Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Кубанский государственный университет»

Факультет компьютерных технологий и прикладной математики

Кафедра прикладной математики

**ОТЧЕТ О ПРОХОЖДЕНИИ УЧЕБНОЙ ПРАКТИКИ**

**научно-исследовательской работы**

**(получение первичных навыков научно-исследовательской работы)**

период с 06.07.2023 г. по 19.07.2023 г.

Лаптев Максим Николаевич

*(Ф.И.О. студента)*

студента 23 группы 2 курса ОФО

Направление подготовки 01.03.02 Прикладная математика и информатика

Руководитель учебной практики

декан факультета компьютерных технологий

и прикладной математики,

к.ф.-м.н., доцент Колотий А.Д.

*ученое звание, должность (подпись) (Ф.И.О)*

Оценка по итогам защиты практики: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

*(дата)*

Краснодар 2023 г.

СОДЕРЖАНИЕ

[1 Условие задачи 3](#_gjdgxs)

[2 Описание алгоритма решения и структур хранения данных 3](#_30j0zll)

[3 Описание входных и выходных данных 4](#_1fob9te)

[4 Текст программы 5](#_3znysh7)

[5 Тестовый набор 9](#_2et92p0)

# **1 Условие задачи**

1. Дано уравнение: , где функция непрерывна на отрезке . Написать программу для вычисления интеграла при помощи *метода средних прямоугольников* с заданной точностью и найти решение уравнения *методом Ньютона* с заданной точностью .

# **2 Описание алгоритма решения и структур хранения данных**

Для решения программы были созданы следующие функции:

1. double f(double t, int switchcase),
2. double F(double a, double x, double b, double eps),
3. double srednie\_priamoug(double a, double x, int n),
4. double primaug(double a, double x, double eps),
5. double newton(double a, double b, double eps1, double eps2),
6. double proizvod(double a, double x, double b, double eps).

Функция 1 f(t, switchcase) определяет вид подынтегральной функции в зависимости от значения switchcase.

Функция 2 F(a, x, b, eps) возвращает значение интеграла f(t) с точностью eps на отрезке [a, x], вычитая в конце значение b.

Функции 3 и 4: srednie\_priamoug(a,b,eps) и primaug(a,b,n) реализуют метод Средних прямоугольников для вычисления интеграла функции f(t) на отрезке [a;x]. Функция primaug(a,b,n) работает доо момента, пока не приблизимся к точности eps.

Таким образом функция srednie\_priamoug(a,b,eps) передает шаг в функцию primaug(a,b,n), где по формуле считается интеграл, а после в функции primaug(a,b,n) происходит сравнение приближений с заданной точностью, чтобы добиться нужного результата.

Функция 5 newton(double a, double b, double eps1, double eps2) реализует метод Ньютона для вычисления корня уравнения F(a, x, b, eps)=0 на отрезке от a до b включительно. Мы объявляем переменные x0, x1 в пределах интервалов a и b. А так же объявляем переменные h, f1, x, F0, f0, xx. Эти переменные нужны для реализации метода Ньютона. В этой функции считаются значения функции и её производной в точке.

Внутри цикла мы сохраняем значение xx, чтобы позже заменить его на значение следующей формулы метода Ньютона:

Функция 6 считает производную функций, которые понадобятся в процессе программы

В функции int WINAPI WinMain задаются значения a, b, eps1 и eps2, происходит вызов вышеописанных функций, а также рисуется графический интерфейс благодаря функциям Windows API.

# **3 Описание входных и выходных данных**

Входные параметры: переменные a – нижний предел интегрирования и b – параметр в уравнении типа string, который преобразуется в double, – значение точности для метода Средних прямоугольников и – значение точности для метода Ньютона типа string, которое преобразуется в double.

На выходе программа выдаёт результат решения уравнения x с заданной точностью и типом double.

# **4 Текст программы**

#include <windows.h>

#include <stdlib.h>

#include <string.h>

#include <tchar.h>

#include <string>

#include <cmath>

#define DlgIndexNumber1 1

#define DlgIndexNumber2 2

#define DlgIndexNumber3 3

#define DlgIndexNumber4 4

#define BufferSize 100

using namespace std;

int switchcase = 1; //Для выбора функций

//Глобальных переменныЕ

static TCHAR szWindowClass[] = \_T("Pract");

static TCHAR szTitle[] = \_T("Windows Desktop");

static TCHAR szPole[] = \_T("edit");

static HWND hStatic;

HINSTANCE hInst;

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

double f(double t, int switchcase) {

switch (switchcase)

{

case 1:

{return t \* pow(t - 9, 3); } break; // ФУНКЦИЯ 1

case 2:

{return t + 1; } break; // ФУНКЦИЯ 2

case 3:

{return exp(t) + 1; } break; // ФУНКЦИЯ 3

}

}

//Вычисление интеграла методом Средних прямоугольников

double srednie\_priamoug(double a, double b, int n) { //основной алгоритм решения интеграла

double h = (b - a) / n; // Шаг интегрирования

double sum = 0.0;

for (int i = 0; i < n; i++) {

double x = a + (i + 0.5) \* h; // Середина каждого прямоугольника

sum += f(x, switchcase);

