**ЗАДАНИЕ**

3. Написать динамическую библиотеку с функцией hello\_from\_dynamic\_lib() и использовать ее с hello.c:

hello.c

*#include <stdio.h>*

*void hello\_from\_dynamic\_lib();*

*int main() {*

*hello\_from\_dynamic\_lib();*

*return 0;*

*}*

hello\_dynamic.c

*#include <stdio.h>*

*void hello\_from\_dynamic\_lib(void) {*

*printf("Hello, World! (dynamic lib)\n");*

*}*

• Динамическая библиотека (shared library) — это библиотека, которая загружается в память во время выполнения программы, а не встраивается в исполняемый файл во время компиляции.

В отличие от статической библиотеки, динамическая библиотека не копируется в исполняемый файл, а остаётся отдельным файлом, который программа использует при запуске.

• Как работает динамическая библиотека?

Компиляция программы с подключением .so/.dylib/.dll, но без встраивания кода в исполняемый файл.

Запуск программы: ОС загружает библиотеку в память, когда она требуется.

Использование кода: Программа вызывает функции из библиотеки, как если бы они были встроены.

Если библиотеки нет в системе → программа не запустится, потому что нужный код отсутствует.

*gcc hello\_dynamic.c -c -fpic -o hello\_dynamic.o -Wall*

• gcc hello\_dynamic.c - запускает компилятор GCC, который компилирует исходный файл hello\_dynamic.c.

• -c (Compile Only) - только компиляция, без линковки.

Обычно GCC компилирует и линкует за один шаг, создавая исполняемый файл.

Флаг -c говорит компилятору сгенерировать объектный файл (.o), но не выполнять линковку. После выполнения команды получается файл hello\_dynamic.o, содержащий машинный код функции hello\_from\_dynamic\_lib().

• -fpic (Position-Independent Code, PIC) - этот флаг делает код позиционно-независимым, что важно для динамических библиотек.

• Что это значит?

Код не привязан к фиксированному адресу в памяти. Позволяет загружать библиотеку в любой адрес памяти без изменений.

Улучшает совместимость и производительность.

• Почему это нужно?

Динамическая библиотека может загружаться в разные программы. У каждой программы своё распределение памяти.

Без -fpic библиотека могла бы работать только в одном месте памяти, что неудобно.

• -o hello\_dynamic.o (Output File) - задаёт имя выходного файла — hello\_dynamic.o.

*gcc hello\_dynamic.o -o libhello\_dynamic.so -shared -Wall*

• gcc hello\_dynamic.o - компилятор GCC получает объектный файл hello\_dynamic.o, который мы скомпилировали ранее.

Важно: На этом этапе не происходит компиляция исходного кода, а только создание динамической библиотеки из уже скомпилированного объектного файла.

• -o libhello\_dynamic.so (Output File) - флаг -o указывает имя выходного файла.

Мы создаём файл libhello\_dynamic.so, который будет динамической библиотекой. Расширение .so означает "shared object" — это динамическая библиотека в Linux/macOS.

• -shared (Create Shared Library) - этот флаг говорит GCC создать динамическую библиотеку (.so) вместо обычного исполняемого файла.

• Что делает -shared?

Сообщает компилятору не линкуйте стандартную библиотеку C (libc) напрямую.

Генерирует заголовки ELF (Executable and Linkable Format) для разрешения символов во время выполнения.

Динамический загрузчик (ld.so) сможет использовать этот файл во время запуска программ.

Без этого флага GCC попытался бы создать обычный исполняемый файл, а не библиотеку.

*gcc hello.c -o hello -lhello\_dynamic -L. -Wall*

• gcc hello.c - компилятор GCC получает файл hello.c и компилирует его в объектный код.

• -o hello (Output File) - указывает имя выходного файла. В результате создаётся исполняемый файл hello.

• -lhello\_dynamic (Link with Library) - флаг -l (-l<имя\_библиотеки>) говорит компилятору: "Подключи библиотеку с указанным именем".

Важно:

GCC ищет библиотеку по имени libhello\_dynamic.so, потому что он автоматически добавляет префикс lib и суффикс .so.

Если файл называется libhello\_dynamic.so, то -lhello\_dynamic найдёт его.

Это означает, что мы подключаем динамическую библиотеку libhello\_dynamic.so.

• -L. (Library Path: Current Directory) - этот флаг указывает путь к библиотекам.

-L. означает: Ищи libhello\_dynamic.so в текущей директории (.).

*DYLD\_LIBRARY\_PATH=./ ./hello*

• DYLD\_LIBRARY\_PATH указывает на пути поиска динамических библиотек.

Когда программа запускается, динамический загрузчик ищет динамические библиотеки, которые ей нужны.

По умолчанию этот загрузчик ищет библиотеки в стандартных системных папках (например, /lib, /usr/lib), но можно указать дополнительные пути с помощью переменной DYLD\_LIBRARY\_PATH.

• В данном случае:

DYLD\_LIBRARY\_PATH=./ указывает, что поиск библиотек должен начинаться в текущей директории (./).

Текущая директория — это место, где находится исполняемый файл hello и где, вероятно, находится динамическая библиотека (libhello\_dynamic.so).

• ./hello - это запуск исполняемого файла hello из текущей директории.

./ указывает, что файл нужно искать в текущей директории.

Запускается программа, и она использует библиотеки, указанные в DYLD\_LIBRARY\_PATH.

• Зачем это нужно?

