

**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: Д. А. Алгиничев
Группа: М8О-208БВ-24
Преподаватель: Е. С. Миронов

Москва, 2025

Условие

Составить и отладить программу, осуществляющую работу с динамическими библиотеками и их загрузку во время выполнения. Программа должна поддерживать возможность смены реализаций алгоритмов без перекомпиляции.

Цель работы

Приобретение навыков в создании динамических библиотек и программ, которые используют функции динамических библиотек.

Задание

Требуется создать динамические библиотеки, которые реализуют заданных вариантом функционал. Далее использовать данные библиотеки 2-мя способами: 1. Во время компиляции (на этапе "линковки"). 2. Во время выполнения программы. Библиотеки загружаются в память с помощью интерфейса ОС для работы с динамическими библиотеками.

Задание варианта

Реализовать программу, которая может загружать различные реализации алгоритмов вычисления производной и сортировки массивов через динамические библиотеки. Программа должна предоставлять интерфейс для смены библиотек во время выполнения.

Вариант

15

Архитектура программы

Общая структура

Программа реализует архитектуру плагинов с использованием динамической загрузки библиотек. Основные компоненты:

- **DynamicLoader** - загрузчик динамических библиотек
- **AbsDerivative** - абстрактный интерфейс для вычисления производных
- **AbsSort** - абстрактный интерфейс для алгоритмов сортировки
- **Конкретные реализации** - различные варианты алгоритмов в отдельных библиотеках

Ключевые алгоритмы

Динамическая загрузка библиотек

Для загрузки библиотек используются системные вызовы:

- `dlopen` - открытие библиотеки
- `dlsym` - получение указателей на функции
- `dlclose` - закрытие библиотеки

Результаты тестирования

Пример работы программы

Ведите, что хотите сделать

-1 Конец программы

0 Смена библиотеки

1 Производная

2 Сортировка

0

Ведите название библиотеки

`libcontract1.so`

Ведите, что хотите сделать

1 Производная

Ведите точку

1.57

Ведите отклонение

0.01

-0.99995

Ведите, что хотите сделать

2 Сортировка

Ведите размер массива

5

Ведите элементы массива

5 3 1 4 2

1 2 3 4 5

Сравнение алгоритмов

Сортировка пузырьком vs Быстрая сортировка:

- **BubbleSort:** Время $O(n^2)$
- **QuickSort:** Время $O(n \log n)$ (В СРЕДНЕМ!)

Производная методом 1 vs методом 2:

- **Derivative1:** $(f(x+x) - f(x)) / x$ - точность $O(x)$
- **Derivative2:** $(f(x+x) - f(x-x)) / (2x)$ - точность $O(x^2)$

Выводы

В процессе выполнения лабораторной работы я получил практический опыт работы с динамическими библиотеками и двумя типами их использования.

Статическое связывание - библиотеки линкуются на этапе компиляции. Преимущества: высокая производительность, простота развертывания. Недостатки: большой размер исполняемого файла, невозможность обновления без перекомпиляции.

Динамическая загрузка - библиотеки загружаются во время выполнения. Преимущества: гибкость, возможность обновления, разделение кода. Недостатки: дополнительная сложность, overhead на загрузку.

Приобретенные навыки

- **Работа с dlopen/dlsym** - освоил механизм динамической загрузки в Linux
- **Архитектура плагинов** - научился проектировать системы с подключаемыми модулями
- **Создание shared libraries** - изучил процесс сборки .so файлов

Заключение

Динамическая загрузка библиотек открывает большие возможности для создания гибких и расширяемых приложений. Несмотря на дополнительную сложность, этот подход незаменим в случаях, когда требуется:

- Поддержка плагинов и расширений
- Обновление компонентов без перекомпиляции

Полученные знания позволяют создавать более архитектурно сложные и поддерживаемые приложения.

