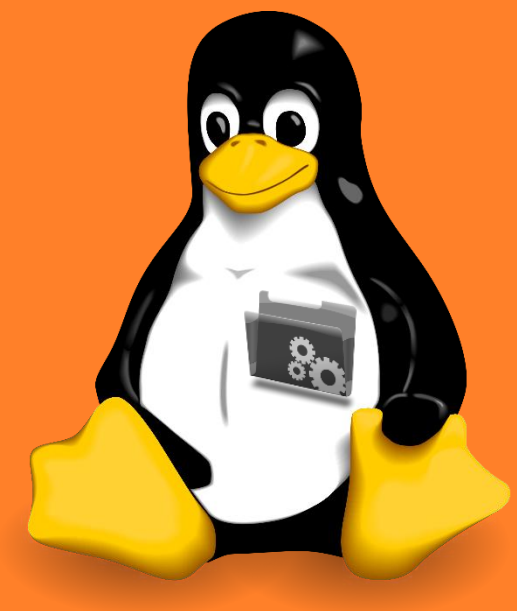
Лабораторная работа #6

Создание и использование библиотек в Linux

[static library (.a), dynamic library (*.so)]*



by Viktor Ivanchenko

ЛАБОРАТОРНАЯ РАБОТА # 6
Создание и использование библиотек в Linux

Цель работы

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

Требования

1) Разработать многофайловый консольный проект на C/C++ согласно варианту задания с использованием шаблона (паттерна) проектирования MVC.
2) Входные данные для программы должны вводиться в виде набора строк! Если в задании сказано про «*слова*» – это значит, что вводится должна одна строка, которая состоит из несколько слов. Если сказано про «*строки*», то должен вводиться набор строк, каждая из которых состоит минимум из двух трёх слов.
3) Предусмотреть два способа инициализации данных: с помощью аргументов командной строки и пользовательского ввода (при запуске программа должна проверять наличие параметров и использовать их для выполнения своего алгоритма при их наличии, в противном случае, запрашивать соответствующие данные у пользователя).
4) Программа должна выводить на консоль исходные данные и конечный результат.
5) ЗАПРЕЩАЕТСЯ в программе использовать под любым предлогом ГЛОБАЛЬНЫЕ переменные!
6) Для повышения производительности программы и закрепления навыков работы с памятью везде, где это возможно, необходимо использовать ДИНАМИЧЕСКОЕ выделение и освобождение памяти, а также осуществлять работу через УКАЗАТЕЛИ.

7) Каждое задание оформить в виде отдельной бизнес-функции. Все функции должны быть сгруппированы по соответствующим отдельным файлам и вынесены в отдельную библиотеку.
8) Предусмотреть создание двух типов библиотек: статической и динамической.
9) Все функции должны быть самодостаточные, т.е. при их разработке необходимо придерживаться принципа ***Single Responsibility Principle***.
10) При выполнении задания разрешается использовать *IDE*, а также задействовать любой текстовый редактор (к примеру, ***gedit***) и набор компиляторов GNU Compiler Collection (GCC), в частности, компиляторы языков программирования C/C++ ***gcc/g++***, а также утилиту для создания файлов-архивов ***ar***.
11) Для автоматизации сборки проекта необходимо использовать утилиту-автосборщик – ***GNU make***.
12) При разработке программ придерживайтесь соглашений по написанию кода на C/C++ (Code-Convention).
13) Контрольные вопросы по лабораторной работе и ответы на них должны быть записаны в конспект.

Основное задание

Для проверки остаточных знаний учеников после рождественских каникул, учитель младших классов решил начинать каждый урок с того, чтобы задавать каждому ученику пример из таблицы умножения, но в классе, к примеру, 15 человек, а примеры среди них не должны повторяться. В помощь учителю разработайте универсальную библиотечную функцию, которая будет выводить на экран 15 (или N) случайных примеров из таблицы умножения (к примеру, от 2*2 до 9*9, потому что задания по умножению на 1 и на 10 – слишком просты). При этом среди 15 примеров не должно быть повторяющихся (примеры 2*3 и 3*2 и им подобные пары считать повторяющимися).

Алгоритм поиска данных комбинаций должен быть эффективным!!!

Напишите тестовую программу, которая бы демонстрировала работоспособность данной библиотечной функции.

Индивидуальное задание

Произвести рефакторинг предыдущей лабораторной работы: вынесите код функций бизнес логики в отдельную библиотеку.

