МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем

и технологий

**Отчет**

**по лабораторной работе № 8**

по дисциплине: ”Системное программирование”

на тему: ***”*** **Создание и использование библиотек в Linux*”***

Выполнил**:** студент группы *10702121* Писарик А.С

Филанович Д.В.

Принял**:** пр. Давыденко Н. В.

Минск 2023

# Лабораторная работа №8.

**Цель работы:** Изучить и закрепить на практике создание и использование статически и динамически подключаемых библиотек в операционных системах семейства Linux.

## 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 библиотеками.

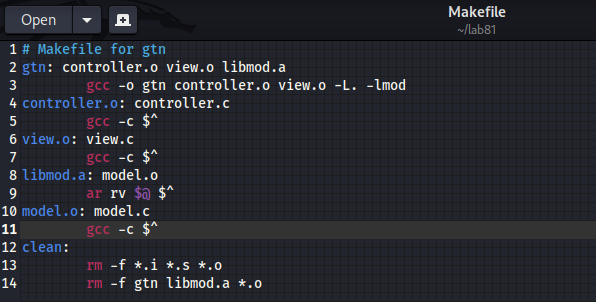
Сравните сумму размеров объектных файлов проекта и исполняемого файла.

Результаты отобразите скриншотами.

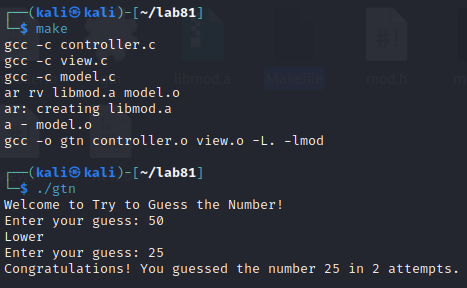
Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат?

Решение

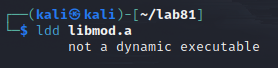
**Создаем Makefile.**



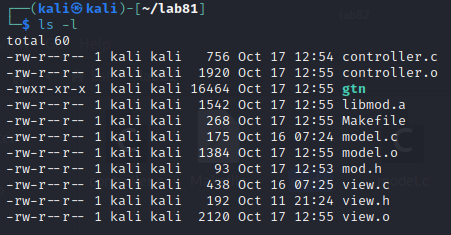
**При помощи утилиты make выполняем автосборку статической библиотеки и проекта и проверяем работоспособность.**



**При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками.**

 мы не можем это сделать, так как у нас статическая библиотека.

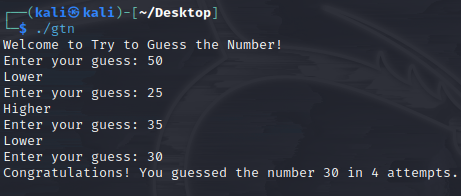
**Сравниваем сумму размеров объектных файлов проекта и исполняемого файла.**



Сумма объёма объектных файлов будет меньше(4.4кб), чем исполняемый файл(16.1кб).

Скорее всего, это за счет наличия разного рода управляющей информации, определения адресов расположения присоединенных к нему функций и т.д.

**Располагаем исполняемый файл на рабочем столе и запускаем программу.**



Программа запустилась успешно!

## 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.

При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками.

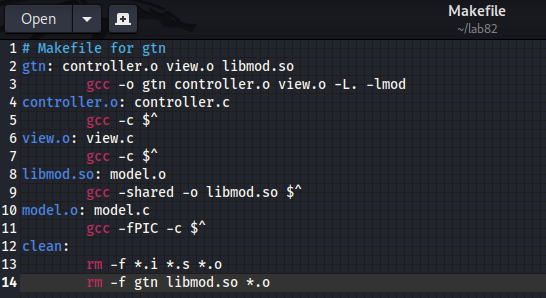
Сравните сумму размеров объектных файлов проекта и исполняемого файла.

Результаты отобразите скриншотами.

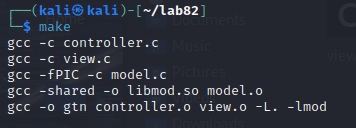
Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом.

Сравните результаты первого и второго упражнения, сделайте вывод.

**Создаем Makefile.**



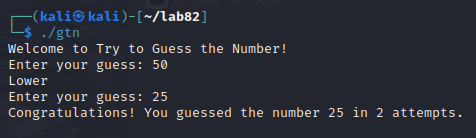
**При помощи утилиты make выполняем автосборку динамической библиотеки и проекта.**



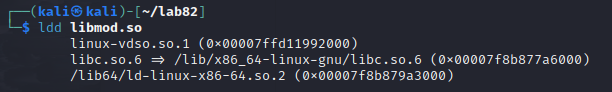
**Устанавливаем переменную LD\_LIBRARY\_PATH и заносим туда абсолютное имя.**



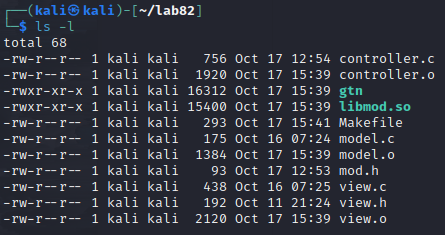
**проверяем работоспособность.**



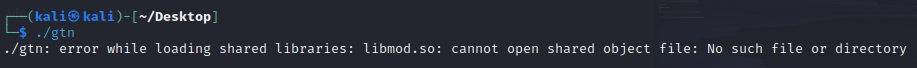
**При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками.**



