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 №1.2

з дисципліни
«Алгоритми і структури даних»

Виконав

Студент групи ІМ-34
Никифоров Артем Михайлович
номер у списку групи: 17

Перевірила:

Молчанова А. А.

Київ 2023

Завдання

1. Задане натуральне число n. Вирахувати значення заданої формули за варіантом.

2. Для вирішення задачі написати дві програми:

1) перша програма повинна використовувати для обчислення формули вкладені цикли;

2) друга програма повинна виконати обчислення формули за допомогою одного циклу з використанням методу динамічного програмування.

3. Виконати розрахунок кількості операцій для кожного з алгоритмів за методикою, викладеною на лекції, додавши до неї підрахунок кількості викликів стандартних функцій.

4. Програма має правильно вирішувати поставлену задачу при будь-якому заданому n, для якого результат обчислення може бути коректно представлений типом double.

5. Результуючі дані вивести у форматі з сімома знаками після крапки.

17.	$S = \sum_{i=1}^{n} \frac{\prod_{j=1}^{i} (j + \cos(j))}{4^i - i}$

Текст програми (з використанням вкладених циклів)

```
#include <math.h>
#include <stdio.h>
int main(){
    int n;
    double S = 0;
    int counter = 0;
    printf("input n:");
    scanf("%d", &n);
    for(int i = 1; i <= n; i++){
        double cpi = 1;
        double power = 1;
        for(int j = 1; j <= i; j++){
            cpi *= j+ cos(j);
            counter += 6;
        }
        counter += 2;
        for (int p = 0; p < i; p++) {
            power *= 4;
            counter += 4;
        }
        counter += 2;
        S += cpi / (power-i);
        counter += 8;
    }
    counter += 1;
    printf("S = %.7lf\n", S);
    printf("counter = %d\n", counter);
    return 0;
```

}

Результати тестування програми (з використанням вкладених циклів)

```
input n:1
S = 0.5134341
counter = 23

Process finished with exit code 0
```

$$\sum_{i=1}^{1} \frac{\prod_{j=1}^{i} j+\cos(j)}{4^i - i}$$

NATURAL LANGUAGE | MATH INPUT

Sum

$$\sum_{i=1}^{1} \frac{\prod_{j=1}^{i} (j+\cos(j))}{4^i - i} = \frac{1}{3}(1+\cos(1))$$

Decimal approximation

0.513434101956046572466978869147

...

```
input n:2
S = 0.6876922
counter = 55

Process finished with exit code 0
```

$$\sum_{i=1}^{2} \frac{\prod_{j=1}^{i} j+\cos(j)}{4^i-i}$$

NATURAL LANGUAGE | MATH INPUT

Sum

$$\sum_{i=1}^{2} \frac{\prod_{j=1}^{i} (j+\cos(j))}{4^i-i} = \frac{1}{42}(1+\cos(1))(20+3\cos(2))$$

Decimal approximation

0.687692150514831156600477932843448058508479

...

Partial sums

```
input n:3
S = 0.7680797
counter = 97

Process finished with exit code 0
```

$$\sum_{i=1}^{3} \frac{\prod_{j=1}^{i} j+\cos(j)}{4^i-i}$$

NATURAL LANGUAGE | MATH INPUT

Sum

$$\sum_{i=1}^{3} \frac{\prod_{j=1}^{i} (j+\cos(j))}{4^i-i} = \frac{(1+\cos(1))(1472+84\cos(3)+\cos(2)(309+42\cos(3)))}{2562}$$

Decimal approximation

0.768079688085862193290446712088882169061350851259361938275708831 2

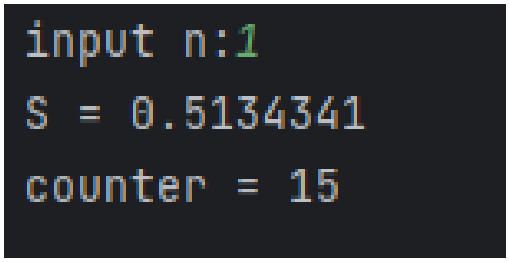
...

