Московский авиационный институт (национальный исследовательский университет)

# Факультет информационных технологий и прикладной математики

**Кафедра вычислительной математики и программирования**

## Лабораторная работа №5 по курсу «Операционные системы»

**Создание и применение динамических библиотек в операционной системе**

## «UNIX».

Студент: Селивёрстов Д. С. Преподаватель: Миронов Е. С.

Группа: М8О-201Б-21

Вариант 9

Дата: Оценка: Подпись:

**Москва, 2023**

# Условие

Требуется создать динамические библиотеки, которые реализуют определенный функ- ционал. Далее использовать данные библиотеки 2-мя способами: Во время компиляции (на этапе «линковки»/linking) Во время исполнения программы. Библиотеки загружа- ются в память с помощью интерфейса ОС для работы с динамическими библиотеками В конечном итоге, в лабораторной работе необходимо получить следующие части: Ди- намические 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 загру- жает библиотеки, используя только их местоположение и контракты. Провести анализ двух типов использования библиотек.

Пользовательский ввод для обоих программ должен быть организован следую- щим образом: Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2). Можно ре- ализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»; «1 arg1 arg2 . . . argN», где после «1» идут аргументы для пер- вой функции, предусмотренной контрактами. После ввода команды происходит вызов первой функции, и на экране появляется результат её выполнения; «2 arg1 arg2 . . . argM», где после «2» идут аргументы для второй функции, предусмотренной контрак- тами. После ввода команды происходит вызов второй функции, и на экране появляется результат её выполнения.

# Задание

Рассчет интеграла функции *sin*(*x*) на отрезке [*A, B*] с шагом *e*: (*FloatSinIntegral*(*float A*, *float B*, *float e*)). Методы: Подсчет интеграла методом прямоугольников, подсчет интеграла методом трапеций.

Рассчет производной функции *floatf′*(*x*) = (*f* (*A* + *deltaX*)*f′*(*x*) = (*f* (*A* + *deltaX*)

# Метод решения

Использовать утилиты компилятора *gcc* для динамической "линковки"библиотек.

# Код программы

## libLab5.h

**typedef double** ( ∗ TDerivative )( **double** , **double** ) ;

**typedef double** ( ∗ TIntegral )( **double** , **double** , **double** ) ;

**double** D e r i v a t i v e ( **double** a , **double** delta X ) ;

**double** In t e g r a t e ( **double** a , **double** b , **double** e p s i l o n ) ;

## lib1.c

**#include** <math . h>

**double** D e r i v a t i v e ( **double** a , **double** delta X ){

**return** ( cos ( a + delta X ) − cos ( a )) / delta X ;

}

**double** In t e g r a t e ( **double** a , **double** b , **double** e p s i l o n )

{

**int** s t e p s = f a b s ( b − a ) / e p s i l o n ;

**double** point = a ;

**double** r e s u l t = 0 ;

**for** ( **int** i = 0 ; i < s t e p s ; ++i )

{

r e s u l t += s i n ( point ) ∗ e p s i l o n ; point += e p s i l o n ;

}

**return** r e s u l t ;

}

## lib2.c

**#include** <s t d i o . h>

**#include** <math . h>

**double** In t e g r a t e ( **double** a , **double** b , **double** e p s i l o n )

{

**int** s t e p s = f a b s ( b − a ) / e p s i l o n ;

**double** point = a ;

**double** r e s u l t = 0 ;

**for** ( **int** i = 0 ; i < s t e p s ; ++i )

{

r e s u l t += s i n ( point + e p s i l o n / 2 ) ∗ e p s i l o n ; point += e p s i l o n ;

}

**return** r e s u l t ;

}

**double** D e r i v a t i v e ( **double** a , **double** delta X ){

**return** ( cos ( a + delta X ) − cos ( a − delta X )) / ( 2 ∗ delta X ) ;

}

## program1.c

**#include** <s t d i o . h>

**#include** <lib Lab 5 . h>

**int** main ()

{

p r i n t f ( "1 ␣−␣ i n t e g r a t e , ␣2 ␣−␣ f i n d ␣ d e r i v a t i v e , ␣−1␣−␣ q u i t \n" ) ;

**int** command ;

**while** ( s c a n f ( "%d" , &command ) != EOF)

{

**i f** ( command == 1 )

{

p r i n t f ( " Pass ␣arguments : ␣\" a␣b␣ e p s i l o n \"\ n" ) ;

