**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Операционные системы»

**Лабораторная работа № 5**

Тема: Управление процессами в ОС

Студент: Калугин К. А.

Группа: М8О-207Б-19

Преподаватель: Миронов Е.С.

Дата:

Оценка:

Москва, 2021

**Постановка задачи:** Требуется создать динамические библиотеки, которые реализуют определенный функционал. Далее использовать данные библиотеки 2-мя способами: 1. Во время компиляции (на этапе «линковки»/linking) 2. Во время исполнения программы. Библиотеки загружаются в память с помощью интерфейса ОС для работы с динамическими библиотеками В конечном итоге, в лабораторной работе необходимо получить следующие части:

• Динамические библиотеки, реализующие контракты, которые заданы вариантом;

• Тестовая программа (программа №1), которая используют одну из библиотек, используя знания полученные на этапе компиляции;

• Тестовая программа (программа №2), которая загружает библиотеки, используя только их местоположение и контракты. Провести анализ двух типов использования библиотек.

5) Расчет значения числа Пи при заданной длине ряда (K) float Pi(int K) [Ряд Лейбница/ Формула Валлиса ]

6) Расчет значения числа е(основание натурального логарифма) Float E(int x) [(1 + 1/x) ^ x/ Сумма ряда по n от 0 до x, где элементы ряда равны: (1/(n!))]

**Алгоритм решения задачи.**

Вначале я написал функции, реализующие методы подсчета числа Пи и Е. После этого, я получил dll библиотеки с этими функциями. Эти библиотеки я использовал в двух программах – одна подключала их при компиляции, другая – динамически.

**Листинг программы**

**1.c**

#include <stdio.h>

\_\_declspec (dllimport) float Pi (int K);

\_\_declspec (dllimport) float E (int x);

int main () {

    int var;

    scanf ("%d", &var);

    if (var == 1) {

        int K;

        scanf ("%d", &K);

        float ans = Pi (K);

        printf ("%f", ans);

    }

    else {

        int x;

        scanf ("%d", &x);

        float ans = E (x);

        printf ("%f", ans);

    }

    return 0;

}

//gcc -c -o \_.o .\\_.c;gcc -o \_.dll -shared .\\_.o

//gcc -c -o 1.o .\1.c;gcc -o 1.exe .\1.o -L. -lLeibniz -lX

//.\1.exe

**2.c**

#include <stdio.h>

#include <windows.h>

typedef float (\*ItoF) (int A);

int main () {

    int flag = 1;

    HINSTANCE libPi;

    ItoF pi;

    libPi = LoadLibrary ("Leibniz.dll");

    pi = (ItoF) GetProcAddress (libPi, "Pi");

    HINSTANCE libE;

    ItoF e;

    libE = LoadLibrary ("X.dll");

    e = (ItoF) GetProcAddress (libE, "E");

    while (1) {

        int in;

        scanf ("%d", &in);

        if (in == 0) {

            if (flag == 1) {

                FreeLibrary (libPi);

                FreeLibrary (libE);

                libPi = LoadLibrary ("Leibniz.dll");

                pi = (ItoF) GetProcAddress (libPi, "Pi");

                libE = LoadLibrary ("X.dll");

                e = (ItoF) GetProcAddress (libE, "E");

                flag = 2;

            }

            else {

                FreeLibrary (libPi);

                FreeLibrary (libE);

                libPi = LoadLibrary ("Vallis.dll");

                pi = (ItoF) GetProcAddress (libPi, "Pi");

                libE = LoadLibrary ("SeriesSum.dll");

                e = (ItoF) GetProcAddress (libE, "E");

                flag = 1;

            }

        }

        else if (in == 1) {

            int K;

            scanf ("%d", &K);

            float ans = (pi) (K);

            printf ("%f\n", ans);

        }

        else if (in == 2) {

            int x;

            scanf ("%d", &x);

            float ans = (e) (x);

            printf ("%f\n", ans);

        }

        else {

            break;

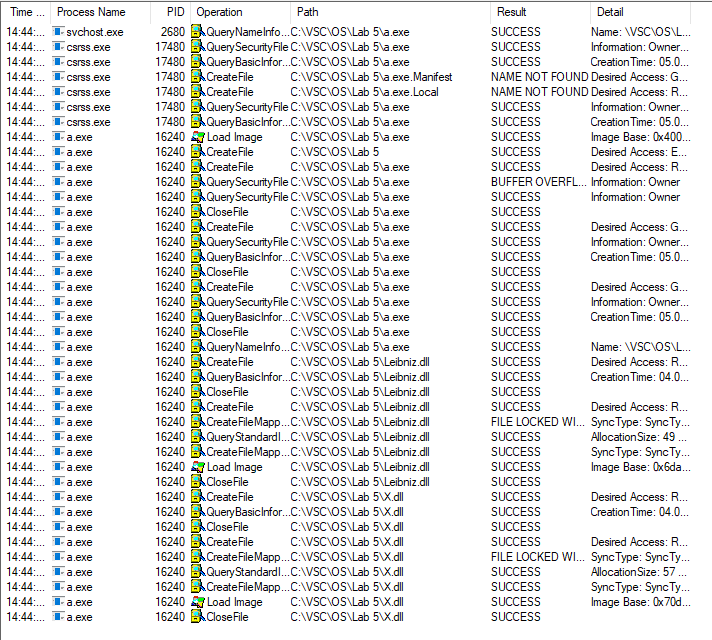
        }

    }

    //printf ("!!!");

    return 0;

}

**ProcMon**

**Выводы**

Я изучил принципы создания dll библиотек, позволяющих быстро и удобно подключать необходимые функции. Также я изучил способы подключения этих библиотек.

**Список литературы**

1. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. —СПб.: Издательский дом «Питер», 2018. — С. 111 - 123