Исходная программа

```
1 #include <iostream>
2 #include "../include/AbsDerivative.hpp"
3 #include "../include/AbsSort.hpp"
4
5 extern "C"
6 {
7     AbsDerivative * CreateDerivative();
8     AbsSort * CreateSort();
9     void DeleteDerivative(AbsDerivative*);
10    void DeleteSort(AbsSort*);
11 }
12
13 int main(int argc, char * argv[])
14 {
15     AbsDerivative * der = CreateDerivative();
```

```
16 AbsSort * sort = CreateSort();
17
18     while (true)
19     {
20         int cmd = 0;
21         std::cout << "Введите, что хотите сделать" << std::endl;
22         std::cout << "0 Конецпрограммы" << std::endl;
23         std::cout << "1 Производная" << std::endl;
24         std::cout << "2 Сортировка" << std::endl;
25         std::cin >> cmd;
26
27         if (cmd == 1)
28         {
29             float a;
30             float dx;
31             std::cout << "Введите точку" << std::endl;
32             std::cin >> a;
33             std::cout << "Введите отклонение" << std::endl;
34             std::cin >> dx;
35             std::cout << der->Derivative(a, dx) << std::endl;
36         }
37
38         else if (cmd == 2)
39         {
40             int n;
41             std::cout << "Введите размер массива" << std::endl;
42             std::cin >> n;
43             int * array = new int[n];
44
45             size_t i = 0;
46             std::cout << "Введите элементы массива" << std::endl;
47             for (i = 0; i < n; ++i) std::cin >> array[i];
48
49             sort->Sort(array, n);
50
51             for (i = 0; i < n; ++i) std::cout << array[i] << " ";
52             std::cout << std::endl;
53
54             delete[] array;
55         }
56
57         else if (cmd == 0) break;
58     }
59     DeleteDerivative(der);
60     DeleteSort(sort);
61
62     return 0;
63 }
```

Листинг 1: main.cpp (program1) - пример статического использования библиотеки с контрактом 1

```
1 #include <iostream>
2 #include "DynamicLoader.hpp"
3
4 int main() {
5     DynamicLoader loader;
6     AbsDerivative* der = nullptr;
7     AbsSort* sorter = nullptr;
8
9     if (!loader.Load("./libcontract1.so")) {
10         std::cout << "Initial load error: " << loader.Error() << std::endl;
11         return 1;
12     }
13
14     der = loader.CreateDerivative();
15     sorter = loader.CreateSort();
16
17     // Проверка создания объектов
18     if (!der || !sorter) {
19         std::cout << "Failed to create objects from initial library" << std::endl;
20         if (!der) std::cout << "Derivative object is null" << std::endl;
21         if (!sorter) std::cout << "Sort object is null" << std::endl;
22         return 1;
23     }
24
25     while (true) {
26         std::cout << "Введите, что хотите сделать" << std::endl;
27         std::cout << "-1 Конец программы" << std::endl;
28         std::cout << "0 Смена библиотеки" << std::endl;
29         std::cout << "1 Производная" << std::endl;
30         std::cout << "2 Сортировка" << std::endl;
31         int cmd;
32         std::cin >> cmd;
33
34         if (cmd == 0) {
35             std::cout << "Введите название библиотеки" << std::endl;
36             std::string lib;
37             std::cin >> lib;
38
39             loader.DeleteDerivative(der);
40             loader.DeleteSort(sorter);
41
42             if (!loader.Load(lib)) {
43                 std::cout << "Load error: " << loader.Error() << std::endl;
44                 continue;
45             }
46         }
47     }
48 }
```

```

45    }
46
47    der = loader.CreateDerivative();
48    sorter = loader.CreateSort();
49 }
50 else if (cmd == 1) {
51     float a;
52     float dx;
53     std::cout << "Введите точку" << std::endl;
54     std::cin >> a;
55     std::cout << "Введите отклонение" << std::endl;
56     std::cin >> dx;
57     std::cout << der->Derivative(a, dx) << std::endl;
58 }
59 else if (cmd == 2) {
60     int n;
61     std::cout << "Введите размер массива" << std::endl;
62     std::cin >> n;
63     int * array = new int[n];
64
65     size_t i = 0;
66     std::cout << "Введите элементы массива" << std::endl;
67     for (i = 0; i < n; ++i) std::cin >> array[i];
68
69     sorter->Sort(array, n);
70
71     for (int i = 0; i < n; i++) std::cout << array[i] << " ";
72     std::cout << std::endl;
73
74     delete[] array;
75 }
76 else if (cmd == -1) break;
77 }
78
79 loader.DeleteDerivative(der);
80 loader.DeleteSort(sorter);
81 }
```

