МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ

БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Факультет автоматики и вычислительной техники

Кафедра радиоэлектронных средств

Отчет по дисциплине «Цифровые устройства и МП»

Лабораторной работе №4

«Модульное программирование»

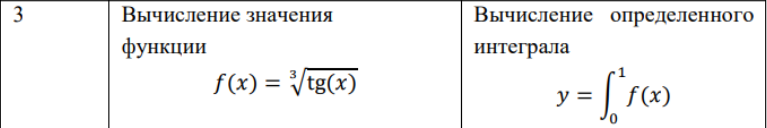
Выполнил: студенты гр. ИНБб – 3301                                              /И.А.Капустин/  
  
Проверил: педагог кафедры РЭС\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/М.А.Земцов/

Киров 2025

**Цель работы:**

Знакомство с технологией применения языка ассемблера при разработке программного обеспечения на языках высокого уровня.

**Задание:**

1. Сформировать проект консольного приложения с использованием модуля на основе языка ассемблера. 2. Разработать основной модуль приложения. Основной модуль обеспечивает ввод данных, вызов подпрограммы ассемблера и вывод результата. 3. Добавить в проект модуль на языке ассемблера. Разработать подпрограмму на языке ассемблера в соответствии с вариантом (табл. 3). 4. Добавить в проект модуль на языке Си. Разработать функцию на языке высокого уровня в соответствии с вариантом. 5. Обеспечить вызов подпрограммы ассемблера из основного модуля на языке С++. Основной модуль обеспечивает ввод данных и вывод результата. 6. Осуществить вызов функции языка Си из модуля ассемблера. 7. Проверить работу приложения в режиме отладки. Записать содержимое стека перед вызовом подпрограммы ассемблера и функции Си. Табл. 3.   
Варианты заданий  
  
  
****

**Текст программы с комментариями:**

Блок кода Sourse.cpp

#include <iostream>

#include <locale.h>

#include <math.h>

using namespace std;

// Объявление внешней функции на ассемблере

extern "C" float CalculateIntegral(int n); // n — количество шагов (разбиений)

int main() {

setlocale(LC\_ALL, "Rus"); // Для вывода русских символов в консоли

int n;

cout << "Введите количество шагов n (например, 1000): ";

cin >> n;

// Вызов функции интегрирования

float result = CalculateIntegral(n);

// Вывод результата

cout << "Интеграл от 0 до 1 равен: " << result << endl;

return 0;

}

extern "C" float fun\_el(float x) {

return cbrtf(tanf(x)); // cbrtf — кубический корень, tanf — тангенс

}

Блок кода AsmSource.asm

.586

.MODEL flat, C

.STACK 4096

.DATA

SUM DD 0.0 ; Общая сумма (хранится в ST(0))

STEP DD 0.0 ; Шаг интегрирования h = 1.0 / n

i DD 0 ; Переменная цикла

.CODE

extern fun\_el:near ; Объявляем внешнюю функцию на C

public CalculateIntegral

CalculateIntegral proc C

push ebp

mov ebp, esp ; Стандартное прологовое сохранение base pointer

finit ; Инициализация сопроцессора FPU

mov eax, [ebp + 8] ; Получаем параметр n из стека

mov ecx, eax ; Копируем n в ECX для цикла

mov i, 0 ; i = 0

; step = 1.0 / n

fild dword ptr [ebp + 8] ; Загружаем n как float (ST(0) = n)

fld1 ; ST(0) = 1.0, ST(1) = n

fdiv ; ST(0) = 1.0 / n

fstp STEP ; Сохраняем шаг в переменную step

fldz ; ST(0) = 0.0 (инициализация суммы)

loop\_start:

cmp i, ecx ; Если i >= n, выходим

jge loop\_end

; x = i \* step

fild i ; ST(0) = i

fld STEP ; ST(0) = step, ST(1) = i

fmul ; ST(0) = i \* step

fst ST(1) ; Дублируем x — один аргумент в стек, другой останется

sub esp, 4

fstp dword ptr [esp] ; Кладем x в стек для передачи в fun\_el

call fun\_el ; Вызов внешней C-функции: f(x)

add esp, 4 ; Очищаем стек

fadd ; ST(0) = sum + f(x)

inc i ; i++

jmp loop\_start

loop\_end:

fld STEP ; Загружаем шаг интегрирования

fmul ; Умножаем сумму на step => завершение метода прямоугольников

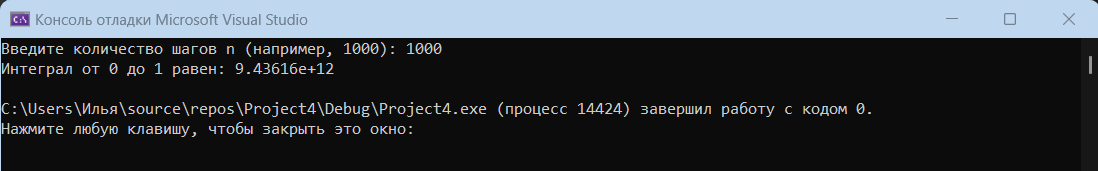
mov esp, ebp

pop ebp

ret ; Возврат результата в ST(0)

CalculateIntegral endp

END

**Верификация программы: результаты расчета заданного выражения, скриншоты, показывающие содержимое регистров и значения переменных после каждого действия программы.**  
****

**Вывод:**Я познакомился с технологией применения языка ассемблера при разработке программного обеспечения на языках высокого уровня.