C和C++相关知识

Unix，Linux，Windows，Java，C++，C#，Python等底层都是靠C实现的

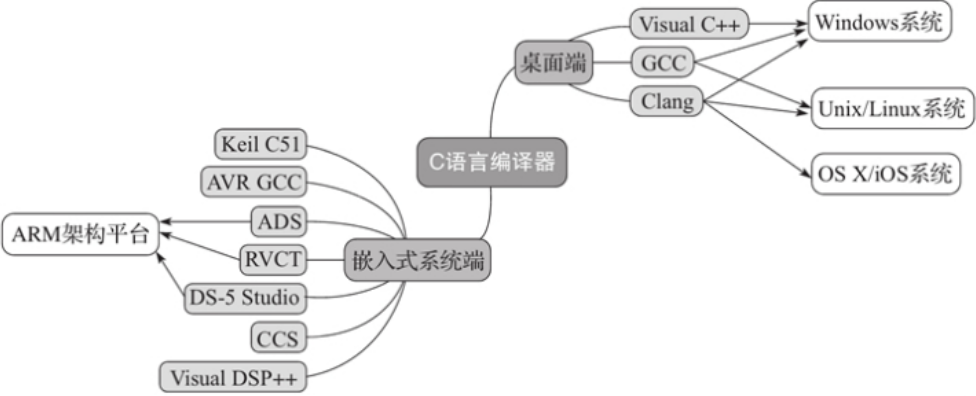
C语言文件后缀名必须以.c 结尾，C++必须以.cpp结尾

编译器

编译器：编译器是一个软件，一个应用程序，它将高级语言文件转化成机器语言的二进制文件

gcc: GNU Compiler Collection，GNU 编译器套件

一、编译器种类



二、编译器的一些基本参数

无论是gcc，clang语法都是都是类似的，比如运行c文件是clang， 运行c++文件时clang++。下面以gcc为例介绍编译流程：

**查看编译器版本**

|  |
| --- |
| **gcc -v // 查看编译器版本** |

**分步骤编译**

|  |  |
| --- | --- |
| gcc 选项 | 含义 |
| -o file | 指定生成的输出文件名为file |
| -E | 只进行预处理 |
| -S | 进行预处理和编译 |
| -c | 进行预处理，编译，汇编 |

|  |
| --- |
| **gcc -E hello.c -o hello.i // 只进行预处理**  **gcc -S hello.i -o hello.s // 只进行编译**  **gcc -c hello.s -o hello.o // 进行汇编**  **gcc hello.o -o hello // 链接库文件**  **gcc hello.c -o hello // 一步到位，其他的步骤还是执行的，只是不生成中间文件了** |

如果命令中不包含输出可执行文件的文件名，可执行文件的文件名会自动生成一个默认名，Linux平台为a.out，Windows平台为a.exe

三、分步编译

一个C文件最终生成二进制可执行文件需要经过**预处理，编译，汇编，链接**的过程。

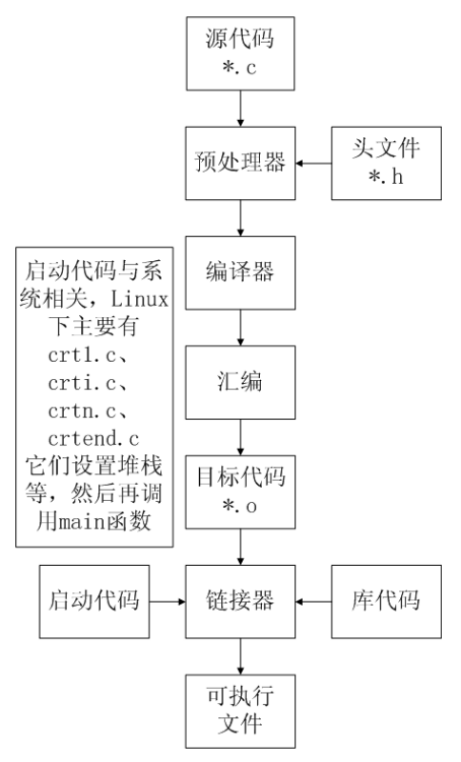
|  |  |
| --- | --- |
| 文件后缀名 | 含义 |
| .c | 正常的C语言文件 |
| .i | 经过预处理后的文件 |
| .s | 编译后的文件，叫汇编文件 |
| .o | 被编译后的文件，叫obj文件，也叫做目标文件 |

