МИНИИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Вятский государственный университет»**

**(ФГБОУ ВО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет по лабораторной работе 4

по дисциплине

«Технологии программирования»

Разработал студент группы ИВТ-23 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Арасланов А.А,/

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Долженкова М.Л./

Киров 2018

1. Постановка задачи

Используя язык Assembler реализовать функцию, проверяющую является ли число простым.

1. Стековый фрейм на момент вызова

|  |  |
| --- | --- |
| Содержимое регистра EBP | EBP |
| Адрес возврата | EBP+4 |
| Адрес переданного числа | EBP+8 |

1. ЛистингAssembler

.MODEL FLAT, C

.CODE

my\_assembler proc C n : dword;

mov ebx, eax

mov esi, 2

LOOP1:

cmp esi, ebx

JZ LOOP\_end0

xor eax, eax

LOOP2:

add eax,esi

cmp eax, ebx

jnae LOOP2

cmp eax, ebx

jz LOOP\_end1

inc esi

jmp LOOP1

LOOP\_end0:

mov eax, 2

ret

LOOP\_end1:

mov eax, 1

ret

my\_assembler endp

end

1. Листинг WINAPI

#include <windows.h>

#include <stdlib.h>

#include <objidl.h>

#include <string>

#include <sstream>

#include <iostream>

#include <iomanip>

#include <stdio.h>

#include <TCHAR.H>

#include <wchar.h>

#include <cstring>

#include <windowsx.h>

#include <windows.h>

using namespace std;

#define \_CRT\_SECURE\_NO\_WARNINGS

#define LABEL\_TEXT 20

#define MAX\_MATRIX\_SIZE 15

// Объявляем прототип CALLBACK функции

LRESULT CALLBACK WndProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

LRESULT CALLBACK WndProcc(HWND hWndinp, UINT uMsg, WPARAM wParam, LPARAM lParam);

LRESULT CALLBACK WndProcg(HWND hWndinp, UINT uMsg, WPARAM wParam, LPARAM lParam);

extern "C" int my\_assembler(int n);

HINSTANCE hInst;

HWND textbox, textbox1, text\_out\_nums, hWndinp, hWndres, rbutton,textbox8,textbox7, textbox9,textout10,textout11;

HWND button1, button2, button3, edit, edit1, edit2, edit3, textbox6, Edit5, button10, edit11, editout1, editout2;

HWND \*labels1 = new HWND[MAX\_MATRIX\_SIZE];

HWND arrayEdits1[15];

HWND arrayEdits2[15];

HWND arrayEdits3[15];

HWND hWindow;

HWND button4, button5, button6;

bool prov = false;

int flag = 0;

int flagop = 1;

int m1;

int siddle1;

bool siddle = false;

int n1;

int mas1[15];

double \*a;

int resmas[15];

string tmpstr = "";

string str;

string str1;

wchar\_t \*err = NULL;

int num;

const int n = 10, m = 10;

template<typename T>

T StringToNumber(const std::string& numberAsString)

{

T valor;

std::stringstream stream(numberAsString);

stream >> valor;

if (stream.fail()) {

std::runtime\_error e(numberAsString);

throw e;

}

return valor;

}

void copy()

{

for (int i = 0; i < m1; i++)

{

resmas[i] = 0;

resmas[i] = mas1[i];

}

}

// Функция WinAPI для точки входа в программу

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)

{

// Регистрация класса окна

// Объявляем переменную типа WNDCLASSEX

WNDCLASSEX wClass, wNewClass, elseClass;

ZeroMemory(&wClass, sizeof(wClass)); // Обнуляем память

ZeroMemory(&wNewClass, sizeof(wNewClass));

ZeroMemory(&elseClass, sizeof(wNewClass));

// Заполняем структуру WNDCLASSEX

wClass.cbSize = sizeof(wClass); // Размер равен размеру структуры

wClass.hbrBackground = (HBRUSH)COLOR\_WINDOW; // Определяем фон окна

wClass.hInstance = hInstance; // hInstance window

wClass.lpfnWndProc = (WNDPROC)WndProc; // Процедура обработки окна

wClass.lpszClassName = L"Class1"; // Имя класса

wNewClass.cbSize = sizeof(wNewClass); // Размер равен размеру структуры

wNewClass.hbrBackground = (HBRUSH)COLOR\_WINDOW; // Определяем фон окна

wNewClass.hInstance = hInstance; // hInstance window

wNewClass.lpfnWndProc = (WNDPROC)WndProcc; // Процедура обработки окна

wNewClass.lpszClassName = L"Class2"; // Имя класса

elseClass.cbSize = sizeof(elseClass); // Размер равен размеру структуры

elseClass.hbrBackground = (HBRUSH)COLOR\_WINDOW; // Определяем фон окна

elseClass.hInstance = hInstance; // hInstance window

elseClass.lpfnWndProc = (WNDPROC)WndProcg; // Процедура обработки окна

elseClass.lpszClassName = L"Class3"; // Имя класса

// Если произошла ошибка, то выводим сообщение

if (!RegisterClassEx(&wClass))