**double** a , b , e p s i l o n ;

s c a n f ( "%l f ␣%l f ␣%l f " , &a , &b , &e p s i l o n ) ;

p r i n t f ( "%l f \n" , In t e g r a t e ( a , b , e p s i l o n ) ) ;

}

**else i f** ( command == 2 )

{

p r i n t f ( " Pass ␣arguments : ␣\" a␣delta X \"\ n" ) ;

**double** a , delta X ;

s c a n f ( "%l f ␣%l f " , &a , &delta X ) ;

p r i n t f ( "%l f \n" , D e r i v a t i v e ( a , delta X ) ) ;

}

**else i f** ( command == −1){

## break ;

}

## else {

**return** 0 ;

}

}

**return** 0 ;

}

## program2.c

**#include** <s t d i o . h> **#include** <s t d l i b . h> **#include** <d l f c n . h> **#include** " l ib Lab 5 . h"

**int** main ()

{

**char** ∗ libNames [ ] = {" . / l ibdynamic 1 . so " , " . / l ibdynamic 2 . so " };

**int** cur Lib = 0 ;

**int** numOfLibs = 2 ;

**void** ∗ l i b C t r l ;

l i b C t r l = dlopen ( libNames [ cur Lib ] , RTLD\_NOW) ;

**i f** ( ! l i b C t r l )

{

p e r r o r ( " Openning␣dynamic␣ l i b ␣ e r r o r ␣ ) " ) ; e x i t (EXIT\_FAILURE) ;

}

TDerivative d e r i v a t i v e ; TIntegral i n t e g r a l ;

#pragma GCC d i a g n o s t i c push

#pragma GCC d i a g n o s t i c i gnored "−Wpedantic"

d e r i v a t i v e = ( TDerivative ) dlsym ( l i b C t r l , " D e r i v a t i v e " ) ; i n t e g r a l = ( TIntegral ) dlsym ( l i b C t r l , " In t e g r a t e " ) ;

#pragma GCC d i a g n o s t i c pop

p r i n t f ( "0 ␣−␣ switch ␣ l i b r a r y ␣ ( d e f a u l t ␣−␣ d e r i v a t i v e ) , ␣1 ␣−␣ f i n d ␣ d e r i v a t i v e , ␣2

**int** command ;

**while** ( s c a n f ( "%d" , &command ) != EOF)

{

**i f** ( command == 0 )

{

cur Lib = ( cur Lib + 1 ) % numOfLibs ;

**i f** ( d l c l o s e ( l i b C t r l ) != 0 )

{

p e r r o r ( " c l o s i n g ␣dynamic␣ l i b ␣ e r r o r " ) ; e x i t (EXIT\_FAILURE) ;

}

**i f** ( ! ( l i b C t r l = dlopen ( libNames [ cur Lib ] , RTLD\_LAZY) ) )

{

p e r r o r ( " c l o s i n g ␣dynamic␣ l i b ␣ e r r o r " ) ; e x i t (EXIT\_FAILURE) ;

}

#pragma GCC d i a g n o s t i c push

#pragma GCC d i a g n o s t i c i gnored "−Wpedantic"

d e r i v a t i v e = ( TDerivative ) dlsym ( l i b C t r l , " D e r i v a t i v e " ) ; i n t e g r a l = ( TIntegral ) dlsym ( l i b C t r l , " In t e g r a t e " ) ;

#pragma GCC d i a g n o s t i c pop

}

**else i f** ( command == 1 )

{

p r i n t f ( " Pass ␣arguments : ␣\" a␣delta X \"\ n" ) ;

**double** a , delta X ;

s c a n f ( "%l f ␣%l f " , &a , &delta X ) ;

**double** r e s u l t = ( d e r i v a t i v e )( a , delta X ) ; p r i n t f ( "%l f \n" , r e s u l t ) ;

}

**else i f** ( command == 2 )

{

p r i n t f ( " Pass ␣arguments : ␣\" a␣b␣ e p s i l o n \"\ n" ) ;

**double** a , b , e p s i l o n ;

s c a n f ( "%l f ␣%l f ␣%l f " , &a , &b , &e p s i l o n ) ;

**double** r e s u l t = ( i n t e g r a l )( a , b , e p s i l o n ) ; p r i n t f ( "%l f \n" , r e s u l t ) ;

}

**else i f** ( command == −1){

## break ;

}

## else {

**return** 0 ;

}

}

**return** 0 ;

}

# Выводы

В процессе выполнения лабораторной работы были приобретены навыки практического применения создания, обработки и отслеживания состояния динамических библиотек.