预处理：宏定义，头文件展开，条件编译，**删除注释**，**但不检查语法错误**。

编译：**检查语法**，将预处理后文件编译生成汇编文件。

汇编：将汇编文件生成**目标文件(二进制文件)**

链接：C语言的程序是需要依赖各种库的，所以编译之后还需要把库链接到最终的可执行程序中去。



**（一）预处理**

1. 凡是#开头的语句都是预处理指令，在预处理的时候都会被执行。**预处理指令没有分号结束**

* #include 表示头文件引用，它可以引用.h .c文件中的内容。预处理的时候，编译器会把头文件里面所有的东西都完整的拷贝到使用#include 的文件中
* #define 表示宏定义。宏可以理解为等价。在预处理的时候，编译器会把宏都替换成后面的表达式

|  |
| --- |
| **#define Pi 3.14 //**  **printf("%d",Pi);**  **// 预处理后，宏被替换成表达式了**  **printf("%d",3.14);** |

1. 预处理时候，所有的注释都会被删除掉，包括单行注释// 和多行注释/\* \*/
2. 预处理不会检查语病
3. 预处理会处理条件编译，条件编译也是预处理指令

|  |
| --- |
| **#if 0 // 0语言中0表示不成立，非零表示成立**  **printf("123",PI); //这句话不会被编译**  **#endif**  **#if 1**  **printf("123",PI) //这句话会被编译**  **#endif** |

条件编译甚至可以用于检验某个宏存不存在，如果存在则编译，如果不存在就不编译

* 比如系统中如果有abc这个宏，则下面的语句会被编译，否则下面的语句不会被留下，和注释一样被舍弃
* 条件编译可以用来控制系统某个宏对编译的影响

|  |
| --- |
| **int main(){**  **#if abc**  **printf("123",PI)**  **#endif**  **}**  **// gcc -E main.c -o main.i -Dabc 就是加上了这个宏的条件，现在无论如何也可以编译了** |

**（二）编译**

会检查语法错误

**（三）汇编**

将汇编文件转化成.0文件，也叫做obj文件。这个文件是二进制文件

.o文件一般不能被直接执行，因为少了一些运行的必要的链接库

**（四）链接**

设置运行环境，堆栈等基本空间

链接一些必要的库

程序分析

System

system 在一个程序里面执行另外一个外部程序，成功返回0，不成功任意数字

参数：外部程序的路径名。有些程序能在环境中找到，就不用加路径，有些找不到，就要加完整路径。

system是标准库函数，<stdlib.h>

例如：System(pause) 阻塞了程序的执行，只有按任意键可以解除阻塞状态

|  |
| --- |
| system("pause"); // 只有输入任意键后后面的printf才能继续执行  printf("%d",123);  system("hello.exe")// 执行自己的可执行程序  windows路径要加两个反斜杠 C:\\123\\12312\\ |

一些标准化的东西

|  |
| --- |
| # define \_CRI\_SECURE\_WARNINGS // 防止Visual Studio C4996 报错的方法  // Visual Studio 的宏是可以添加到Visual Studio中宏管理的，自行查阅方法  #include <stdio.h>: // 标准输入输出库  #include <string.h>// 字符串处理库 strcpy strcmp strcat  #include <stdlib.h>// 标准库函数 malloc free system  System(“pause”)// 阻塞作用，按任意键才能继续执行 |

System(pause) 阻塞了程序的执行，只有按任意键可以解除阻塞状态

|  |
| --- |
| system("pause"); // 只有输入任意键后后面的printf才能继续执行  printf("%d",123); |

程序分析

|  |
| --- |
| **int main(){**  **printf("hello.h\n");**  **return 0;**  **}** |

1. # 是预处理指令，在预编译的的时候会被执行

* #include <stdio.h> 文件：包含头文件。头文件会包含一些函数和变量的基本定义

1. 两个斜杠 // 单行注释 是可以直接加到语句后面的，/\*\*/多行注释
2. main() 函数：也叫做主函数：整个程序中必须包含一个，并且只能包含一个main函数。程序一定是从main函数开始执行的，到main函数的结束截止

* 一个程序要执行的时候，它先要去找main函数，有了main函数，它才能开始执行，没有main，直接报错
* 不能有多个main，只能由一个main
* int main() 代表main函数结束后的返回值类型。结束后main函数程序返回值会返回到**寄存器**里面。

1. 在C语言对于返回值的要求很松，可以不写void，即使写了写void + return int也不会报错
2. return 结束一个函数，并返回值。返回值的返回类型和定义的类型必须一致
3. 每一条代码必须以分号结尾；