IAX0584 Programming II
Recursion
Homework no.2

Ivan Symotiuk
223686MVEB

Supervisor: Associate Professor Vladimir Viies

Copyright declaration

I have prepared this work independently. All the work of other authors used in the preparation of the work, important points of view, data from literary sources, and the seller are cited.

Ivan Symotiuk

Contents

Setting up a task

The task is to create a program that takes the initial value x (between -1 and 1) and the deviation value ε (*eps* for later, between 0 and 1) and evaluates items of sequence A using recursion. The A is formed as

$A_1 = x$

$A_2 = \frac{1}{2}\frac{x^3}{3}$

$A_3 = \frac{1}{2}\frac{3}{4}\frac{x^5}{5}$

Recursively this sequence can be represented as

$A_{i+1} = A_i \frac{(2i+1)^2 x^2}{(2i+2)(2i+3)}$

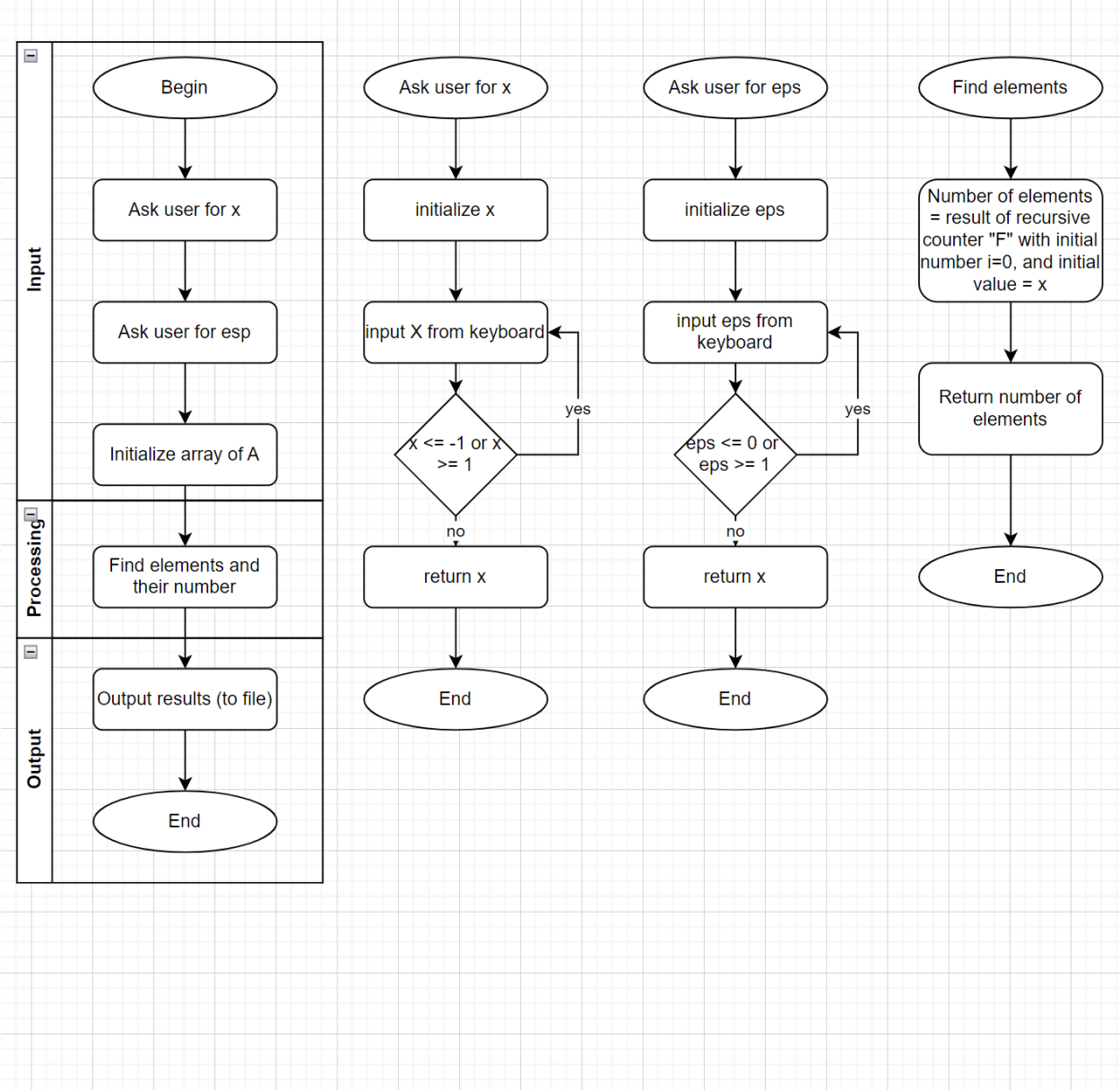
Assuming that “i” starts from 0, where $A_0 = x$

