НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии
Дисциплина: «Архитектура вычислительных систем»

РАЗРАБОТАТЬ ПРОГРАММУ ЧИСЛЕННОГО ИНТЕГРИРОВАНИЯ

Пояснительная записка

Выполнил:

Асатиани Тимур,

студент гр. БПИ199.

Москва
2020

Содержание

1. Текст задания

Разработать программу интегрирования функции y = a + b*x^3x, (задаётся целыми числами определённом диапазоне целых (задаётся так же) методом трапеций (шаг 1)

2. Применяемые расчетные методы

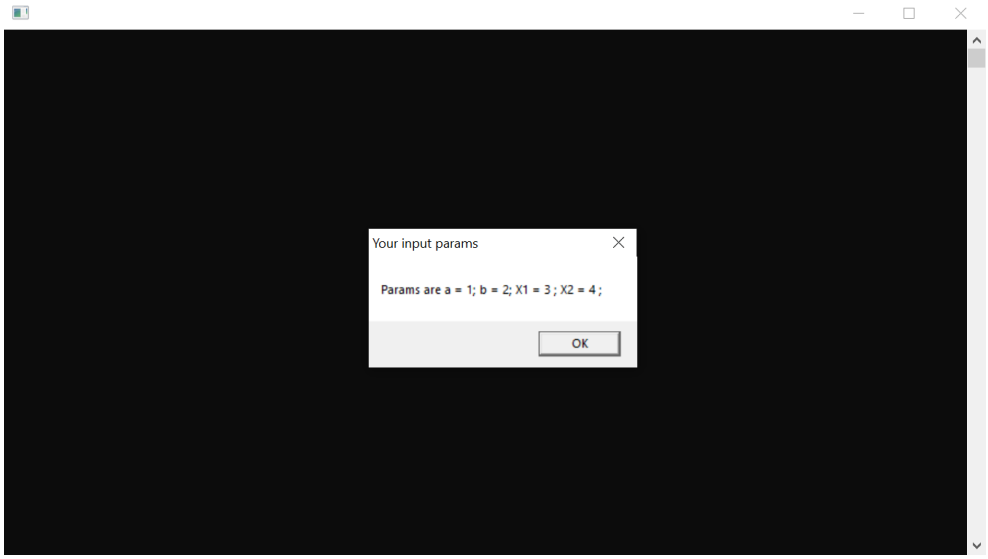
2.1. Теория решения задания

2.2. Дополнительный функционал программы

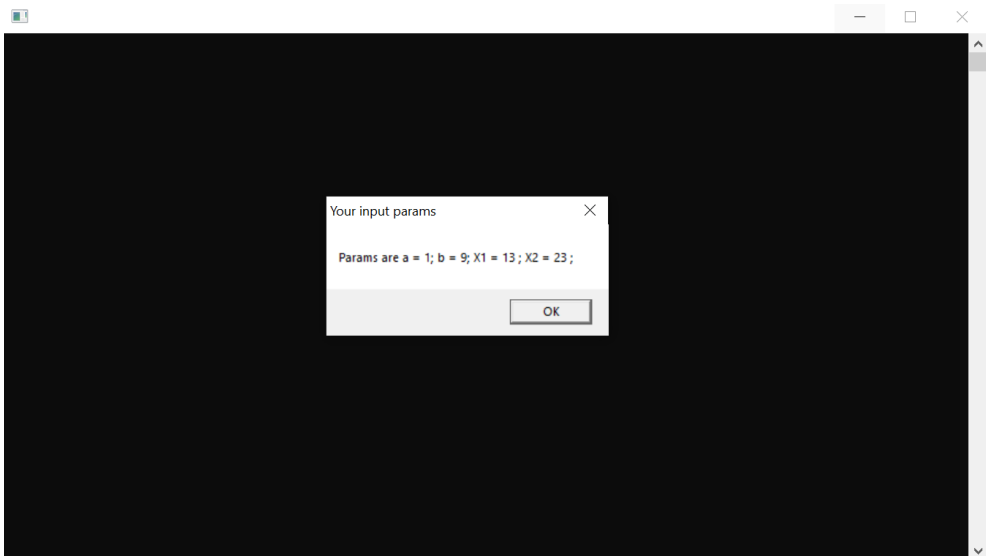
3. Тестирование программы

3.1. Корректные значения

Тут как в пми к курсачу: скрины с вводом и выводом которые подписываются как ниже



1. Рисунок 1.



2. Рисунок 2.

3.2. Некорректные значения

3.3.

