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:

студент групи ІМ-22
Коваленко Михайло Володимирович
номер у списку групи: 11

Перевірила:

Молчанова А. А.

Київ 2023

ЛАБОРАТОРНА РОБОТА №2.1.

РЕКУРСИВНІ АЛГОРИТМИ

Мета лабораторної роботи

Метою лабораторної роботи No 2.1 є засвоєння теоретичного матеріалу та набуття практичного досвіду створення рекурсивних алгоритмів та написання відповідних їм програм.

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.

Варіант 11

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

$$\sum F_i = \sin x.$$

Текст програм

1. Обчислення і членів ряду, і суми на рекурсивному спуску (RecursiveDescent.c):

```c
#include <stdio.h>

double multiply(int i, double x);

double recursionSum (double previousValue, int iteration, double sum, double
x, int n) {
    double f;
    if (iteration == 1) {
        f = x;
    } else {
        f = (-1) * previousValue * multiply(iteration - 1, x);
    }
    sum += f;
    if (iteration == n) {
        return sum;
    } else {
        return recursionSum (f, iteration + 1, sum, x, n);
    }
}

double multiply (int i, double x) {
    return (x*x)/((4*i*i)+(2*i));
}

double recursion(double x, int n){
    return recursionSum(0,1,0,x,n);
}

int main() {
    double x;
    printf("Input x:\n");
    scanf("%lf" , &x);
    int n;
    printf("Input n:\n");
    scanf("%d" , &n);
    double sum = recursion(x, n);
    printf("First program output: %.30lf", sum);
    return 0;
}
```

2. Обчислення і членів ряду, і суми на рекурсивному поверненні (RecursiveReturn.c):

```c
#include <stdio.h>

double multiply (int i, double x);

double summationRecursion (int n, double *sum, double x) {
    if (n == 1) {
        *sum += x;
        return x;
    }
    double f = (-1) * summationRecursion(n - 1, sum, x) * multiply(n - 1, x);
    *sum += f;
    return f;
```

```
}

double multiply (int i, double x) {
    return (x*x)/((4*i*i)+(2*i));
}

double summation (double x, int n, double *sum) {
    *sum = 0;
    return summationRecursion(n, sum, x);
}

int main() {
    double x;
    printf("Input x:\n");
    scanf("%lf" , &x);
    int n;
    printf("Input n:\n");
    scanf("%d" , &n);
    double sum;
    summation(x, n, &sum);
    printf("Second program output: %.30lf", sum);
    return 0;
}
```

3. Обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні (Mixed.c):

```
#include <stdio.h>

double multiply (int i, double x);

double recursionSum (double previousValue, int iteration, double x, int n) {
    double f;
    if (iteration == 1) {
        f = x;
    } else {
        f = (-1) * previousValue * multiply(iteration - 1, x);
    }
    if (iteration == n) {
        return f;
    } else {
        double sum = f + recursionSum(f, iteration + 1, x, n);
        return sum;
    }
}

double multiply (int i, double x) {
    return (x*x)/((4*i*i)+(2*i));
}

double recursion(double x, int n){
    return recursionSum(0,1,x,n);
}

int main() {
    double x;
    printf("Input x:\n");
    scanf("%lf" , &x);
    int n;
    printf("Input n:\n");
```

```
    scanf("%d" , &n);
    double sum = recursion(x, n);
    printf("Third program output: %.30lf", sum);
    return 0;
}
```

4. Циклічний варіант рішення задачі для тестування (LoopVersion.c):

```
#include <stdio.h>

double multiply(int i, double x);

double recursion (double x, int n) {
    double sum = x;
    double f = x;
    for (int i = 2; i <= n; i++) {
        f *= (-1) * multiply(i - 1, x);
        sum += f;
    }
    return sum;
}

double multiply (int i, double x) {
    return (x*x)/((4*i*i)+(2*i));
}

int main() {
    double x;
    printf("Input x:\n");
    scanf("%lf" , &x);
    unsigned int n;
    printf("Input n:\n");
    scanf("%d" , &n);
    double sum = recursion(x, n);
    printf("Test output: %.30lf", sum);
    return 0;
}
```

Тестування програм

Дані з калькулятора:

Input

sin(1)

Decimal approximation

0.841470984807896506652502321630298999622563060798371065672751709 9

...

Результати обчислення програм:

1.
```
D:\Coding\CLionProjects\lab2.3\cmake-build-debug\lab2_3.exe
Input x:
1
Input n:
5
First program output: 0.841471009700176408863114829728
Process finished with exit code 0
```
2.
```
D:\Coding\CLionProjects\lab2.3(2)\cmake-build-debug\lab2_3_2_.exe
Input x:
1
Input n:
5
Second program output: 0.841471009700176408863114829728
Process finished with exit code 0
```
3.
```
D:\Coding\CLionProjects\lab2.3(3)\cmake-build-debug\lab2_3_3_.exe
Input x:
1
Input n:
5
Third program output: 0.841471009700176408863114829728
Process finished with exit code 0
```
4.
```
D:\Coding\CLionProjects\lab2.3(4)\cmake-build-debug\lab2_3_4_.exe
Input x:
1
Input n:
5
Test output: 0.841471009700176408863114829728
Process finished with exit code 0
```

Обчислення похибок

X	Табличне значення	Отримане значення	Похибка
0,5	0,479426	0,479425539	4,61384E-07
1	0,841470985	0,84147101	2,48923E-08
1,5	0,997495	0,997497123	2,12263E-06
2	0,909297427	0,909347443	5,00159E-05
2,5	0,598472	0,599046199	0,000574199
3	0,141120008	0,1453125	0,004192492
3,5	-0,350783	-0,328388996	0,022394004
4	-0,756802495	-0,661728395	0,0950741
4,5	-0,97753	-0,639023699	0,338506301
5	0,087155743	0,089630181	0,002474438
5,5	-0,70554	2,195305325	2,900845325
6	0,104528463	7,028571429	6,924042965