 2i}$	✕
$S(x) = \sum_{i=1}^n F(i,x)$	✕
$n = 5$ 0  10	✕

Обчислень елементів ряду та їх суми

```
n is set to 5. Enter x (|x| < 10^6):0.5  
  
F1: 0.5000000000000000  
F2: 0.0208333333333333  
F3: 0.0002604166666667  
F4: 0.0000015500992063  
F5: 0.0000000053822889  
S(0.5): 0.5210953054814953
```

$a = 0.5$ -10  10	✕
$F(1,a)$ = 0.5	✕
$F(2,a)$ = 0.020833333333333	✕
$F(3,a)$ = 0.000260416666667	✕
$F(4,a)$ = 0.00000155009920635	✕
$F(5,a)$ = $5.3822889109 \times 10^{-9}$	✕
$S(a)$ = 0.521095305481	✕

```
n is set to 5. Enter x (|x| < 10^6):10

F1: 10.0000000000000000
F2: 166.666666666666856
F3: 833.333333333334849
F4: 1984.1269841269845529
F5: 2755.7319223985896315
S(10.0): 5749.8589065255746391
```

$a = 10$

-10

10

$F(1,a)$

=

10

$F(2,a)$

=

166.666666667

$F(3,a)$

=

833.333333333

$F(4,a)$

=

1984.12698413

$F(5,a)$

=

2755.7319224

$S(a)$

=

5749.85890653

```

n is set to 5. Enter x (|x| < 10^6):100

F1: 100.0000000000000000
F2: 166666.6666666666860692
F3: 83333333.3333333432674408
F4: 19841269841.2698440551757812
F5: 2755731922398.5893554687500000
S(100.0): 2775656692339.859375

```

$a = 100$		✕
$F(1,a)$	= 100	✕
$F(2,a)$	= 166666.666667	✕
$F(3,a)$	= 83333333.3333	✕
$F(4,a)$	= $1.9841269841 \times 10^{10}$	✕
$F(5,a)$	= $2.7557319224 \times 10^{12}$	✕
$S(a)$	= $2.7756566923 \times 10^{12}$	✕

```

n is set to 5. Enter x (|x| < 10^6):150

F1: 150.0000000000000000
F2: 562500.0000000000000000
F3: 632812500.0000000000000000
F4: 339006696428.5714111328125000
F5: 105939592633928.5625000000000000
S(150.0): 106279232705507.140625

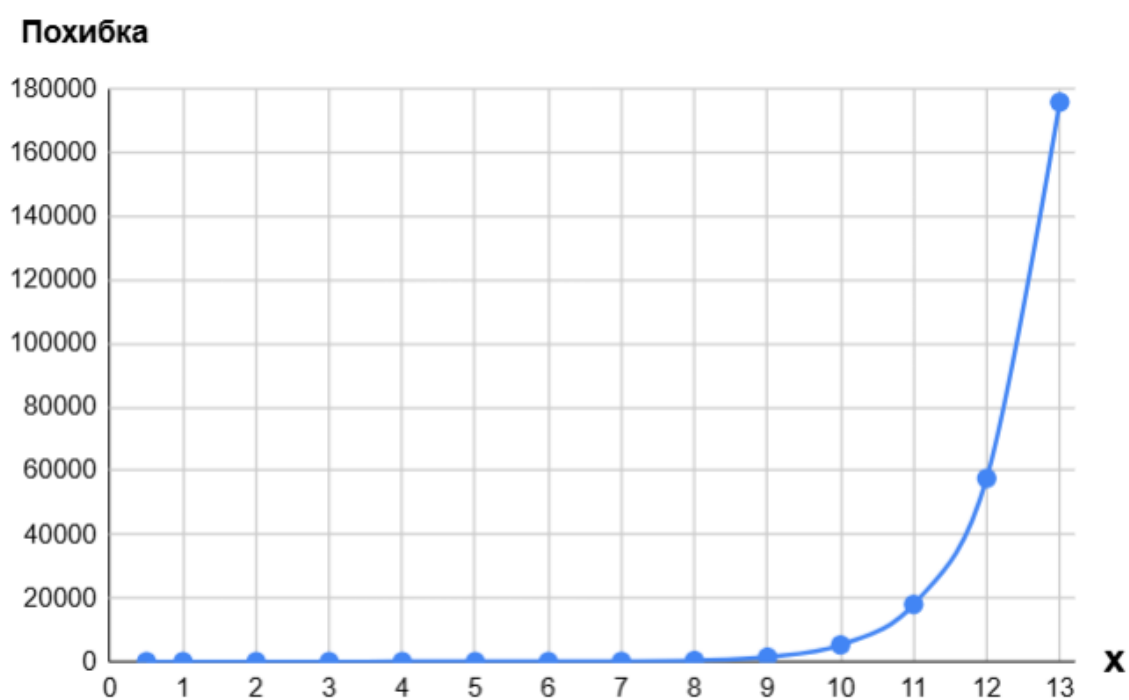
```

