МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем и технологий

**Отчёт по лабораторной работе № 25(9)**

по дисциплине: ”Системное программирование”

на тему: «***Создание и использование***

***библиотек в Linux»***

Вариант -

Выполнил**:** студент группы 10701222 Зухта К. М.

Медведский Е. М.

Принял**:** пр. Давыденко Н.В.

# Лабораторная работа № 25.

## Задание 1.

Произвести рефакторинг проекта предыдущей лабораторной работы по

автосборке проекта: вынесите код функций бизнес логики в отдельную

статическую библиотеку. Для автосборки проекта используйте утилиту make,

обязательно.

При помощи программы ldd проверим зависимости, образовавшиеся между биб-

лиотеками.

Сравните сумму размеров объектных файлов проекта и исполняемого файла.

Результаты отобразите скриншотами.

Копию исполняемого файла расположите на «Рабочем столе», запустите про-

грамму. Какой будет результат?

**Листинг кода**

#### arithmetic.h (объявления функций для арифметических операций)

#ifndef ARITHMETIC\_H

#define ARITHMETIC\_H

int add(int a, int b);

int subtract(int a, int b);

#endif

#### arithmetic.c (реализует функции, объявленные в arithmetic.h)

#include "arithmetic.h"

int add(int a, int b) {

return a + b;

}

int subtract(int a, int b) {

return a - b;

}

#### main.c(основная логика программы)

#include <stdio.h>

#include "arithmetic.h"

int main() {

int a = 5, b = 3;

printf("Add: %d\n", add(a, b));

printf("Subtract: %d\n", subtract(a, b));

return 0;

}

#### Makefile(

CC=gcc

CFLAGS=-Iinclude

OBJ=src/main.o src/arithmetic.o

LIB=libarithmetic.a

# Основная цель

all: $(LIB) my\_program

# Правило для создания статической библиотеки

$(LIB): src/arithmetic.o

ar rcs $@ $^

# Правило для сборки исполняемого файла

my\_program: $(OBJ) $(LIB)

$(CC) -o $@ $(OBJ) -L. -larithmetic

# Правила для сборки объектных файлов

src/main.o: src/main.c include/arithmetic.h

$(CC) $(CFLAGS) -c src/main.c -o src/main.o

src/arithmetic.o: src/arithmetic.c include/arithmetic.h

$(CC) $(CFLAGS) -c src/arithmetic.c -o src/arithmetic.o

# Очистка

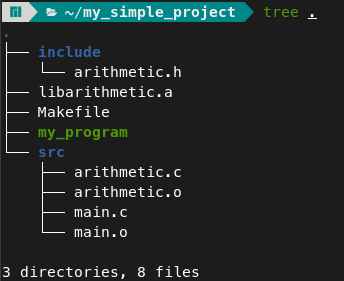
clean:

rm -f src/\*.o \*.a my\_program

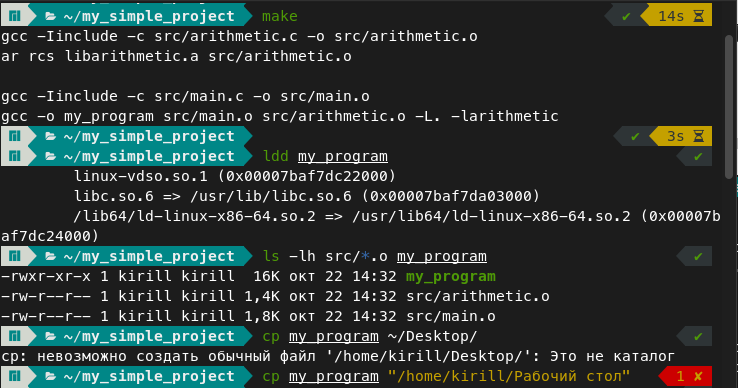
**Скриншоты**

****

Создание файлов arithmetic.h, arithmetic.c, main.c, Makefile.



Структура файлов



1. Собрали проект make

2. Проверили зависимости ldd my\_program

3. Проверка размеров объектных файлов и исполняемого файлаls -lh \*.o my\_program

4. Скопировали файл на рабочий стол cp my\_program «////»

5. Запуск программы.



При запуске программы она выполняет следующие действия:

1. **Сложение**: Вычисляет сумму двух чисел (5 + 3 = 8) и выводит результат.
2. **Вычитание**: Вычисляет разность двух чисел (5 - 3 = 2) и выводит результат

Задание 2.

Произвести рефакторинг проекта предыдущей лабораторной работы по

автосборке проекта: вынесите код функций бизнес логики в отдельную

динамическую библиотеку со статическим вызовом. Для автосборки проекта

используйте утилиту make, обязательно.

Создание и использование библиотек в Linux

При помощи программы ldd проверим зависимости, образовавшиеся между биб-

лиотеками.

Сравните сумму размеров объектных файлов проекта и исполняемого файла.

Результаты отобразите скриншотами.

Копию исполняемого файла расположите на «Рабочем столе», запустите про-

грамму. Какой будет результат? Если возникла проблема, решите ее и поясните

каким способом.

Сравните результаты первого и второго упражнения, сделайте вывод.