ПРИЛОЖЕНИЕ 1

Список литературы

ПРИЛОЖЕНИЕ 2

Код программы

```
format          PE console 4.0                                          ; ?????????? ??????????.
entry start                                                         ; ????? ?????
include "win32ax.inc"

section '.idata' import data readable writeable

library kernel32,'KERNEL32.DLL',\
                user32,'USER32.DLL',\
                gdi32,'GDI32.DLL',\
                advapi32,'ADVAPI32.DLL',\
                comctl32,'COMCTL32.DLL',\
                comdlg32,'COMDLG32.DLL',\
                shell32,'SHELL32.DLL',\
                wsock32,'WSOCK32.DLL',\
                msvcrt,'MSVCRT.DLL'

                import_kernel32
                import_user32
                import_gdi32
                import_advapi32
                import_comctl32
                import_comdlg32
                import_shell32
                import_wsock32

                all_api



import msvcrt,\
         sprintf, 'sprintf', sscanf, 'sscanf'

.data
        endl FIX 10, 13 ;endline \n for strings; ÔÂ¤ÂÎÓÒ ÒÚ¤ÓÍË
        launchstr db "This program will calculate definite integral a+b*x^3 on [x1;x2]",endl,"using the method
of rectangles(width = 1)",endl, "Made by Asatiani Timur, BSE199_2, 2020",0
        debugstr db "debug"
        message db 256 DUP(?)
        commandLine dd ?
        result_integral dd 0
        h dd 1; Ë¤ËÌ‡ ÓÚ¤ÂÁÍ‡; width of the rectangle
        XX1 dd ?
        XX2 dd ?
        AA dd ?
        BB dd ?

macro show_result{
        mov [result_integral], esi
        invoke sprintf, message,  "The calculated integral is %d",[result_integral]
        invoke MessageBox, 0, message, "Result", MB_OK
        invoke ExitProcess, 0
}
macro show_help{
      invoke MessageBox, 0,"Be careful: Usage: in cmd> filename [a] [b] [x1] [x2] X1 <= X2, A and B != 0",
"Help", MB_OK
      invoke ExitProcess, 0
}
macro overflow{
      invoke MessageBox, 0,"Overflow due to big numbers", "Error", MB_OK
      invoke ExitProcess, 0
}
macro check_collect_input{

     mov eax, [AA]
     mov ebx, [BB]
     mov ebx, [XX1]
    ; invoke MessageBox, 0, launchstr, "Welcome", MB_OK
     cmp ebx,[XX2];Ô¤Ó,Â¤Í‡ x1 = x2
     jge end_help; x1 >= x2

     OR eax, [BB];check a and b != 0
     cmp eax, 0
     je end_help

}
macro input{
        cinvoke GetCommandLine
        mov [commandLine], eax
        cinvoke sscanf,[commandLine], '%*s %d %d %d %d', AA,BB,XX1,XX2
        invoke sprintf, message,  "Params are a = %d; b = %d; X1 = %d ; X2 = %d ;",[AA],[BB], [XX1],[XX2]
        invoke MessageBox, 0, message, "Your input params", MB_OK

}
```

```
macro check_overflow{
                cmp OF, 1b
                je overflow_occured
}
macro calc_point1{
                mov edi, ebx
                imul edi, ebx
                imul edi, ebx ; here is x^3
                imul edi, [BB];*b
                add edi, [AA];  = a+b*x3
}
macro calc_point2{

                mov ebp, ebx
                imul ebp, ebx
                imul ebp, ebx ; here is x^3
                imul ebp,[BB];*b
                add ebp, [AA];  = a+b*x^3

}
.code
    start:
        invoke MessageBox, 0, launchstr, "Welcome", MB_OK  ;launch the start welcome window; Á‡ÔÛÒÍ‡ÂÏ ÓÍÌÓ
Ô◆Ë,ÂÚÒÚ,Ë˘
        input;launch input
        check_collect_input; checking values

        ;main section
        ;while xx1 < xx2
        mov esi, 0
        mov ebx, [XX1]
        start_cycle:

                cmp ebx,[XX2]
                jge cycle_exit ;check if cycle is over


                ;previous x1
                calc_point1
                inc ebx ;increment ebx
                calc_point2; calc for next x


                cmp edi,ebp ;compare previous and next
                jle add_left

                add_right:
                       add  esi, ebp;added right value because it is lower
                       jmp start_cycle
                add_left:
                       add  esi, edi;added left value because it is lower
                       jmp start_cycle
        cycle_exit:;label for exit
                show_result;show calculated result

        end_help:;label for exit because of wrong arguments
                show_help
        overflow_occured:
                overflow
```