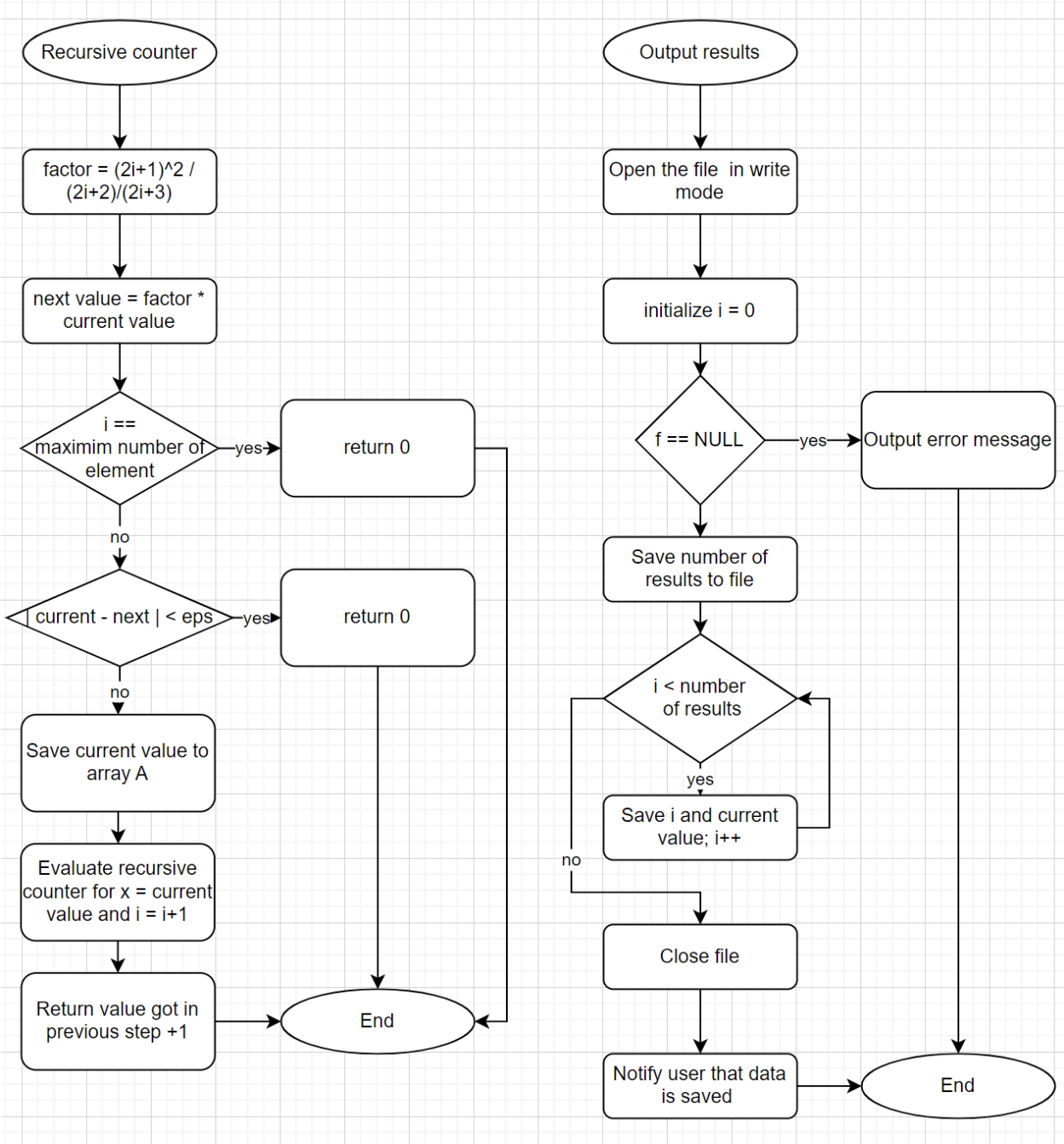
The program should calculate numbers until i = 15 or $|A_{i+1} - A_i| \leq \varepsilon$ and write down them into the file.

