Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №5 по курсу**

**«Операционные системы»**

Студент: Копылов Михаил Юрьевич

Группа: М8О-201Б-21

Вариант: 27

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2022

**Содержание**

1. Репозиторий
2. Постановка задачи
3. Общие сведения о программе
4. Исходный код
5. Выводы

**Репозиторий**

https://github.com/Mikhail-cWc/OS\_mai/tree/main/lab5

**Постановка задачи**

**Цель работы**

Целью является приобретение практических навыков в:

* Создание динамических библиотек
* Создание программ, которые используют функции динамических библиотек
* Работа со сборочной системой

**Задание**

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.

**Общие сведения о программе**

Используются системные вызовы Windows. Динамические библиотеки формата DLL и LIB

**Исходный код**

|  |
| --- |
| **Main1.cpp** |
| #include "utils.h"  int main()  {  const char \*fileWithOutput2 = "output2.txt";  programm1(std::cin, fileWithOutput2);  return 0;  } |

|  |
| --- |
| **Main2.cpp** |
| #include "utils.h"  int main()  {  const char \*fileWithOutput2 = "output2.txt";  programm1(std::cin, fileWithOutput2);  return 0;  } |

|  |
| --- |
| **Programm1.cpp** |
| #include "utils.h"  #include "MathUnit.h"  \_\_declspec(dllimport) float \_\_cdecl Pi\_V1(int);  \_\_declspec(dllimport) float \_\_cdecl E\_V1(int); // a function from a DLL  int argument(std::string s)  {  std::string res = "";  for (int i = 2; i < s.size(); i++)  res += s[i];  return std::stoi(res);  }  int programm1(std::istream &inFile, const char \*fileWithOutput1)  {  std::ofstream outFile(fileWithOutput1);  std::string command;  outFile << std::fixed;  outFile.precision(15);  while (std::getline(inFile, command))  {  if (command[0] == '1')  outFile << Pi\_V1(argument(command)) \* 4 << std::endl;  if (command[0] == '2')  outFile << E\_V1(argument(command)) << std::endl;  if (command[0] != '1' && command[0] != '2')  outFile << "invalid\_command\n";  }  outFile.close();  return 0;  } |

|  |
| --- |
| **Programm2.cpp** |
| #include "utils.h"  int programm2(std::istream &inFile, const char \*fileWithOutput2)  {  std::ofstream outFile(fileWithOutput2);  HINSTANCE hinstLib;  MYPROC Pi, E;  bool flag = FALSE;  std::string command;  outFile << std::fixed;  outFile.precision(15);  hinstLib = LoadLibrary(TEXT("libmath.dll"));  // If the handle is valid, try to get the function address.  if (hinstLib != NULL)  {  Pi = (MYPROC)GetProcAddress(hinstLib, "Pi\_V1");  E = (MYPROC)GetProcAddress(hinstLib, "E\_V1");  while (std::getline(inFile, command))  {  if (command[0] == '0')  {  outFile << "Swithing\_the\_implementation\_of\_mathematical\_function\n";  if (!flag)  {  Pi = (MYPROC)GetProcAddress(hinstLib, "Pi\_V2");  E = (MYPROC)GetProcAddress(hinstLib, "E\_V2");  flag = TRUE;  }  else  {  Pi = (MYPROC)GetProcAddress(hinstLib, "Pi\_V1");  E = (MYPROC)GetProcAddress(hinstLib, "E\_V1");  flag = FALSE;  }  }  if (command[0] == '1')  {  if (NULL != Pi && flag)  outFile << Pi(argument(command)) << std::endl;  else if (NULL != Pi)  outFile << Pi(argument(command)) \* 4 << std::endl;  else  std::cerr << "Error load proc Pi\n";  }  if (command[0] == '2')  {  if (NULL != E)  outFile << E(argument(command)) << std::endl;  else  std::cerr << "Error load proc E\n";  }  if (command[0] != '0' && command[0] != '1' && command[0] != '2')  outFile << "invalid\_command\n";  }  // Free the DLL module.  if (!FreeLibrary(hinstLib))  std::cerr << "Error FleeLibrary\n";  }  else  std::cerr << "Error load library DLL\n";  return 0;  } |

|  |
| --- |
| **MathLibrary.cpp** |
| // The myPuts function writes a null-terminated string to  // the standard output device.  // The export mechanism used here is the \_\_declspec(export)  // method supported by Microsoft Visual Studio, but any  // other export method supported by your development  // environment may be substituted.  #include "MathUnit.h"  float \_\_cdecl Pi\_V1(int K)  {  float res = 0;  for (int i = 0; i <= K - 1; i++)  {  res += pow(-1, i) / (2 \* i + 1);  }  return res;  }  float \_\_cdecl Pi\_V2(int K)  {  float res = 1;  for (int i = 1; i <= K - 1; i++)  {  res \*= (4. \* i \* i) / ((2. \* i - 1.) \* (2. \* i + 1.));  }  res \*= ((2. \* K) / (2. \* K - 1.)) \* (((2. \* K) / (8. \* K + 4.)) + 1.) + 3. / 4.;  return res;  }  float \_\_cdecl E\_V1(int x)  {  float res = 1;  float fact = 1;  for (int i = 1; i <= x; i++)  {  res += fact \* (1.0 / i);  fact = fact \* (1.0 / i);  }  return res;  }  float \_\_cdecl E\_V2(int x)  {  float res = 1;  float fact = 1;  for (int i = 1; i <= x; i++)  {  res += fact \* (1.0 / i);  fact = fact \* (1.0 / i);  }  return res;  } |

**Выводы**

Составлены и отлажены программы на языке С++, осуществляющие работу с динамическими библиотеками. Одна подключает библиотеки на этапе линковки, другая во время работы по средствам ОС.