一.C语言的起源

1. 丹尼斯里奇和肯汤普森在开发UNIX操作系统时设计了C语言
2. C语言是在B语言的基础上开发的

二.C语言的优缺点

优点：

- 设计特性：自顶向下规划，结构化编程，模块化设计
- 高效性：具有汇编语言才有的微调控制能力，可以获得最大运行速度或最有效使用内存（空间和时间）
- 可移植性：（但根据编译器不同，同一个C程序在不同OS中执行后的内容可能也不同，如UB）

缺点：

- 危险性高：C语言是静态**弱类型**语言（因为他可能出现forbidden behaviors）
- 写大项目时比较复杂：这是由于C是面向过程编程语言的特性导致的

三.了解计算机

1. CPU:承担绝大部分运算工作。RAM：存储程序和文件的工作区。ROM：永久内存设备。
2. CPU工作过程：从内存中获取并执行一条指令，然后从内存中获取并执行下一条指令。
3. 寄存器：寄存器是中央处理器内的组成部分。寄存器是有限存贮容量的高速存贮部件，它们可用来暂存指令、数据和地址。

四.高级计算机和编译器

1. 编译器是把高级语言程序翻译成计算机能理解的机器语言指令集的程序。
2. 虽然不同CPU制造商使用的指令系统和编码格式不同，但只要使用合适的编译器或编译器集，便可以把一种高级语言程序转换成供各种不同类型CPU使用的机器语言程序。
3. 编译器**自带头文件，**并把它们放在别处，一般用**#include<>**包含；而如果要包含**同一个文件夹下的文件**（一般是自定义），则要用**#include""**包含。

五.C语言标准

1. 第一个标准：K&R C：定义了C语言，但没有定义C库。UNIX实现所提供的库成了标准库

2. 第二个标准：ANSI(American National Standards Institute)和ISO（International Organization for Standardization）的C标准，被称为C89或C90

3. 第三个标准：C99标准，注意并非所有的编译器都完全实现C99的所有改动。

4. 现行标准：C11标准

六.编程机制

1. 过程总结：编译器把源代码转换成**中间代码**，链接器把中间代码和其他代码合并，生成**可执行文件**。

2. 链接器的具体作用：是将你编写的**目标代码**（即中间代码），系统的标准**启动代码**（充当着程序和操作系统之间的接口）和**库代码**（目标代码缺少库函数）

3. 目标文件（**不包含汇编**）和可执行文件的共同点和区别：

- 共同点：都是由机器语言指令所组成的。
- 区别：目标文件中只包含编译器为你编写的代码翻译的机器语言代码。

可执行文件中还包含你编写的程序中使用的库函数和启动代码的机器代码

4. 关于链接器的启动：有些系统中，必须分别运行编译程序和链接程序，而在另一些系统中，编译器会自动启动链接器。

七.编程环境

1. UNIX(包含UNIX的衍生系统，如FreeBSD)：编译：用cc指令。运行：在终端中输入文件名即可。注意其中间文件最后会被删除。

2. GNU和LLVM项目：GNU是一个开发大量免费UNIX软件的集合（全名：GNU is not UNIX）,GCC(即GNU编译器集合，其中包含C的编译器——GCC C)是该项目的产品之一。对应指令gcc（别名：cc）；LLVM则是与编译器相关的开源软件集合，它的Clang处理C代码，可以通过clang调用（别名也是cc）

3. Linux：Linux可以在不同平台上（包括PC和Mac）上运行，与UNIX不同的是他要使用GNU提供的GCC公共域C编译器。

4. Windows:注意Windows并不内置C编译器，因此需要从别处获得，著名的编译器有：Cygwin和MinGW，注意编译器在完成编译后不删除中间文件。

5. Windows/Linux:许多Linux发行版都可以安装在Windows系统中，以创建双系统。不能通过Windows访问Linux文件，但可以通过Linux系统访问Windows文档。

6. Macintosh:这个还是去关注他的Xcode

7. IDE：谁用谁知道