

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»**

Группа: М80-206Б-22

Студент: Ларченко А.О.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_

Дата: 29.12.23

Москва, 2023

## Постановка задачи

Требуется создать динамические библиотеки, которые реализуют определенный функционал. Далее использовать данные библиотеки 2-мя способами:

- Во время компиляции (на этапе «линковки»/linking)
- 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 использует 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. Пользовательский ввод для обеих программ должен быть организован следующим образом:

- Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2). Можно реализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»;
- «1 arg1 arg2 ... argN», где после «1» идут аргументы для первой 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 контрактами. После ввода команды происходит вызов второй функции, и на экране появляется результат её выполнения.

### Вариант 15.

Функции	Сигнатура	Реализация 1	Реализация 2
2. Расчет производной функции $\cos(x)$ в точке A с приращением $\Delta x$	Float Derivative (float A, float $\Delta x$ )	$f'(x) = (f(A + \Delta x) - f(A))/\Delta x$	$f'(x) = (f(A + \Delta x) - f(A - \Delta x))/(2 * \Delta x)$
9. Отсортировать целочисленный массив	Int * Sort(int * array)	Пузырьковая сортировка	Сортировка Хоара

## Общий метод и алгоритм решения

Использованные системные вызовы:

- `hld = void *dlopen(const char *filename, int flag)`. **dlopen** загружает динамическую библиотеку, имя которой указано в строке filename, и возвращает прямой указатель на начало динамической библиотеки. Если filename не является полным именем файла (т.е. не

начинается с "/"), то файл ищется в следующих местах(1) в разделенном двоеточием списке каталогов, в переменной окружения пользователя LD\_LIBRARY\_PATH, 2) В списке библиотек, кэшированных в файле /etc/ld.so.cache 3) В /usr/lib и далее в /lib. 4)Если filename указывает на NULL, то возвращается указатель на основную программу.); hld- указатель на библиотеку.

- void\* *dlsym*(hld, "function name") - поиск адреса функции в библиотеке
- 

## Код программы

### Makefile

```
MATH_FLAG = -lm
PIC_FLAG = -fPIC
SHARED_FLAG = -shared
DYNAMIC_LOADING_FLAG = -ldl
EXT = c
CC = gcc
CFLAGS = -std=c99 -pedantic -Wall

all: stat_main dynamic

stat_main: stat_main.$(EXT) first_realization.$(EXT) function.h
    $(CC) $(CFLAGS) stat_main.$(EXT) first_realization.$(EXT) -o stat_main
    $(MATH_FLAG)

dynamic: dyn_main dynamic_realization1 dynamic_realization2
    $(CC) $(CFLAGS) dyn_main.o -o dynamic $(DYNAMIC_LOADING_FLAG)

dyn_main : dyn_main.$(EXT)
    $(CC) $(CFLAGS) -c dyn_main.$(EXT)

dynamic_realization2: second_realization.$(EXT) function.h
    $(CC) $(CFLAGS) $(PIC_FLAG) $(SHARED_FLAG) -o libsecond_realization.so
    second_realization.$(EXT) $(MATH_FLAG)

dynamic_realization1: first_realization.$(EXT) function.h
    $(CC) $(CFLAGS) $(PIC_FLAG) $(SHARED_FLAG) -o libfirst_realization.so
    first_realization.$(EXT) $(MATH_FLAG)

clean :
    rm *.o *.so stat_main dynamic
```

### Пояснения:

*dynamic\_realization1, dynamic\_realization2* - собираем не объектные файлы, а динамические библиотеки. Используем нестандартные флаги:

- -fPIC (генерация position independent code, используем т.к. пишем на 64-разрядной системе)
- -shared (флаг разделяемой библиотеки)

Тут же стоит отметить, что все библиотеки в си имеют стандарт оформления(lib\*libname\*.so либо можно заменить -l\*libname\*).

*dyn\_main* - компилируем наш динамический main(dyn\_main.c), флаг -с обозначает, что компилируем без этапа линковки!!

В *dynamic*: -ldl - флаг-обозначение динамической библиотеки для линковщика, который находит указатель на эту библиотеку и связывает его с именем файла

*stat\_main* - это база

### first\_realization.c

```
#include <stdio.h>
```

```
#include "function.h"
```

```
float Derivative(float a, float dx){ // на вход получаем угол в градусах, а приращение в радианах
```

```
    float answ=(cosf((a*PI)/180+dx)- cosf((a*PI)/180))/dx;
```

```
    return answ;
```

```
}
```

```
void Sort(int *array, int size){
```

```
    // printf("obobus");
```

```
    // for(int i=0; i<size; ++i){
```

```
    //     printf("%d ", array[i]);
```

```
    // }
```

```
    printf("Bubblesort\n");
```

```
    for(int i=0; i< size;++i){
```

```
        int len1=size-i;
```

```
        bool change_stat=false;
```

```
        for(int j=0; j<len1-1;++j){
```

```
            if(array[j]>array[j+1]){
```

```
                int tmp=array[j+1];
```

```
                array[j+1]=array[j];
```

```
                array[j]=tmp;
```

```
                change_stat=true;
```

```
            }
```

```
        }
```

```
        if(change_stat==false){
```

```
            break;
```

```
    }  
}  
  
}
```