Листинг 2: main.cpp (program2) - пример динамического использования библиотек с переключением контрактов

```

1 #include "DynamicLoader.hpp"
2
3 #include <dlfcn.h>
4
5 DynamicLoader::DynamicLoader()
6     : handle_(nullptr), createDer(nullptr), createSort(nullptr),
7       deleteDer(nullptr), deleteSort(nullptr) {}
8
9 DynamicLoader::~DynamicLoader()
```

```

10 | {
11 |     if (handle_) dlclose(handle_);
12 | }
13 |
14 | bool DynamicLoader::Load(const std::string& path)
15 | {
16 |     if (handle_) dlclose(handle_);
17 |     handle_ = dlopen(path.c_str(), RTLD_LAZY);
18 |     if (!handle_)
19 |     {
20 |         lastErr = dlerror();
21 |         return false;
22 |     }
23 |
24 |     createDer = (CreateDerivativeFunc)dlsym(handle_, "CreateDerivative");
25 |     createSort = (CreateSortFunc)dlsym(handle_, "CreateSort");
26 |     deleteDer = (DeleteDerivativeFunc)dlsym(handle_, "DeleteDerivative");
27 |     deleteSort = (DeleteSortFunc)dlsym(handle_, "DeleteSort");
28 |
29 |     if (!createDer || !createSort || !deleteDer || !deleteSort) {
30 |         lastErr = dlerror();
31 |         dlclose(handle_);
32 |         handle_ = nullptr;
33 |         return false;
34 |     }
35 |
36 |     return true;
37 | }
38 |
39 | std::string DynamicLoader::Error() const {
40 |     return lastErr;
41 | }
42 |
43 | AbsDerivative* DynamicLoader::CreateDerivative() {
44 |     return createDer ? createDer() : nullptr;
45 | }
46 |
47 | AbsSort* DynamicLoader::CreateSort() {
48 |     return createSort ? createSort() : nullptr;
49 | }
50 |
51 | void DynamicLoader::DeleteDerivative(AbsDerivative* p) {
52 |     if (deleteDer) deleteDer(p);
53 | }
54 |
55 | void DynamicLoader::DeleteSort(AbsSort* p) {
56 |     if (deleteSort) deleteSort(p);
57 | }

```

Листинг 3: DynamicLoader.cpp - класс, выгружающий библиотеки в память во время

выполнения программы

```
1 #include "Derivative_1.hpp"
2 #include "BubbleSort.hpp"
3 #include "AbsDerivative.hpp"
4 #include "AbsSort.hpp"
5
6 extern "C"
7 {
8     __attribute__((visibility("default"))) AbsDerivative * CreateDerivative()
9     {
10         return new Derivative1();
11     }
12
13     __attribute__((visibility("default"))) AbsSort * CreateSort()
14     {
15         return new BubbleSort();
16     }
17
18     __attribute__((visibility("default"))) void DeleteDerivative(AbsDerivative *
19                     p)
20     {
21         delete p;
22     }
23
24     __attribute__((visibility("default"))) void DeleteSort(AbsSort * p)
25     {
26         delete p;
27     }
}
```

