C 프로그래밍에서의 기본 전략은, 자신이 만든 소스 코드 파일을 실행 가능한 기계어 코드를 가진 파일인 실행 파일로 변환시켜 주는 프로그램을 사용하는 것이다. 이 과정은 컴파일링(compiling)과 링킹(linking)이라는 두 단계를 거쳐서 이루어진다. 컴파일러(compiler)는 소스 코드를 중간 단계의 코드로 변환시키며, 링커(linker)는 실행 파일을 만들기 위해 다른 코드(시스템 라이브러리 파일과 중간 단계에서 생성되는 오브젝트 코드)를 결합해 주는 역할을 한다. C는 프로그램의 모듈화가 가능하도록 이와 같은 두 가지 접근법을 사용한다. 컴파일러로 각각의 모듈을 따로 컴파일한 후, 컴파일된 모듈들을 나중에 링커로 합칠 수 있다. 따라서, 전체 프로그램 중에서 특정 모듈만 수정한 겨우, 수정한 모듈을 제외한 나머지 다른 모듈들은 다시 컴파일 할 필요가 없다.

중간 단계 파일의 형태에는 몇 가지가 있다. 가장 널리 쓰이는 방식은 소스 코드를 오브젝트 코드 파일(object code file, 또는 간단히 줄여서 오브젝트 파일)이라 불리는 기계어로 변환하는 것이다.(여기서는 소스 파일이 한 개의 파일로 이루어져쌀고 가정하자). 비록 오브젝트 파일이 기계어 코드를 포함하고 있지만 아직 실행할 준비는 되어 있지 않다. 오브젝트 파일이란 소스 코드를 단순히 기계어 코드로 번역했을 뿐 완전한 프로그램이 아니기 때문이다.

오브젝트 파일이 가지고 있지 않은 첫번째 요소가 스타트 업(start-up) 코드다. 이 코드는 프로그램과 운영체제 사이의 인터페이스를 담당한다. 예를 들어, IBM PC에서는 DOS나 Linux를 운영체제로 사용할 수 있다. 두 경우 모두 똑같은 하드웨어에서 운영되므로 똑 같은 오브젝트 코드가 있다면 두 운영체제에서 똑 같은 작용을 한다. 그러나 Linux와 Dos 운영체제는 프로그램들을 처리하는 방식이 다르기 때문에 각자 독특한 스타트 업 코드가 필요하다.

두 번째로 부족한 코드 요소는 라이브러리 루틴에 대한 코드다. 거의 모든 C 프로그램은 표준 C 라이브러리의 일부분인 루틴들을 사용한다. – 생략 – 실제 코드는 라이브러리(library)라 불리는 다른 파일에 저장되어 있다. 라이브러리 파일은 수많은 함수에 대한 오브젝트 코드를 포함하고 있다. 링커(linker)의 역할은 이러한 세 가지 요소(오브젝트 코드, 각 시스템에 대한 표준 스타트 업 코드 및 라이브러리 코드)를 묶어서 하나의 파일 즉, 실행 파일로 만들어주는 것이다. 링커는 사용하려는 함수에 대한 코드를 라이브러리로부터 추출한다.

간단히 말해서 오브젝트 파일과 실행 파일 모두 기계어 명령어들로 이루어져 있다. 그러나 오브젝트 파일은 사용한 코드를 단순히 기계어로 번역한 것에 불과하지만, 실행 파일은 사용된 라이브러리 루틴에 대한 기계어 코드는 물론 스타트 업 코드도 포함하고 있다.

이제 컴파일러라고 부르는 언어프로그램을 이용해서 컴퓨터프로그램을 만드는 과정을 간단하게 설명하겠습니다. 컴퓨터프로그램을 만든다는 이야기는 실행파일에 해당하는 *.COM 파일과 *.EXE 파일을 만든다는 뜻입니다. 물론 그외의 부속 파일들도 만들어야 하지만 가장 중요한 파일인 실행파일을 만드는 것이 제일 중요합니다. 컴퓨터를 사용하는 분들은 *.COM 파일과 *.EXE 파일을 어떻게 만들고 그것이 어떻게 동작하는지 매우 궁금할 겁니다. 간단하게 언어프로그램을 이용해서 일반적인 프로그램을 짜는 과정을 정리하면 아래와 같습니다.