**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа № 5**

**По курсу:  
«Операционные системы»**

Студент: Патрикеева Лидия

Группа: 80-207Б-18

Преподаватель: Миронов Е.С.

Дата:

Оценка:

Москва, 2019

**Цель работы**

Приобретение практических навыков в:

* Создание динамических библиотек
* Создание программ, которые используют функции динамических библиотек

**Задание**

Требуется создать динамическую библиотеку, которая реализует определенный функционал. Далее использовать данную библиотеку 2-мя способами:

1. Во время компиляции (на этапе «линковки»/linking)
2. Во время исполнения программы, подгрузив библиотеку в память с помощью системных вызовов

В конечном итоге, программа должна состоять из следующих частей:

* Динамическая библиотека, реализующая заданных вариантом интерфейс;
* Тестовая программа, которая используют библиотеку, используя знания, полученные на этапе компиляции;
* Тестовая программа, которая использует библиотеку, используя только местоположение динамической библиотеки и ее интерфейс.

Провести анализ между обоими типами использования библиотеки.

Вариант 22:

Структура данных, с которой должна обеспечивать работу библиотека:

1. Работа со списком

Тип данных, используемый структурой: 4. Md5 суммы

**Исходный код**

dinamic.c

#include <stdlib.h>

#include <stdio.h>

#include <windows.h>

struct Node

{

unsigned long long value;

struct Node \*next;

};

typedef void(\_\_stdcall \*F\_ADD)(struct Node \*\*,unsigned long long);

typedef int (\_\_stdcall \*F\_SEARCH)(struct Node\*\* , unsigned long long );

typedef int (\_\_stdcall \* F\_DELETE)(struct Node\*\* , unsigned long long );

typedef void (\_\_stdcall \* F\_PRINT)(struct Node\*\* );

int main(){

HMODULE lib = LoadLibrary(TEXT("list.dll"));

if (lib == NULL)

{

printf("Load error : %d\n", GetLastError());

return 0;

}

F\_ADD addFun;

F\_SEARCH searchFun;

F\_DELETE deleteFun;

F\_PRINT printFun;

addFun = (F\_ADD)GetProcAddress(lib, TEXT("Add"));

searchFun = (F\_SEARCH)GetProcAddress(lib, TEXT("Search"));

deleteFun = (F\_DELETE)GetProcAddress(lib, TEXT("DeleteEl"));

printFun = (F\_PRINT)GetProcAddress(lib, TEXT("Print"));

if( addFun == NULL || searchFun == NULL || deleteFun == NULL || printFun == NULL){

printf("Error: Cannot find function %d\n", GetLastError());

}

struct Node \*head = NULL;

int command = -1;

unsigned long long md5sum ;

printf("1-add 2-delete 3-find 4-print 5-exit\n");

while (1)

{

scanf("%d", &command);

switch (command)

{

case 1:

printf("Enter md5sum to add: ");

scanf("%llx", &md5sum);

addFun(&head, md5sum);

break;

case 2:

printf("Enter md5sum to delete: ");

scanf("%llx", &md5sum);

if (deleteFun(&head, md5sum))

{

printf("Deleted\n");

}

else

{

printf("Not in list\n");

}

break;

case 3:

printf("Enter md5sum to find: ");

scanf("%llx", &md5sum);

if (searchFun(&head, md5sum))

{

printf("In list\n");

}

else

{

printf("Not in list\n");

}

break;

case 4:

printFun(&head);

break;

case 5:

return 1;

default:

printf("No such command\n");

break;

}

printf("1-add 2-delete 3-find 4-print 5-exit\n");

}

FreeLibrary(lib);

return 0;

}

**Вывод**

В процессе выполнения данной лабораторной работы я научилась создавать dll библиотеки. Однако для каждого компилятора команды для создания такой библиотеки отличаются, что не очень удобно. Та же ситуация с командами подключения этих библиотек к программам, которые интуитивно никак не могут быть понятны. Динамические библиотеки удобны тем, что разные программы могут совместно использовать одну копию такой библиотеки, это помогает сэкономить используемое пространство.