Листинг 4: fact1.cpp - сборщик динамической библиотеки, реализующей контракт 1

```
1 #include "Derivative_2.hpp"
2 #include "QuickSort.hpp"
3 #include "AbsDerivative.hpp"
4 #include "AbsSort.hpp"
5
6 extern "C"
7 {
8     __attribute__((visibility("default"))) AbsDerivative * CreateDerivative()
9     {
10         return new Derivative2();
11     }
12
13     __attribute__((visibility("default"))) AbsSort * CreateSort()
14     {
15         return new QuickSort();
```

```

16    }
17
18    __attribute__((visibility("default"))) void DeleteDerivative(AbsDerivative *
19        p)
20    {
21        delete p;
22    }
23
24    __attribute__((visibility("default"))) void DeleteSort(AbsSort * p)
25    {
26        delete p;
27    }

```

Листинг 5: fact2.cpp - сборщик динамической библиотеки, реализующей контракт 2

```

1 #include "Derivative_1.hpp"
2
3 #include <cmath>
4
5 float Derivative1::Derivative(float A, float deltaX)
6 {
7     return (std::cos(A + deltaX) - std::cos(A)) / deltaX;
8 }

```

Листинг 6: Derivative1.cpp - класс, реализующий 1-й способ нахождения производной $\cos(x)$

```

1 #include "Derivative_2.hpp"
2
3 #include <cmath>
4
5 float Derivative2::Derivative(float A, float deltaX)
6 {
7     return (std::cos(A + deltaX) - std::cos(A - deltaX)) / (2.0f * deltaX);
8 }

```

Листинг 7: Derivative2.cpp - класс, реализующий 2-й способ нахождения производной $\cos(x)$

```

1 #include "BubbleSort.hpp"
2
3 #include <algorithm>
4
5 int * BubbleSort::Sort(int * array, int size)
6 {
7     int i = 0;

```

```

8     int j = 0;
9
10    for (i = 0; i < size - 1; ++i)
11    {
12        for (j = 0; j < size - i - 1; ++j)
13        {
14            if (array[j] > array[j + 1])
15            {
16                std::swap(array[j], array[j + 1]);
17            }
18        }
19    }
20
21    return array;
22 }
```

Листинг 8: BubbleSort.cpp - класс, алгоритм пузырьковой сортировки для целочисленных массивов

```

1 #include "QuickSort.hpp"
2
3 #include <algorithm>
4
5 void QuickSort::quicksort(int * array, int low, int high)
6 {
7     int i = low;
8     int j = high;
9     int pivot = array[high];
10
11    while (i <= j)
12    {
13        while (array[i] < pivot) ++i;
14        while (array[j] < pivot) ++j;
15
16        if (i <= j)
17        {
18            std::swap(array[i], array[j]);
19            ++i;
20            --j;
21        }
22    }
23
24    if (low < j) quicksort(array, low, j);
25    if (high > i) quicksort(array, high, i);
26 }
27
28 int * QuickSort::Sort(int * array, int size)
29 {
30     if (size > 1)
31     {
```

```

32     quicksort(array, 0, size - 1);
33 }
34
35 return array;
36 }
```

Листинг 9: QuickSort.cpp - класс, алгоритм быстрой сортировки для целочисленных массивов

```

1 #pragma once
2
3 class AbsDerivative
4 {
5     public:
6         virtual float Derivative(float A, float deltaX) = 0;
7         virtual ~AbsDerivative() = default;
8 };
```

Листинг 10: AbsDerivative.hpp - абстрактный класс для реализации производной косинуса

```

1 #pragma once
2
3 class AbsSort
4 {
5     public:
6         virtual int * Sort(int * array, int size) = 0;
7         virtual ~AbsSort() = default;
8 };
```

Листинг 11: AbsSort.hpp - абстрактный класс для реализации сортировки целочисленных массивов