### Листинг кода

Измененный файл makefile

CC=gcc

CFLAGS=-Iinclude -fPIC

OBJ=src/arithmetic.o src/main.o

LIB=libarithmetic.so

# Основная цель

all: $(LIB) my\_program

# Правило для создания динамической библиотеки

$(LIB): src/arithmetic.o

$(CC) -shared -o $@ $^

# Правило для сборки исполняемого файла

my\_program: src/main.o $(LIB)

$(CC) -o $@ src/main.o -L. -larithmetic

# Правила для сборки объектных файлов

src/main.o: src/main.c include/arithmetic.h

$(CC) $(CFLAGS) -c src/main.c -o src/main.o

src/arithmetic.o: src/arithmetic.c include/arithmetic.h

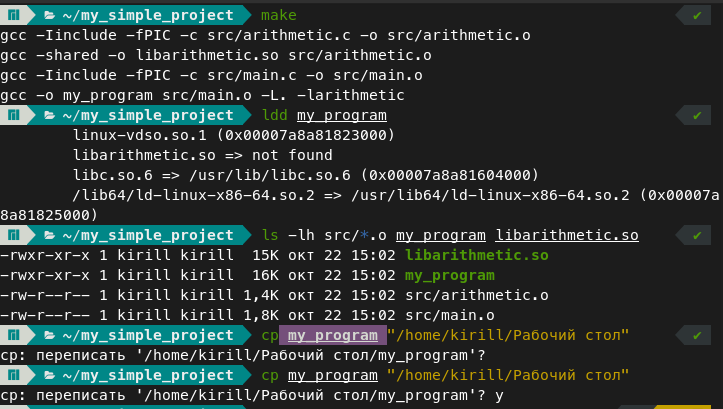
$(CC) $(CFLAGS) -c src/arithmetic.c -o src/arithmetic.o

# Очистка

clean:

rm -f src/\*.o \*.so my\_program

Скриншоты



1. Собрали проект make

2. Проверили зависимости ldd my\_program

3. Проверка размеров объектных файлов и исполняемого файла ls -lh src/\*.o my\_program libarithmetic.so

4. Скопировали файл на рабочий стол cp my\_program «////»

5. Запуск программы.



Сравните результаты первого и второго упражнений:

1. **Первый проект** использует статическую библиотеку, тогда как второй проект использует динамическую библиотеку.
2. **Размеры файлов**: В динамической библиотеке размер может быть меньше, так как она может использоваться несколькими программами одновременно.
3. **Зависимости**: Динамическая библиотека требует, чтобы библиотека была доступна во время выполнения программы, в отличие от статической, которая включается в исполняемый файл во время компиляции.

Задание 3

Произвести рефакторинг проекта предыдущей лабораторной работы по

автосборке проекта: вынесите код функций бизнес логики в отдельную

динамическую библиотеку с динамическим вызовом. Для автосборки проекта

используйте утилиту make, обязательно.

При помощи программы ldd проверим зависимости, образовавшиеся между биб-

лиотеками.

Сравните сумму размеров объектных файлов проекта и исполняемого файла.

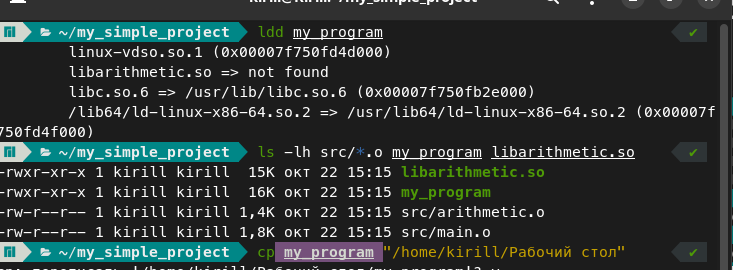
Результаты отобразите скриншотами.

Копию исполняемого файла расположите на «Рабочем столе», запустите про-

грамму. Какой будет результат? Если возникла проблема, решите ее, поясните

причину проблемы и каким способом вы ее решили.

Сравните результаты с предыдущими, сделайте вывод.



1. Собрали проект make

2. Проверили зависимости ldd my\_program

3. Проверка размеров объектных файлов и исполняемого файла ls -lh src/\*.o my\_program libarithmetic.so

4. Скопировали файл на рабочий стол cp my\_program «////»

5. Запуск программы.



Контрольные вопросы:

Зухта

7. Как создать статическая и динамическую библиотеки и подключить их к программе?

Шаги: Статическая

* 1. Создаем файл библиотеки libr,c
  2. Скомпилируем библ gcc… (ar rcs libmylib.a libr.o )
  3. Создадим программу main.c
  4. Скомпилируем программу gcc main.c -L. -libr -o myprogram

Динамическая

* + 1. Создаем такой же файл библиотеки
    2. Скомпилируем библиотеку (создаем динамическую библ )
    3. Создаем программу
    4. Компилируем программу

Медведский

10. Если в целевом каталоге присутсвуют два типа одной и той же библиотеки, какая из них будет подключаться всегда по умолчанию ? А как явно подключить вторую ?

Ответ:

Если в целевом каталоге присутсвуют два типа одной и той же библиотеки, то при компиляции по умолчанию будет подключатся динамическая библиотека.

Чтобы явно подключить статистическую библиотеку мы можем использовать флаг -Wl, --whole-archive.