For example, when x=0.8 and eps=0.0001 the result will be:

F: 9
1: 0.800000
2: 0.085333
3: 0.024576
4: 0.009362
5: 0.004078
6: 0.001922
7: 0.000954
8: 0.000491
9: 0.000260

Algorithm


Input
Begin
Ask user for x
Ask user for esp
Initialize array of A
Processing
Find elements and their number
Output
Output results (to file)
End
Ask user for x
initialize x
input X from keyboard
x <= -1 or x >= 1
yes
no
return x
End
Ask user for eps
initialize eps
input eps from keyboard
eps <= 0 or eps >= 1
yes
no
return x
End
Find elements
Number of elements = result of recursive counter "F" with initial number i=0, and initial value = x
Return number of elements
End


Recursive counter
factor = (2i+1)^2 / (2i+2)/(2i+3)
next value = factor * current value
i == maximim number of element
yes
return 0
no
| current - next | < eps
yes
return 0
no
Save current value to array A
Evaluate recursive counter for x = current value and i = i+1
Return value got in previous step +1
End
Output results
Open the file in write mode
initialize i = 0
f == NULL
yes
Output error message
Save number of results to file
i < number of results
yes
Save i and current value; i++
no
Close file
Notify user that data is saved
End

Folder structure

The code of the program is divided into files. The root of the project will look like this:

/project
- main.c
- F.txt
- functions.h
- functions.c

The *main.c* file is located at the root of the project and is the entry point of the program. *F.txt* is the output file.

File *functions.c* contains all necessary declarations of functions and *functions.h* contains their prototypes and structure declaration

Source code

In the next sections, there is a source code of the program

main.c

```c
#include "functions.h"
int main()
{
    double x = inputX();
    double eps = inputEps();
    double a[MAX];

    int count = findElements(x, eps, a);

    output(count, a);
}
```

functions.h

```c
#ifndef FUNCTIONS_H
#define FUNCTIONS_H


#include <stdio.h>

#define OUTPUT_FILE "F.txt"
#define MAX 15

double inputEps();
double inputX();
void output(int, double[]);
int findElements(double, double, double[]);


#endif
```

functions.c

```c
#include "functions.h"

int f(double, int, double, double, double[]);

double inputEps(){
    double eps;
    do {
    printf("Enter eps: ");
    scanf("%lf", &eps);
    } while (eps <= 0 || eps >= 1);
    return eps;
}

double inputX(){
    double x;
    do {
        printf("Enter x: ");
        scanf("%lf", &x);
    } while (x <= -1 || x >= 1);
    return x;
}

void output(int count, double results[]){
    FILE *f = fopen(OUTPUT_FILE, "w");
    int i;

    if (f == NULL)
    {
        printf("Error opening file!\n");
        return;
    }

    fprintf(f, "F: %d\n", count);

    for(i = 0; i < count; i++)
    {
        fprintf(f, "%d: %lf\n", i+1, results[i]);
    }
```

```
    fclose(f);
    printf("Results saved to %s\n", OUTPUT_FILE);
}

int findElements(double x, double eps, double results[]){
    int count = f(x, 0, x, eps, results);
    return count;
}

int f(double current, int i, double x, double eps, double results[]){
    double factor = (2*(double)i + 1)*(2*(double)i + 1)/(2*(double)i +
2)/(2*(double)i + 3)*x*x;
    double next = current * factor;

    if(i == MAX){
        return 0;
    } else if ( -eps < current - next && current - next < eps){
        return 0;
    } else {
        results[i] = current;
        return f(next, i+1, x, eps, results) + 1;
    }
}
```

Code Explanation

The program asks user for x and esp values and initializes array to store data.

Then it calls the function "*findElements"* which takes *x*, *eps*, and *a* as parameters because they will be used in the further calculation. This function returns a number of elements (either index of element where esp condition is met, or limit 15)

The function calls a recursive function *f* which takes these parameters: *eps, x, a*$_i$ (current index of element) and *current* - last calculated value. Tha i evaluates the factor (quotient of next and current element) and next element. After that, the function checks if the last index is smaller than 15 (the maximum number of elements in sequence) and if the absolute value of *current-next* is smaller than *eps.* These are the exit conditions. If they are met the functions returns 0 (which means that no more calculations will be done).

If not, the recursive function calls itself, but now for the *next* value, and *i = i+1*. The return value will be the number of further calculations, so we add 1 to it and return it.

In the end, the program saves the number of results and values to the file "F.txt"