}

return sum \* h;

}

double primaug(double a, double b, double eps) { // Интегрирование до момента, пока не приблизимся к точности эпс

int n = 1;

double s, s1 = srednie\_priamoug(a, b, n); //Начальное значение для дальнейшего решения

do {

s = s1; //Второе приближение функции

n = 2 \* n; //Увеличение числа шагов в два раза

s1 = srednie\_priamoug(a, b, n);

} while (fabs(s1 - s) > eps); //Сравнение приближений с заданной точностью

return s1;

}

//Функция для связки интегрирования с решением уравнения и отниманием константы

double F(double a, double x, double b, double eps) {

return primaug(a, x, eps) - b;

}

double proizvod(double a, double x, double b, double eps) { // Функция Производной

return (F(a, x + eps, b, eps) - F(a, x - eps, b, eps)) / (2.0 \* eps);

}

//Функция, вычисляющая решение уравнения методом Ньютона

double newton(double a, double b, double eps1, double eps2)

{

double F0, f0;

int x;

double f1;

double h = 0.1;

double xx;

double x0, x1, x2 = 0, tmp;

x0 = a; // Левая граница

x1 = b; // Правая граница

do

{

f0 = F(a, x0, b, eps1);

F0 = proizvod(a, x0, b, eps1);

xx = x0 - (f0 / F0);

x0 = xx;

f1 = F(a, x0, b, eps1);

} while (fabs(f0) >= eps1); // Ищем точку

return xx;

}

// Стартовая функция

int WINAPI WinMain(

\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPSTR lpCmdLine,

\_In\_ int nCmdShow

)

{

WNDCLASSEX wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(wcex.hInstance, IDI\_APPLICATION);

wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = NULL;

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, IDI\_APPLICATION);

if (!RegisterClassEx(&wcex))

{

MessageBox(

NULL,

\_T("Call to RegisterClassEx failed!"),

\_T("Windows Desktop Guided Tour"),

NULL);

return 1;

}

// Сохраняем дескриптор экземпляра в глобальной переменной

hInst = hInstance;

HWND hWnd = CreateWindowEx(

WS\_EX\_OVERLAPPEDWINDOW,

szWindowClass,

szTitle,

WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT,

400, 500,

NULL,

NULL,

hInstance,

NULL

);

if (!hWnd)

{

MessageBox(NULL,

\_T("Call to CreateWindow failed!"),

\_T("Windows Desktop Guided Tour"),

NULL);

return 1;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

MSG msg;

while (GetMessage(&msg, NULL, 0, 0))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return (int)msg.wParam;

}

LRESULT CALLBACK WndProc(HWND hwnd, UINT Message, WPARAM wparam, LPARAM lparam)

{

PAINTSTRUCT ps;

HDC hdc;

TCHAR greeting[] = \_T("Hello, Windows desktop!");

//Дескрипторы

static HWND btn1; static HWND btn2; static HWND btn3; static HWND btn4;// Кнопка

static HWND wind1, wind2, wind3, wind4; // Поля редактирования

static HWND hStat; // Статического текста

static HBRUSH hbrush;

char Buffer[BufferSize];

double a, b, sum, Len, eps1, eps2;

switch (Message)

{

case WM\_CREATE:

{

hInst = ((LPCREATESTRUCT)lparam)->hInstance; // Дескриптор приложения

wind1 = CreateWindow(szPole, \_T(""), WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_RIGHT, 75, 220, 60, 20, hwnd, (HMENU)DlgIndexNumber1, hInst, NULL); // создание окошек для ввода с параметрами

ShowWindow(wind1, SW\_SHOWNORMAL); // Вывод окошек

wind2 = CreateWindow(szPole, \_T(""), WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_RIGHT, 245, 220, 60, 20, hwnd, (HMENU)DlgIndexNumber2, hInst, NULL);

ShowWindow(wind2, SW\_SHOWNORMAL);

wind3 = CreateWindow(szPole, \_T(""), WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_RIGHT, 75, 280, 60, 20, hwnd, (HMENU)DlgIndexNumber3, hInst, NULL);

ShowWindow(wind3, SW\_SHOWNORMAL);

wind4 = CreateWindow(szPole, \_T(""), WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_RIGHT, 245, 280, 60, 20, hwnd, (HMENU)DlgIndexNumber4, hInst, NULL);

ShowWindow(wind4, SW\_SHOWNORMAL);

// создание четырех кнопок с параметрами

btn1 = CreateWindow(\_T("button"), \_T("t\*(t - 9)^3"), WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 145, 1, 100, 45, hwnd, 0, hInst, NULL);

ShowWindow(btn1, SW\_SHOWNORMAL);

btn2 = CreateWindow(\_T("button"), \_T("t + 1"), WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 145, 51, 100, 45, hwnd, 0, hInst, NULL);

