编译器（Compiler），是一种计算机程序，它会将用某种编程语言写成的源代码（原始语言），转换成另一种编程语言（目标语言）。

它主要的目的是将便于人编写，阅读，维护的高级计算机语言所写作的源代码程序，翻译为计算机能解读、运行的低阶机器语言的程序，也就是可执行文件。编译器将原始程序（Source program）作为输入，翻译产生使用目标语言（Target language）的等价程序。源代码一般为高阶语言（High-level language），如Pascal、C、C++、C# 、Java等，而目标语言则是汇编语言或目标机器的目标代码（Object code），有时也称作机器代码（Machine code）。 一个现代编译器的主要工作流程如下： 源代码（source code）→ 预处理器（preprocessor）→ 编译器（compiler）→ 汇编程序（assembler）→ 目标代码（object code）→ 链接器（Linker）→ 可执行文件（executables）， 最后打包好的文件就可以给电脑去判读运行了。

编译(compilation, compile)

1、利用编译程序从源语言编写的源程序产生目标程序的过程。

2、用编译程序产生目标程序的动作。编译就是把高级语言变成计算机可以识别的2进制语言，计算机只认识1和0，编译程序把人们熟悉的语言换成2进制的。编译程序把一个源程序翻译成目标程序的工作过程分为五个阶段：词法分析、语法分析、语义检查和中间代码生成、代码优化、目标代码生成。主要是进行词法分析和语法分析，又称为源程序分析，分析过程中发现有语法错误，给出提示信息。

编译器的构造与优化是计算机专业的大学课程，[1]课程名称一般为编译原理。通常在课程中包含了如何实现一种教学用程序语言的编译器。一个著名的例子是20世纪70年代，尼克劳斯·维尔特用于讲解编译器的构造时使用的PL/0编译器。尽管它很简单，PL/0编译器介绍了这个领域的几个有影响的概念：

1、逐步求精的程序开发（也是 1971年 Wirth 的论文的标题）[2]

2、使用递归下降解析器

3、使用EBNF指定语言的语法。

4、代码生成器产生便携P-code

5、在自举问题的正式描述中使用T-diagram。

编译器的一种分类方式是按照生成代码所运行的系统平台划分，这个平台称为目标平台。

有一些编译器输出的代码，将运行于与编译器所在相同类型的计算机和操作系统之上，这种编译器叫做本地编译器。输出可以运行于不同的平台之上的编译器，叫做交叉编译器。由于嵌入式系统通常没有软件开发环境，因此，为这类系统开发软件时，通常需要使用交叉编译器。

编译器所输出于虚拟机上运行之代码，编译器和编译器输出的运行平台有可能相同，也有可能不同。因此，对于这类编译器，不去区分它是本地编译器还是交叉编译器。