## **second\_realization.c**

```
#include <stdio.h>
```

```
#include "function.h"
```

```
float Derivative(float a, float dx){ // на вход получаем угол в градусах, а приращение в радианах
```

```
    float answ=(cosf((a*PI)/180+dx)- cosf((a*PI)/180-dx))/(2*dx);
```

```
    return answ;
```

```
}
```

```
int minim(int a, int b, int c){
```

```
    int mn;
```

```
    if (a<b){
```

```
        mn= a;
```

```
    }else {
```

```
        mn =b;
```

```
    }
```

```
    if (c<mn){
```

```
        mn=c;
```

```
    }
```

```
    return mn;
```

```
}
```

```
int maxim(int a, int b, int c){
```

```
    int mx;
```

```
    if(a>b){
```

```
        mx=a;
```

```
    } else{
```

```
        mx=b;
```

```
    }
```

```

    if(mx<c){

        mx=c;

    }

    return mx;

}

```

```

void swap(int *array, int i1, int i2){

    if(i1 == i2){

        return;

    }

    int tmp = array[i1];

    array[i1]=array[i2];

    array[i2]=tmp;

}

```

```

int find_pivot(int *array, int size){

    int a,b,c, pivot;

    a=array[0];

    b=array[size-1];

    c=array[size/2];

    pivot=a+b+c-maxim(a,b,c)-minim(a,b,c);

    // printf("Pivot = %d\n", pivot);

    return pivot;

}

```

```

void partition(int *array, int *kf_i, int *kf_k, int size){

    int pivot=find_pivot(array, size);

    int i=0;

    int j=0;

    int k=0;

    while(j<size){

        if(array[j]<pivot){

```

```

        swap(array, i, j);

        if(i!=k){

            swap(array, k, j);

        }

        i++;

        k++;

    } else if(array[j]==pivot){

        swap(array, k, j);

        k++;

    }

    j++;

}

*kf_k=k;

*kf_i=i;

}

```

```

void quicksort(int *array, int size){

    if(size <2){

        return;

    } else if(size==2){

        if(array[0]>array[1]){

            swap(array, 0, 1);

        }

        return;

    }

    int i,k;

    partition(array, &i, &k, size);

    quicksort(array, i);

    quicksort(&array[k], size-k);

}

```

```

void Sort(int *array, int size){

    printf("Quicksort\n");

    quicksort(array, size);

}

```

### **stat\_main.c**

```

#include <stdio.h>

#include "function.h"

#include <stdbool.h>


int main(){

    int com_numb;

    while(true){

        printf("\nChoose comand. Press: \n- 1 for cos derivative\n- 2 for sorting
array\n");

        if(scanf("%d", &com_numb)==EOF){

            printf("\nProgram has been ended. Bye!\n");

            break;

        }

        switch(com_numb){

            case 1:

                {

                    float angle, dx;

                    printf("Input angle in degrees ");

                    scanf("%f", &angle);

                    printf("Input Dx in radians ");

                    scanf("%f", &dx);

                    float answ=Derivative(angle, dx);

                    printf("Answ: %f\n", answ);

                    break;

                }

            case 2:

                {

```



```

        int a_size;

        printf("Input array size ");

        scanf("%d", &a_size);

        int array[a_size];

        printf("\nInput elements: ");

        for(int i=0; i<a_size;++i){

            int tmp;

            scanf("%d", &tmp);

            array[i]=tmp;

            // printf("hh%d ", array[i]);

            // printf("hui%d ", i);

        }

        Sort(array, a_size);

        printf("Array was sorted: ");

        for(int i=0; i<a_size;++i){

            printf("% d", array[i]);

        }

        break;

    }

    default:

        printf("Uncorrect inputting. Try again\n");

        break;

}

}

}