{

int nResult = GetLastError();

MessageBox(NULL, L"Класс окна не был создан!", L"Ошибка", MB\_ICONERROR);

}

if (!RegisterClassEx(&wNewClass))

{

int nResult = GetLastError();

MessageBox(NULL, L"Класс окна не был создан!", L"Ошибка", MB\_ICONERROR);

}

if (!RegisterClassEx(&elseClass))

{

int nResult = GetLastError();

MessageBox(NULL, L"Класс окна не был создан!", L"Ошибка", MB\_ICONERROR);

}

// Создаем окно при помощи функции WinApi CreateWindowEx

hWindow = CreateWindowEx(NULL,

L"Class1", // Имя класса, который мы определили ранее

L"Лабораторная №4", // Заголовок окна

WS\_TILED,

300, // x координата по горизонтали

300, // y координата по вертикали

500, // ширина создаваемого окна

200, // высота создаваемого окна

NULL,

NULL,

hInstance, // переменная экземпляра приложения

NULL);

// Если окно не было создано, то выдаем сообщение

if (!hWindow)

{

int nResult = GetLastError();

MessageBox(NULL, L"Окно не было создано!", L"Ошибка", MB\_ICONERROR);

}

hWndinp = CreateWindowEx(NULL,

L"Class2", // Имя класса, который мы определили ранее

L"Ввод числа", // Заголовок окна

WS\_TILED,

100, // x координата по горизонтали

20, // y координата по вертикали

620, // ширина создаваемого окна

330, // высота создаваемого окна

hWindow,

NULL,

hInstance, // переменная экземпляра приложения

NULL);

if (!hWndinp)

{

int nResult = GetLastError();

MessageBox(NULL, L"Окно не было создано!", L"Ошибка", MB\_ICONERROR);

}

hWndres = CreateWindowEx(NULL,

L"Class3", // Имя класса, который мы определили ранее

L"ASSEMBLER", // Заголовок окна

WS\_TILED,

100, // x координата по горизонтали

20, // y координата по вертикали

400, // ширина создаваемого окна

400, // высота создаваемого окна

hWindow,

NULL,

hInstance, // переменная экземпляра приложения

NULL);

if (!hWndres)

{

int nResult = GetLastError();

MessageBox(NULL, L"Окно не было создано!", L"Ошибка", MB\_ICONERROR);

}

// Показываем окно

ShowWindow(hWindow, SW\_SHOW);

UpdateWindow(hWindow);

// Объявляем переменную для сообщений типа MSG

MSG msg;

// Обнуляем память по размеру структуры MSG

ZeroMemory(&msg, sizeof(MSG));

// Цикл обработки сообщений

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

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return 0;

}

// определяем процедуру обратного вызова (WinApi)

LRESULT CALLBACK WndProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

switch (uMsg)

{

case WM\_CREATE:

{

textbox = CreateWindow(L"STATIC", L"Лабораторная работа №4", WS\_VISIBLE | WS\_CHILD | WS\_BORDER,

10, 30, 200, 20, hwnd, NULL, NULL, NULL);

textbox8 = CreateWindow(L"STATIC", L"Определение простоты числа", WS\_VISIBLE | WS\_CHILD | WS\_BORDER,

10, 70, 460, 20, hwnd, NULL, NULL, NULL);

HWND buttonpitch = CreateWindow(L"BUTTON", L"Ввести число", WS\_VISIBLE | WS\_CHILD

| WS\_BORDER, 30, 110, 120, 30, hwnd, (HMENU)1, NULL, NULL);

HWND button0 = CreateWindow(L"BUTTON", L"Закрыть", WS\_VISIBLE | WS\_CHILD

| WS\_BORDER, 350,110, 70, 30, hwnd, (HMENU)3, NULL, NULL);

SendMessage(button1, EM\_SETLIMITTEXT, 8, 0);

break;

}

case WM\_COMMAND:

{

switch (LOWORD(wParam))

{

// Обработка команд (нажатие кнопок, мыши, полей ввода и т.д.)

case 1:

{

ShowWindow(hWndinp, SW\_SHOW);

EnableWindow(hWindow, FALSE);

break;