Когда вы работаете с динамическими библиотеками, компоновщик использует системные директории по умолчанию для поиска библиотек. Но если библиотека не в стандартных местах, нужно явно указать путь. С помощью DYLD\_LIBRARY\_PATH можно указать дополнительные каталоги для поиска библиотек, например, текущую директорию (где находится libhello\_dynamic.so).

*export DYLD\_LIBRARY\_PATH=./*

• export — это команда в оболочке (например, в bash или zsh), которая используется для установки переменной окружения и делает её доступной для всех процессов, запущенных в текущей сессии терминала.

• DYLD\_LIBRARY\_PATH — это переменная окружения в macOS, которая указывает пути поиска для динамических библиотек.

• Когда вы пишете команду: DYLD\_LIBRARY\_PATH=./ ./hello

Вы указываете путь к библиотеке только для этого одного запуска программы. То есть это работает однократно.

Однако, если вы хотите избежать указания DYLD\_LIBRARY\_PATH каждый раз, вы можете использовать export: export DYLD\_LIBRARY\_PATH=./

Это установит переменную окружения для всей сессии терминала. После этого, вам не нужно будет указывать DYLD\_LIBRARY\_PATH каждый раз перед запуском программы. Вы просто можете выполнить: ./hello

*./hello*

*Hello, World! (dynamic lib)*



a. посмотреть состояние функции hello\_from\_dynamic\_lib в получившимся исполняемом файле. Объяснить увиденное.

*nm ./hello*

*0000000100000000 T \_\_mh\_execute\_header*

*U \_hello\_from\_dynamic\_lib*

*0000000100003f70 T \_main*

• Команда nm показывает информацию о символах в исполняемом файле (или объектном файле).

• 0000000100000000 T \_\_mh\_execute\_header - это заголовок исполняемого файла, который используется для начала исполнения программы в macOS. Он имеет тип T, что означает символ, который определен в коде.

Этот символ не является функцией, а служит для внутренних нужд macOS, чтобы указать на точку входа в программу.

• U \_hello\_from\_dynamic\_lib - U означает неопределенный символ. Это указывает на то, что функция или символ с именем \_hello\_from\_dynamic\_lib не определена в текущем файле, но требуется для выполнения программы.

Префикс U (Unresolved) говорит о том, что эта функция должна быть разрешена во время связывания. В данном случае это означает, что функция hello\_from\_dynamic\_lib должна быть предоставлена внешней библиотекой, т.е. динамической библиотекой, которую вы загружаете во время выполнения.

Это подтверждает, что ваша программа зависит от функции, которая находится в динамической библиотеке libhello\_dynamic.so, но сама она в исполняемом файле не содержится.

• 0000000100003f70 T \_main - T означает, что символ определен в тексте программы (в коде). Это говорит о том, что функция \_main определена в вашем исходном коде программы и будет точкой входа в программу (это стандартная функция main, с которой начинается выполнение программы).

Адрес 0000000100003f70 — это адрес, по которому эта функция будет находиться в памяти при загрузке программы.

• \_hello\_from\_dynamic\_lib — это функция, которая не определена в вашем исполняемом файле, и поэтому отображается как неопределённый символ (U), так как она должна быть загружена из динамической библиотеки.

Ваш исполняемый файл ссылается на эту функцию, но сама она будет предоставлена внешней библиотекой (libhello\_dynamic.so), и именно в этой библиотеке будет находиться её код.

• Почему функция неопределена в исполняемом файле?

Потому что она реализована в динамической библиотеке, а не в основном исходном коде программы. При запуске программы система загрузит библиотеку и свяжет функцию hello\_from\_dynamic\_lib с её реализацией в библиотеке.

*otool -L hello*

*hello:*

*libhello\_dynamic.so (compatibility version 0.0.0, current version 0.0.0)*

*/usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1336.61.1)*

• Команда otool -L hello используется для отображения зависимостей динамических библиотек (то есть, каких библиотек требует ваша программа для работы).

• libhello\_dynamic.so - libhello\_dynamic.so — это динамическая библиотека, которую ваша программа использует для работы. Она была указана в процессе компиляции с флагом -lhello\_dynamic и находилась в текущей директории (-L.). Это ваша динамическая библиотека, которая содержит реализацию функции hello\_from\_dynamic\_lib().

compatibility version 0.0.0, current version 0.0.0 — эти версии говорят о том, что для вашей программы не заданы отдельные версии совместимости и текущая версия этой библиотеки — это 0.0.0. Это может означать, что библиотека была собрана с версией по умолчанию, или же это просто placeholder (заглушка), если библиотека используется в контексте разработки.

• /usr/lib/libSystem.B.dylib -

/usr/lib/libSystem.B.dylib — это системная библиотека, которая используется в macOS для выполнения основных операций в системе (например, работы с файлами, памятью и т. д.). Она является неотъемлемой частью macOS и часто используется большинством программ.

compatibility version 1.0.0, current version 1336.61.1 — эта информация указывает на совместимость и текущую версию библиотеки. В данном случае это системная библиотека, и она имеет очень стабильную версию, которая может изменяться с обновлениями операционной системы.

• hello зависит от двух библиотек:

1. libhello\_dynamic.so — это библиотека, которую вы создали, и она содержит реализацию функции hello\_from\_dynamic\_lib(), которая используется в вашем коде.

2. libSystem.B.dylib — системная библиотека macOS, которая содержит основные функции, требующиеся большинству программ.

Программа будет пытаться загрузить эти библиотеки при запуске. Если libhello\_dynamic.so не будет найдена, то программа не сможет запуститься, так как будет отсутствовать требуемая функция из динамической библиотеки.