Системные вызовы статической библиотеки

```

execve("./program1", ["/./program1"], 0x7ffd9a10e080 /* 36 vars */) = 0
brk(NULL)                               = 0x55a72a072000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7be415b94000
access("/etc/ld.so.preload", R_OK)        = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/daalginichev/OS/Lab_4/build/glibc-hwcaps/x86-64-v3/libcontract",
      = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/daalginichev/OS/Lab_4/build/glibc-hwcaps/x86-64-v3/", 0x7ff
      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/daalginichev/OS/Lab_4/build/glibc-hwcaps/x86-64-v2/libcontract",
      = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/daalginichev/OS/Lab_4/build/glibc-hwcaps/x86-64-v2/", 0x7ff
      = -1 ENOENT (No such file or directory)
```



```
getrandom("\xf4\x58\xc1\xde\x95\xd0\x9b\x3e", 8, GRND_NONBLOCK) = 8
brk(NULL) = 0x55a72a072000
brk(0x55a72a093000) = 0x55a72a093000
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88,0x5), ...}) = 0
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265,\321\207\321\202\320\265\321\205\320\276\321\202\320\270\321"..., 51) = 51
write(1, "0 \320\232\320\276\320\275\320\265\321\206 \320\277\321\200\320\276\320\263\321"..., 32)
write(1, "1 \320\237\321\200\320\276\320\270\320\267\320\262\320\276\320\264\320\275\321"..., 25)
write(1, "2 \320\241\320\276\321\200\321\202\320\270\321\200\320\276\320\262\320\272\321"..., 23)
fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88,0x5), ...}) = 0
read(0, "1\n", 1024) = 2
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \321\202\320\276\321\205\320\276\321\202\320\270\321"..., 26)
read(0, "0\n", 1024) = 2
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\276\321\202\320\276\321\205\320\276\321\202\320\270\321"..., 36)
read(0, "0.00000001\n", 1024) = 11
write(1, "0\n", 2) = 2
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265,\321\207\321\202\320\265\321\205\320\276\321\202\320\270\321"..., 51)
write(1, "0 \320\232\320\276\320\275\320\265\321\206 \320\277\321\200\320\276\320\263\321"..., 32)
write(1, "1 \320\237\321\200\320\276\320\270\320\267\320\262\320\276\320\264\320\275\321"..., 25)
write(1, "2 \320\241\320\276\321\200\321\202\320\270\321\200\320\276\320\262\320\272\321"..., 23)
read(0, "2\n", 1024) = 2
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \321\200\320\260\320\274\320\260"..., 43)
read(0, "5\n", 1024) = 2
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \321\215\320\273\321"..., 47) = 47
read(0, "5 4 3 2 1\n", 1024) = 10
write(1, "1 2 3 4 5 \n", 11) = 11
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265,\321\207\321\202\320\265\321\205\320\276\321\202\320\270\321"..., 51)
write(1, "0 \320\232\320\276\320\275\320\265\321\206 \320\277\321\200\320\276\320\263\321"..., 32)
write(1, "1 \320\237\321\200\320\276\320\270\320\267\320\262\320\276\320\264\320\275\321"..., 25)
write(1, "2 \320\241\320\276\321\200\321\202\320\270\321\200\320\276\320\262\320\272\321"..., 23)
read(0, "0\n", 1024) = 2
lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Illegal seek)
exit_group(0) = ?
```