```

### **dyn\_main.c**

```

#include <stdio.h>

#include <stdbool.h>

#include <dlfcn.h>

#include <stddef.h>

```

```

#include <stdlib.h>

int main(){

    void *hld=dlopen("./libfirst_realization.so", RTLD_LAZY);

    if(hld==NULL){

        fputs(dlerror(), stderr);

        exit(-1);

    }

    int reliz_numb=1;

    int com_numb;

    char *error;

    while(true){

        printf("\nChoose comand. Press: \n- 0 for changing realization \n- 1 for cos
derivative\n- 2 for sorting array\n");

        if(scanf("%d", &com_numb)==EOF){

            printf("\nProgram has been ended. Bye!\n");

            break;

        }

        switch(com_numb){

            case 0:

                {

                    if(dlclose(hld)!=0){

                        fputs(dlerror(), stderr);

                        exit(-1);

                    }

                    int last_reliz_numb=reliz_numb;

                    if(reliz_numb==1){

                        hld=dlopen("./libsecond_realization.so", RTLD_LAZY);

                        reliz_numb=2;

                    } else{

                        hld=dlopen("./libfirst_realization.so", RTLD_LAZY);

                        reliz_numb=1;

                    }

                    if(hld==NULL){

```

```

        fputs(dlerror(), stderr);

        exit(-1);

    }

    printf("Realization was changed from realization%d to realization%d.\n",
last_reliz_numb ,reliz_numb);

    break;

}

case 1:

{

    float angle, dx;

    printf("Input angle in degrees ");

    scanf("%f", &angle);

    printf("Input Dx in radians ");

    scanf("%f", &dx);


    float (*Derivative)(float, float);

    *(float **) (&Derivative)=dlsym(hld, "Derivative");

    if ((error = dlerror()) != NULL)  {

        fprintf(stderr, "%s\n", error);

        exit(1);

    }


    float answ=Derivative(angle, dx);

    printf("Answ: %f\n", answ);

    break;

}

case 2:

{

    int a_size;

    printf("Input array size ");

    scanf("%d", &a_size);

    int array[a_size];

    printf("\nInput elements: ");

```

```

    for(int i=0; i<a_size;++i){

        int tmp;

        scanf("%d", &tmp);

        array[i]=tmp;

        // printf("hh%d ", array[i]);

        // printf("hui%d ", i);

    }


    void (*Sort)(int *, int);

    *(void **)(&Sort) = dlsym(hld, "Sort");

    if ((error = dlerror()) != NULL)  {

        fprintf(stderr, "%s\n", error);

        exit(1);

    }


    Sort(array, a_size);

    printf("Array was sorted: ");

    for(int i=0; i<a_size;++i){

        printf("% d", array[i]);

    }

    break;

}

default:

    printf("Uncorrect inputting. Try again\n");

    break;

}

}

if (dlclose(hld) != 0) {

    perror("dlclose");

    exit(1);

```

}

}

## Протокол работы программы

### Тестирование:

```
arsenii@PC-Larchal4:~/Documents/VS_code_prog/OSI/laba_4$ ./dynamic
```

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

1

Input angle in degrees 60

Input Dx in radians 0.001

Answ: -0.866294

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

2

Input array size 5

Input elements: 4 9 2 1 4

### **Bubblesort**

Array was sorted: 1 2 4 4 9

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

0

Realization was changed from realization1 to realization2.

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

2

Input array size 6

Input elements: 4 9

2 3 1 1

### **Quicksort**

Array was sorted: 1 1 2 3 4 9

Choose comand. Press:

- 0 for changing realization

- 1 for cos derivative
- 2 for sorting array

Program has been ended. Bye!

arsenii@PC-Larcha14:~/Documents/VS\_code\_prog/OSI/laba\_4\$ ./dynamic

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

1

Input angle in degrees 60

Input Dx in radians 0.001

Answ: -0.866294

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

2

Input array size 5

Input elements: 4 9 2 1 4

Bubblesort

Array was sorted: 1 2 4 4 9

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

0

Realization was changed from realization1 to realization2.

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative
- 2 for sorting array

2

Input array size 6

Input elements: 4 9

2 3 1 1

Quicksort

Array was sorted: 1 1 2 3 4 9

Choose comand. Press:

- 0 for changing realization
- 1 for cos derivative



```

0 prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) =
munmap(0x7f1072bc3000, 80671) = 0
getrandom("\xc1\x4a\x8a\xbd\x71\x53\xad\xf2", 8, GRND_NONBLOCK) = 8
brk(NULL) = 0x556daa872000
brk(0x556daa893000) = 0x556daa893000
openat(AT_FDCWD, "./libfirst_realization.so", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0775, st_size=15664, ...}, AT_EMPTY_PATH) = 0
getcwd("/home/arsenii/Documents/VS_code_prog/OSI/laba_4", 128) = 48
mmap(NULL, 16440, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f1072bd2000
mmap(0x7f1072bd3000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f1072bd3000
mmap(0x7f1072bd4000, 4096, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f1072bd4000
mmap(0x7f1072bd5000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f1072bd5000
close(3) = 0
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=80671, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 80671, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f1072bac000
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\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f1072ac5000
mmap(0x7f1072ad3000, 507904, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7f1072ad3000
mmap(0x7f1072b4f000, 372736, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8a000) = 0x7f1072b4f000
mmap(0x7f1072baa000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7f1072baa000
close(3) = 0
mprotect(0x7f1072baa000, 4096, PROT_READ) = 0
mprotect(0x7f1072bd5000, 4096, PROT_READ) = 0
munmap(0x7f1072bac000, 80671) = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH)
= 0
write(1, "\n", 1
) = 1
write(1, "Choose comand. Press: \n", 23Choose comand. Press:
) = 23
write(1, "- 0 for changing realization \n", 30- 0 for changing realization
) = 30
write(1, "- 1 for cos derivative\n", 23- 1 for cos derivative