ShowWindow(btn2, SW\_SHOWNORMAL);

btn3 = CreateWindow(\_T("button"), \_T("e^t + 1"), WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 145, 101, 100, 45, hwnd, 0, hInst, NULL);

ShowWindow(btn3, SW\_SHOWNORMAL);

btn4 = CreateWindow(\_T("button"), \_T("Ответ"), WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 145, 375, 90, 30, hwnd, 0, hInst, NULL);

ShowWindow(btn4, SW\_SHOWNORMAL);

// Создаем и показываем поле текста для результата

hStat = CreateWindow(\_T("static"), \_T("0"), WS\_CHILD | WS\_VISIBLE, 165, 420, 120, 20, hwnd, 0, hInst, NULL);

ShowWindow(hStat, SW\_SHOWNORMAL);

SetClassLongPtr(hwnd, GCLP\_HBRBACKGROUND, (LONG\_PTR)CreateSolidBrush(RGB(153, 51, 255))); //background (добавление цвета для главного окна)

}break;

case WM\_COMMAND: // Cообщение о команде

{

// Если нажали на кнопки (разные функции для вычисления + остальные кнопки)

if (lparam == (LPARAM)btn1)

{

switchcase = 1;

SetWindowText(hStatic, L"Будет вычислена функция - 1");

}

if (lparam == (LPARAM)btn2)

{

switchcase = 2;

SetWindowText(hStatic, L"Будет вычислена функция - 2");

}

if (lparam == (LPARAM)btn3)

{

switchcase = 3;

SetWindowText(hStatic, L"Будет вычислена функция - 3");

}

//Если нажали на кнопку "Ответ"

if (lparam == (LPARAM)btn4) {

GetDlgItemTextA(hwnd, DlgIndexNumber1, Buffer, BufferSize);

a = atof(Buffer); // Считывание данных из окна и перевод в нужный нам тип

GetDlgItemTextA(hwnd, DlgIndexNumber2, Buffer, BufferSize);

b = atof(Buffer);

GetDlgItemTextA(hwnd, DlgIndexNumber3, Buffer, BufferSize);

eps1 = atof(Buffer);

GetDlgItemTextA(hwnd, DlgIndexNumber4, Buffer, BufferSize);

eps2 = atof(Buffer);

// Изменение окна

SetWindowTextA(hStat, to\_string(newton(a, b, eps1, eps2)).c\_str());

}

}break;

case WM\_CTLCOLORSTATIC: // Цвета для статических объектов

{

HDC hdcStatic = (HDC)wparam;

SetTextColor(hdcStatic, RGB(0, 0, 0)); // Цвета для текста

SetBkColor(hdcStatic, RGB(153, 51, 255)); // Background

if (hbrush == NULL)

{

hbrush = CreateSolidBrush(RGB(153, 51, 255)); // Цвета для пустых элементов

}

return (INT\_PTR)hbrush;

}

case WM\_PAINT: // Перерисовка окна

{

hdc = BeginPaint(hwnd, &ps); // Начало перерисовки

SetTextColor(hdc, RGB(0, 0, 0)); // Цвета для текста TextOut

SetBkColor(hdc, RGB(153, 51, 255)); // Цвета для текста TextOut

TextOut(hdc, 75, 200, \_T("Точка 'A'"), 9);// Вывод текстовых сообщений

TextOut(hdc, 245, 200, \_T("Точка 'B'"), 9);

TextOut(hdc, 87, 260, \_T("Eps 1"), 5);

TextOut(hdc, 256, 260, \_T("Eps 2"), 5);

TextOut(hdc, 140, 420, \_T("X = "), 3);

hStatic = CreateWindow

(

L"static",

L"Необходимо выбрать функцию",

WS\_CHILD | WS\_VISIBLE,

75, 150, 230, 20,

hwnd, NULL, NULL, NULL

);

EndPaint(hwnd, &ps); // Конец перерисовки

}break;

case WM\_DESTROY: // Закрытие окна

{

PostQuitMessage(0);

}break;

default: // Обработка сообщения по умолчанию

return DefWindowProc(hwnd, Message, wparam, lparam);

}

return 0;

}

# **5 Тестовый набор**

На рисунке 1 представлено вычисление для функции :



Рисунок 1 – результат работы программы при

Вычислим интеграл вручную, чтобы убедиться в корректности работы программы.

Пусть . Вычислим значение выражения:

..

Программа работает корректно.

На рисунке 2 представлено вычисление для функции :



Рисунок 2 – результат работы программы при

Вычислим интеграл вручную, чтобы убедиться в корректности работы программы.

Пусть . Вычислим значение выражения:

x = 2.4641..

Программа работает корректно.

На рисунке 3 представлено вычисление для функции + 1



Рисунок 3 – результат работы программы при  *+ 1*

Вычислим интеграл вручную, чтобы убедиться в корректности работы программы.

Пусть . Вычислим значение выражения:

x = 1.6274..

Программа работает корректно.

Корни проверены на сайтах:

<https://www.integral-calculator.ru/>

<https://mathforyou.net/online/numerical/newton/>

https://mathdf.com/int/ru/