}

case 3:

{

PostQuitMessage(0);

return 0;

break;

}

}

break;

}

case WM\_DESTROY:

{

PostQuitMessage(0);

return 0;

break;

}

}

return DefWindowProc(hwnd, uMsg, wParam, lParam);

}

LRESULT CALLBACK WndProcc(HWND hWndinp, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

int tmpnum = 0;

LPWSTR tmp = L"0";

char \*err = NULL;

unsigned int TextLength;

char \*Text = NULL;

switch (uMsg)

{

case WM\_CREATE:

{

HWND button0 = CreateWindow(L"BUTTON", L"Закрыть", WS\_VISIBLE | WS\_CHILD | WS\_BORDER,530,250, 70, 30, hWndinp, (HMENU)1, NULL, NULL);

SendMessage(button1, EM\_SETLIMITTEXT, 8, 0);

{

textbox1 = CreateWindow(L"STATIC", L" Число:", WS\_VISIBLE | WS\_CHILD,

20, 50, 200, 20, hWndinp, NULL, NULL, NULL);

edit = CreateWindow(L"EDIT", L"", WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_NUMBER,

180, 50, 30, 20, hWndinp, NULL, NULL, NULL);

Edit\_LimitText(edit, 3);

button2 = CreateWindow(L"BUTTON", L"ОК", WS\_VISIBLE | WS\_CHILD | WS\_BORDER, 220, 50, 80, 25, hWndinp, (HMENU)5, NULL, NULL);

flag = 0;

text\_out\_nums = CreateWindow(L"STATIC", L"Введите число", WS\_VISIBLE | WS\_CHILD | WS\_BORDER,

10, 120, 420, 20, hWndinp, NULL, NULL, NULL);

break;

}

break;

}

case WM\_COMMAND:

{

switch (LOWORD(wParam))

{

case 1:

{

SetForegroundWindow(hWindow); ShowWindow(hWndinp, SW\_HIDE);

EnableWindow(hWindow, TRUE);

SetFocus(hWindow);

break;

}

case 5:

{

// TextLength = GetWindowTextLength(edit) + 1;

Text = new char[TextLength];

GetWindowTextA(edit, Text, TextLength + 1);

m1 = atoi(Text);

delete(Text);

if (m1 < 2)

{

MessageBox(hWndinp, L"Введите число >=2", L"Ошибка", MB\_OK);

}

else{

num = my\_assembler(m1);

if (num == 2) {

SetWindowText(text\_out\_nums, L"Число является простым");

}

else {

SetWindowText(text\_out\_nums, L"Число не является простым");

}

}

break;

} case 8:

{

break;

}

}

break;

}

case WM\_DESTROY:

{

PostQuitMessage(0);

return 0;

break;

}

}

return DefWindowProc(hWndinp, uMsg, wParam, lParam);

}

LRESULT CALLBACK WndProcg(HWND hWndres, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

HDC hdc;

PAINTSTRUCT ps;

switch (uMsg)

{

case WM\_CREATE:

{

HWND button6 = CreateWindow(L"BUTTON", L"Закрыть", WS\_VISIBLE | WS\_CHILD | WS\_BORDER, 660, 310, 90, 30, hWndres, (HMENU)1, NULL, NULL);

wchar\_t str[65];

string buf;

if (flagop == 1)

{

copy();

for (int i = 0; i < m1; i++)

{

DestroyWindow(arrayEdits3[i]);

buf = to\_string(resmas[i]);

arrayEdits3[i] = CreateWindow(L"EDIT", L"", WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_READONLY,

i \* 40 + 10, 40 + i \* 25 + 25, 40, 25, hWndres, NULL, NULL, NULL);

}

}

if ((flagop != 2) && (flagop != 1))

{

MessageBox(hWndres, L"Матрица не была введена!", L"Ошибка", MB\_OK);

}

break;

}

case WM\_COMMAND:

{

switch (LOWORD(wParam))

{

// Обработка команд (нажатие кнопок, мыши, полей ввода и т.д.)

case 1:

{

SetForegroundWindow(hWindow);

ShowWindow(hWndres, SW\_HIDE);

EnableWindow(hWindow, TRUE); SetFocus(hWindow);

break;

} break;

}

break;

case WM\_DESTROY:

{

PostQuitMessage(0);

return 0;

}

default:

return DefWindowProc(hWndres, uMsg, wParam, lParam);

}

}

}

1. Выводы

В ходе выполнения лабораторной работы разработано приложение Windows, выполняющее работу заданных функции. Решение задачи реализовано через модуль, написанный на языке Assembler–my\_assemler.asm.