**Сравните сумму размеров объектных файлов проекта и исполняемого файла.**



**Копию исполняемого файла располагаем на «Рабочем столе», запускаем программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом.**



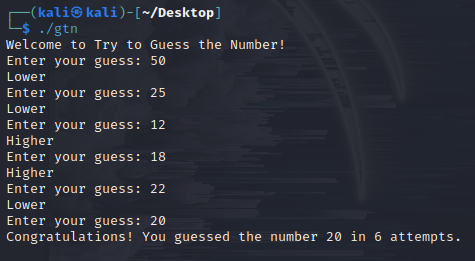
В текущем каталоге не найдено данной библиотеки.

Решаем проблему:



Указываем переменной LD\_LIBRARY\_PATH путь к библиотеки.

запускаем исполняемый файл



**Сравниваем результаты первого и второго упражнения, делаем вывод.**

Процесс создания и подключения совместно используемых библиотек несколько сложнее, чем статических

## 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 библиотеками.

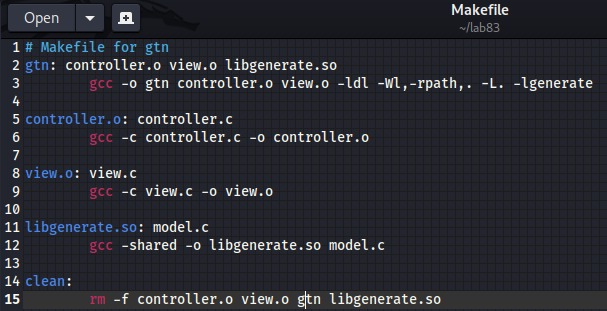
Сравните сумму размеров объектных файлов проекта и исполняемого файла.

Результаты отобразите скриншотами.

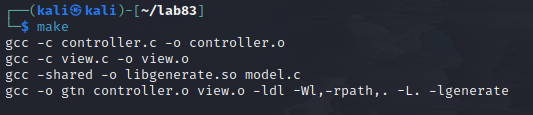
Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом.

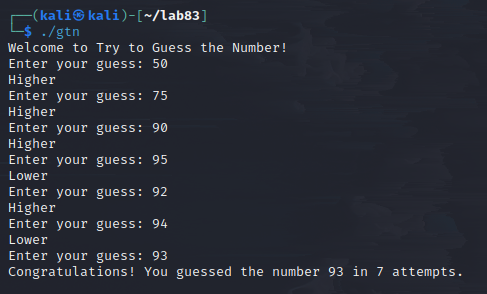
Сравните результаты с предыдущими упражнения, сделайте вывод

**Создаем Makefile.**

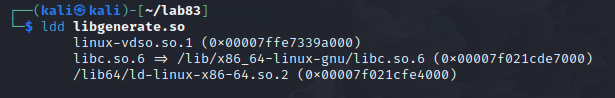


**При помощи утилиты make выполняем автосборку динамической библиотеки и проекта.**

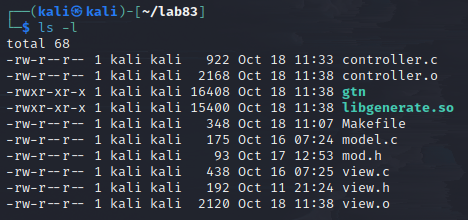




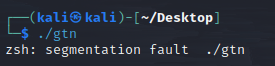
**При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками.**



**Сравните сумму размеров объектных файлов проекта и исполняемого файла.**



**Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом.**



Эта ошибка возникает, когда программа пытается обратиться к памяти, к которой у нее нет доступа.



устанавливаем переменную окружения **LD\_LIBRARY\_PATH** в соответствующий каталог, где находится библиотека.

**Сравните результаты с предыдущими упражнения, сделайте вывод**

Поскольку весь код в статических библиотеках включается в исполняемый файл, это уменьшает вероятность конфликтов между версиями библиотек и повреждения системных библиотек.

Исполняемые файлы, созданные с использованием статических библиотек, могут быть крупными из-за включенного кода, что занимает дополнительное дисковое пространство.

Если динамическая библиотека требует обновления, достаточно заменить её на более новую версию без перекомпиляции программы.

1) **Что такое библиотеки и для чего они используются?**

Библиотеки позволяют разным программам использовать один и тот же объектный код. Библиотека (library) – это набор соединенных определенным образом объектных файлов. Библиотеки подразделяются на две категории: − статические (архивы); − динамические (совместно используемые)

2) **Чем отличается заголовочные файлы (файлы с расширением \*.h) от файлов библиотек?**

- Заголовочные файлы содержат объявления функций, структур, классов и переменных, которые могут быть использованы в других частях программы. Они обычно содержат только сигнатуры (прототипы) функций и объявления данных без их фактической реализации.

- Файлы библиотек, как статические (.a) или динамические (.so или .dll), содержат скомпилированный исполняемый код, который может быть использован другими программами. Они включают в себя реализации функций и данные, которые экспортируются для использования в других программах.