Best of LUCK with it, and remember to HAVE FUN while you're learning :)

Victor Ivanchenko

Что нужно знать (краткие тезисы)

1. Любая библиотека (*library*) в Linux – это набор объектных фалов, сгруппированные таким образом, чтобы ими могли многократно пользоваться разные программы.
2. Библиотеки подключаются к программе на стадии компоновки (сборки или линковки). Для подключения библиотеки в процессе линковки необходимо указать линковщику параметр ***–l*** (от слова *library*) и имя подключаемой библиотеки (между параметром и именем библиотеки имя можно не ставить).
3. Если необходимо указать путь к подключаемой библиотеки, то используют следующую опцию компоновщика –L, после которой нужно указать целевой каталог. Стандартные библиотеки расположены в специальных каталогах (/usr/local/lib, /usr/lib, /lib и т.д.).
4. Обычно имена файлов-библиотек начинаются с префикса lib, который при подключении отбрасывается.
5. Пример подключения библиотеки libsuper, которая расположена в каталоге /usr/lib/super, к программе lab6 с использование gcc-компилятора:

   gcc -o *lab6 lab6.o* ***-L****/usr/lib/super* ***-l****super*

6. В Linux подобных ОС все библиотеки делятся на две категории: статические и динамические.
7. Статическая библиотека (static library) представляет собой архив объектного кода, полностью внедряется в исполняемую программу и создаётся программой-архиваторов ***ar*** из пакета GNU binutils. Файлы статической библиотеки имеют расширение ****.a***.
8. Чтобы создать статическую библиотеку с именем *libsuper.a* из объектных фалов (к примеру, из *super1.o* и *super2.o*) нужно выполнить следующую команду:

   ar -r *libsuper.a super1.o super2.o*

9. Для подключения данной библиотеки к программе *lab6* из текущего каталога необходимо выполнить следующую команду:

   gcc -o *lab6 lab6.o* ***–L****. -l****super*

10. Динамические (или совместно используемые – dynamic/shared library) библиотеки создаются компоновщиком при вызове gcc с опцией ***-shared*** и имеют расширение ****.so***. В процессе линковки динамические библиотеки не помещают свой код непосредственно в программу, а лишь создают в ней ссылки на себя.
11. Динамические библиотеки создаются при помощи gcc по следующей схеме:

gcc -shared -o *name_of_library file1.o file2.o …*

12. При создании динамических библиотек необходимо учитывать следующее:
    a. для компоновки библиотеки должны участвовать только объектные файлы, которые содержат позиционно-независимый код (для чего данные фалы должны компилироваться с опцией компилятора ***–fPIC***);
    b. программист должен учитывать местоположение подключаемой библиотеки, для чего можно установить переменную ***LD_LIBRARY_PATH***.
13. Чтобы создать динамическую библиотеку с именем *libsuper.so* из объектных фалов (к примеру, из *super1.o* и *super2.o, которые были созданы с помощью компилятора gcc с опцией -fPIC*) нужно выполнить следующую команду:

gcc -shared -o *libsuper.so super1.o super2.o*

14. Для подключения данной библиотеки к программе *lab6* из текущего каталога необходимо выполнить следующие команды:

export LD_LIBRARY_PATH=.

gcc -o *lab6 lab6.o* ***–L****. -l**super*

15. Если в целевом каталоге есть одновременно два типа одной и той же библиотеки, то по умолчанию используется динамическая. Если необходимо подключить статическую библиотеку, то нужно явно это указать компоновщику с использованием его опции ***-static***.

Контрольные вопросы

1. Что такое библиотеки и для чего они используются?
2. Опишите существующие типы библиотек и принципы их использования.
3. Опишите преимущества и недостатки каждого из типа библиотек.
4. Как подключить библиотеку к программе? Опишите специфику подключения.
5. На какой стадии происходит подключение библиотек (объектных файлов) к основному модулю программы?
6. Чем отличается заголовочные файлы (файлы с расширением *.h) от файлов библиотек?
7. Как создать статическую (динамическую) библиотеку и подключить её к программе?
8. Что такое «позиционно-независимый код» (position independent code, PIC)?
9. Какие способы можно использовать для того, чтобы указать местоположение подключаемых динамических библиотек?
10. Если в целевом каталоге присутствует два типа одной и той же библиотеки, какая из низ всегда будет подключаться по умолчанию? А как явно подключить вторую?