+++ exited with 0 +++

Системные вызовы динамических библиотек


```
mmap(0x725ea6604000,4096,PROT_READ,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0x3000)
= 0x725ea6604000
mmap(0x725ea6605000,8192,PROT_READ|PROT_WRITE,MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,3,0
= 0x725ea6605000
close(3) = 0
mprotect(0x725ea6605000,4096,PROT_READ) = 0
fstat(1,{st_mode=S_IFCHR|0620,st_rdev=makedev(0x88,0x5),...}) = 0
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265,\\321\\207\\321\\202\\32
\\321\\205\\320\\276\\321\\202\\320\\270\\321\"...,51) = 51
write(1,"-1 \\320\\232\\320\\276\\320\\275\\320\\265\\321\\206 \\320\\277\\321\\200\\320\\276\\320\\263
= 33
write(1,"0 \\320\\241\\320\\274\\320\\265\\320\\275\\320\\260 \\320\\261\\320\\270\\320\\261\\320\\273\\
= 34
write(1,"1 \\320\\237\\321\\200\\320\\276\\320\\270\\320\\267\\320\\262\\320\\276\\320\\264\\320\\275\\3
= 25
write(1,"2 \\320\\241\\320\\276\\321\\200\\321\\202\\320\\270\\321\\200\\320\\276\\320\\262\\320\\272\\3
= 23
fstat(0,{st_mode=S_IFCHR|0620,st_rdev=makedev(0x88,0x5),...}) = 0
read(0,"1\\n",1024) = 2
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265 \\321\\202\\320\\276\\32
= 26
read(0,"0\\n",1024) = 2
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265 \\320\\276\\321\\202\\32
= 36
read(0,"0.0000000001\\n",1024) = 14
write(1,"0\\n",2) = 2
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265,\\321\\207\\321\\202\\32
\\321\\205\\320\\276\\321\\202\\320\\270\\321\"...,51) = 51
write(1,"-1 \\320\\232\\320\\276\\320\\275\\320\\265\\321\\206 \\320\\277\\321\\200\\320\\276\\320\\263
= 33
write(1,"0 \\320\\241\\320\\274\\320\\265\\320\\275\\320\\260 \\320\\261\\320\\270\\320\\261\\320\\273\\
= 34
write(1,"1 \\320\\237\\321\\200\\320\\276\\320\\270\\320\\267\\320\\262\\320\\276\\320\\264\\320\\275\\3
= 25
write(1,"2 \\320\\241\\320\\276\\321\\200\\321\\202\\320\\270\\321\\200\\320\\276\\320\\262\\320\\272\\3
= 23
read(0,"2\\n",1024) = 2
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265 \\321\\200\\320\\260\\32
\\320\\274\\320\\260\"...,43) = 43
read(0,"5\\n",1024) = 2
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265 \\321\\215\\320\\273\\32
\"...,47) = 47
read(0,"5 4 3 2 1\\n",1024) = 10
write(1,"1 2 3 4 5 \\n",11) = 11
write(1,"\\320\\222\\320\\262\\320\\265\\320\\264\\320\\270\\321\\202\\320\\265,\\321\\207\\321\\202\\32
\\321\\205\\320\\276\\321\\202\\320\\270\\321\"...,51) = 51
write(1,"-1 \\320\\232\\320\\276\\320\\275\\320\\265\\321\\206 \\320\\277\\321\\200\\320\\276\\320\\263
= 33
```



```
= 34
write(1,"1 \320\237\321\200\320\276\320\270\320\267\320\262\320\276\320\264\320\275\320\274\320\260"...,43) = 43
read(0,"2\n",1024) = 2
write(1,"\\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \\321\200\320\260\320\274\320\260"...,43) = 43
read(0,"5\n",1024) = 2
write(1,"\\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \\321\215\320\273\320\274\320\260"...,47) = 47
read(0,"5 4 3 2 1\n",1024) = 10
write(1,"1 2 3 4 5 \n",11) = 11
write(1,"\\320\222\320\262\320\265\320\264\320\270\321\202\320\265,\\321\207\321\202\320\271\320\260\320\276\321\202\320\270\321"...,51) = 51
write(1,"-1 \\320\232\320\276\320\275\320\265\321\206 \\320\277\321\200\320\276\320\263") = 33
write(1,"0 \\320\241\320\274\320\265\320\275\320\260 \\320\261\320\270\320\261\320\273\320\274\320\260"...,34) = 34
write(1,"1 \\320\237\321\200\320\276\320\270\320\267\320\262\320\276\320\264\320\275\320\274\320\260"...,25) = 25
write(1,"2 \\320\241\320\276\321\200\320\270\321\202\320\270\321\200\320\276\320\262\320\272\320\274\320\260"...,23) = 23
read(0,"-1\n",1024) = 3
munmap(0x725ea6601000,20640) = 0
lseek(0,-1,SEEK_CUR) = -1 ESPIPE (Illegal seek)
exit_group(0) = ?
+++ exited with 0 +++
```