Многофайловые программы в языке Си. Объектные файлы. Линковка. Сборка через Makefile

Многофайловые программы в языке Си
Программы на Си часто разбивают на несколько файлов для удобства разработки и поддержки. Основные типы файлов:

- **Исходные файлы (`*.c`)** – содержат исполняемый код (функции, переменные).
- **Заголовочные файлы (`*.h`)** – содержат объявления (прототипы функций, макросы, структуры), которые подключаются через `#include`.
  Пример структуры:
  program/
  ├── main.c // Главная программа
  ├── utils.c // Вспомогательные функции
  └── utils.h // Объявления для utils.c

Преимущества:

- Упрощение разработки (каждый файл отвечает за свою часть функционала).
- Ускорение компиляции (изменения в одном файле не требуют пересборки всего проекта).
- Повторное использование кода (заголовочные файлы можно подключать в разных проектах).

Объектные файлы (`*.o` или `*.obj`)

- Это промежуточные файлы, создаваемые компилятором (например, `gcc -c file.c` → `file.o`).
- Содержат машинный код, но ещё не готовы к исполнению (требуется линковка).

```
gcc -c main.c -o main.o      # Создаёт main.o
gcc -c utils.c -o utils.o    # Создаёт utils.o
```

Линковка - процесс соединения объектных файлов и библиотек в один исполняемый файл.
Типы линковки:

- Статическая линковка: Библиотека включается в исполняемый файл

- Динамическая линковка: Библиотека загружается при выполнении программы
  Разница между компиляцией и линковкой:
- Компиляция:

1. перевод исходного кода в машинный код
2. создает объектные файлы

- Линковка

1. соединяет объектные файлы и библиотеки в один исполняемый файл
2. разрешает внешние ссылки

Сборка через Makefile
Makefile – это скрипт для утилиты `make`, который автоматизирует процесс сборки программ. Он особенно полезен для многофайловых проектов на Си.

```
CC = gcc
CFLAGS = -Wall -Wextra

# Цель по умолчанию
all: program

# Сборка исполняемого файла
program: main.o utils.o
    $(CC) $(CFLAGS) main.o utils.o -o program

# Компиляция каждого .c файла в .o
main.o: main.c utils.h
    $(CC) $(CFLAGS) -c main.c

utils.o: utils.c utils.h
    $(CC) $(CFLAGS) -c utils.c

# Очистка
clean:
    rm -f *.o program
```

Основные элементы:

- **Переменные** — для хранения настроек компиляции (компилятор, флаги компиляции, линковки, исходные файлы, объектные файлы)
- **Правила** — определяет, как собирается та или иная часть проекта.

- **Команды** — действия, выполняемые для каждого правила