```



```

) = 23
write(1, "- 2 for sorting array\n", 22- 2 for sorting array
) = 22
= 0 newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH)
read(0, 1
"1\n", 1024) = 2
write(1, "Input angle in degrees ", 23Input angle in degrees ) = 23
read(0, 60
"60\n", 1024) = 3
write(1, "Input Dx in radians ", 20Input Dx in radians ) = 20
read(0, 0.001
"0.001\n", 1024) = 6
write(1, "Answ: -0.866294\n", 16Answ: -0.866294
) = 16
write(1, "\nChoose comand. Press: \n- 0 for "..., 77и
Choose comand. Press:

```

- 0 for changing realization

- 1 for cos derivative

```
) = 77
```

```
write(1, "- 2 for sorting array\n", 22- 2 for sorting array
```

```
) = 22
```

```
read(0, 0
```

```
"0\n", 1024) = 2
```

```
munmap(0x7f1072bd2000, 16440) = 0
```

```
munmap(0x7f1072ac5000, 942344) = 0
```

```
openat(AT_FDCWD, "/libsecond_realization.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
```

```
newfstatat(3, "", {st_mode=S_IFREG|0775, st_size=16008, ...}, AT_EMPTY_PATH) = 0
```

```
getcwd("/home/arsenii/Documents/VS_code_prog/OSI/laba_4", 128) = 48
```

```
mmap(NULL, 16496, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f1072bd2000
```

```
mmap(0x7f1072bd3000, 4096, PROT_READ|PROT_EXEC,
```

```
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f1072bd3000
```

```
mmap(0x7f1072bd4000, 4096, PROT_READ,
```

```
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f1072bd4000
```

```
mmap(0x7f1072bd5000, 8192, PROT_READ|PROT_WRITE,
```

```
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f1072bd5000
```

```
close(3) = 0
```

```
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
```

```
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=80671, ...}, AT_EMPTY_PATH) = 0
```

```
mmap(NULL, 80671, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f1072bac000
```

```
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\3\0>\0\1\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f1072ac5000
mmap(0x7f1072ad3000, 507904, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7f1072ad3000
mmap(0x7f1072b4f000, 372736, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8a000) = 0x7f1072b4f000
mmap(0x7f1072baa000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7f1072baa000
close(3) = 0
mprotect(0x7f1072baa000, 4096, PROT_READ) = 0
mprotect(0x7f1072bd5000, 4096, PROT_READ) = 0
munmap(0x7f1072bac000, 80671) = 0
write(1, "Realization was changed from rea"..., 59Realization was changed from realization1 to
realization2.
) = 59
write(1, "\nChoose comand. Press: \n- 0 for "..., 77
Choose comand. Press:
- 0 for changing realization
- 1 for cos derivative
) = 77
write(1, "- 2 for sorting array\n", 22- 2 for sorting array
) = 22
read(0, 1
"1\n", 1024) = 2
write(1, "Input angle in degrees ", 23Input angle in degrees ) = 23
read(0, 60
"60\n", 1024) = 3
write(1, "Input Dx in radians ", 20Input Dx in radians ) = 20
read(0, 0.001
"0.001\n", 1024) = 6
write(1, "Answ: -0.866026\n", 16Answ: -0.866026
) = 16
write(1, "\nChoose comand. Press: \n- 0 for "..., 77
Choose comand. Press:
- 0 for changing realization
- 1 for cos derivative
) = 77
write(1, "- 2 for sorting array\n", 22- 2 for sorting array
) = 22
read(0, "", 1024) = 0
write(1, "\n", 1
) = 1
write(1, "Program has been ended. Bye!\n", 29Program has been ended. Bye!
) = 29

```

```
munmap(0x7f1072bd2000, 16496)      = 0
munmap(0x7f1072ac5000, 942344)    = 0
exit_group(0)                     = ?
+++ exited with 0 +++
```

## Вывод

В ходе работы над данной лабораторной, я познакомился с динамическими библиотеками в си, а также научился их применять.

Данная работа особых сложностей не вызвала.

В итоге у меня получился исправно работающий код, считаю, что с поставленной задачей справился успешно.