$a = 150$		✕
$F(1,a)$	= 150	✕
$F(2,a)$	= 562 500	✕
$F(3,a)$	= 632 812 500	✕
$F(4,a)$	= $3.3900669643 \times 10^{11}$	✕
$F(5,a)$	= $1.0593959263 \times 10^{14}$	✕
$S(a)$	= $1.0627923271 \times 10^{14}$	✕

Таблиця з похибками обчислення функції при $n=5$

Похибка	x
0,0000000000122521	0,5
0,0000000252134662	1
0,0000526477059255	2
0,0047052845527595	3
0,116724957268846	4
1,44492960336192	5
11,5988716559935	6
69,4860113905304	7
338,109160886552	8
1412,33810743993	9
5263,37396817781	10
17997,7719432736	11
57675,3385635727	12
176060,160980733	13

Графік за таблицею



Висновок

Під час виконання лабораторної роботи № 1 я засвоїв теоретичний матеріал та набув практичних навичок створення рекурсивних алгоритмів та написання відповідних їм програм.

Я на практиці закріпив знання рекурсивних алгоритмів, набуті під час лекції, та реалізував розв'язання програми трьома способами: обчислення і членів ряду, і суми на рекурсивному спуску; обчислення і членів ряду, і суми на рекурсивному поверненні; обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

Для перевірки правильності реалізації рекурсивних функцій розробив додатковий варіант рішення із використанням циклу. Тестування показало збіг отриманих результатів, що підтвердило коректність реалізації алгоритмів.

Також було побудовано графік залежності похибки від значення параметра x . Графік показав, що похибка збільшується із ростом x , що пояснюється накопиченням обчислювальних похибок при використанні обмеженого числа членів ряду.

Також навчився використовувати сигналізуючу функцію, щоб дозволити викликам вирішити підзадачі та повернутися до підпрограм, які їх викликали, та уникнути переповнення стекового сегмента і виникнення переривання.

У ході виконання лабораторної роботи на практиці зрозумів та побачив переваги й недоліки рекурсії. Рекурсивні алгоритми у багатьох випадках є простішими і наочнішими. Вони незамінні у випадках, коли їх майже неможливо представити ітераційними алгоритмами, наприклад, в програмах бінарного пошуку, обходу графу, граматичного розбору рядків. Але в той самий час мають значно меншу ефективність ніж еквівалентні їм ітеративні алгоритми з циклами. Час виконання програми збільшується, так само як і вимоги об'єму пам'яті на збереження багатьох екземплярів контексту.

Здобув нові знання щодо `struct` та `typedef struct` для розв'язання задачі другим способом.

Отже, виконання лабораторної роботи № 1 було корисним, дозволило закріпити теоретичні знання та набути практичних навичок в області програмування мовою C.