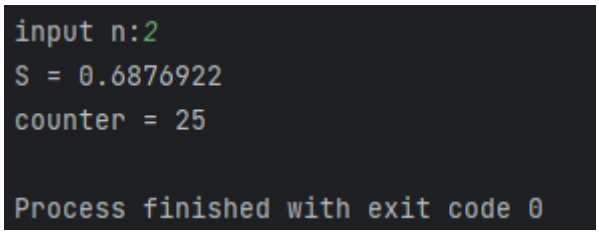
Текст програми (з використанням методу динамічного програмування)

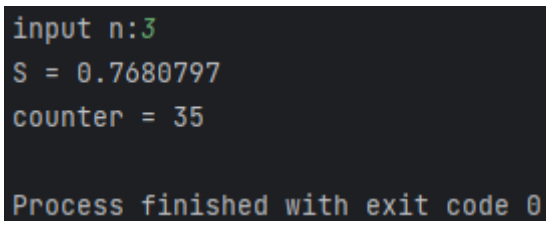
```
##include <math.h>
#include <stdio.h>

int main() {
    int n;
    double S = 0;
    int counter = 0;
    double cpi = 1;
    double power = 1;
    printf("input n:");
    scanf("%d", &n);
    for (int i = 1; i <= n; i++) {
        power *= 4;
        cpi *= i + cos(i);
        S += cpi / (power - i);
        counter += 10; // i <= n, i++, *=, *= , +, cos, +=, /, -, jmp
    }
    counter += 5; // =, =, =, i = 1, i <= n
    printf("S = %.7lf\n", S);
    printf("counter = %d\n", counter);
    return 0;
}
```

Результати тестування програми (з використанням методу динамічного програмування)



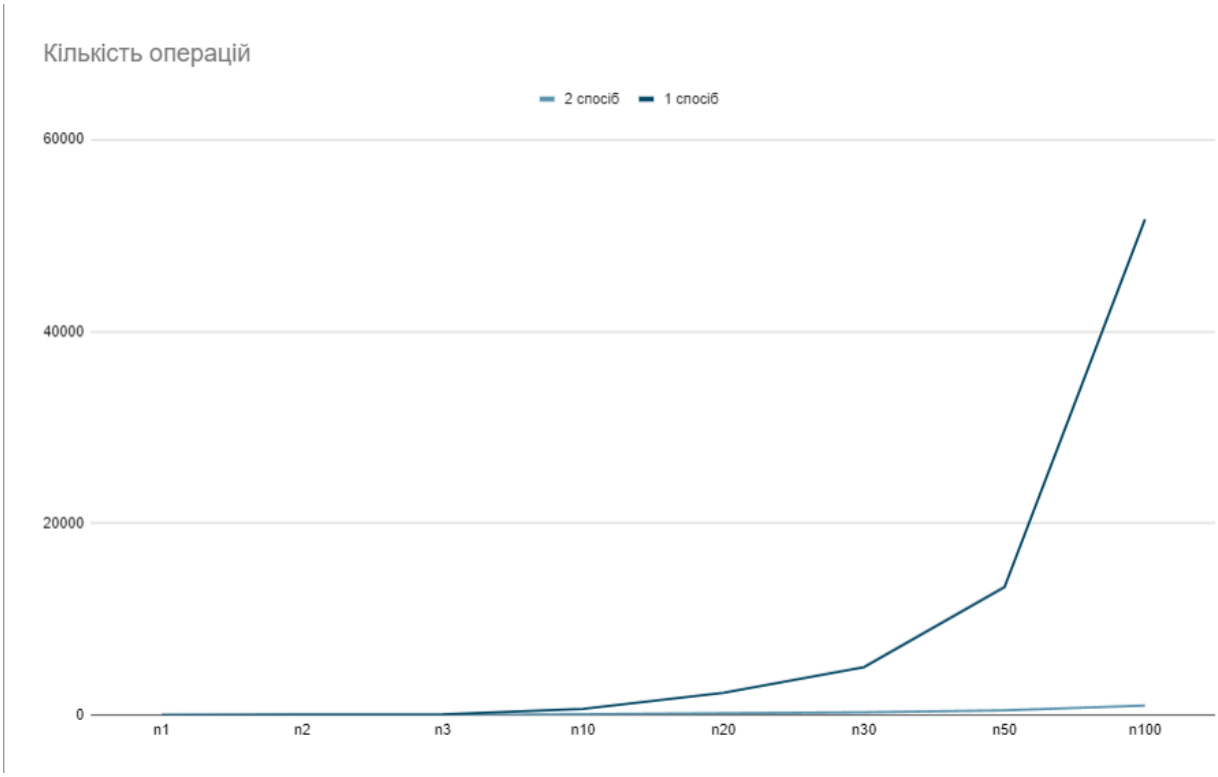




Таблиця тестування програми

n		1	2	3	10	20	30	50	100
Кількість операцій	1 спосіб	23	55	97	671	2341	5011	13351	51701
	2 спосіб	15	25	35	105	205	305	505	1005

Графік за результатами таблиці



Висновок:

Звичайно, метод динамічного програмування набагато практичніший ніж використання вкладених циклів, тому що він дає нам змогу зробити менше обчислень (операцій) тим самим зекономивши ресурси комп'ютера.