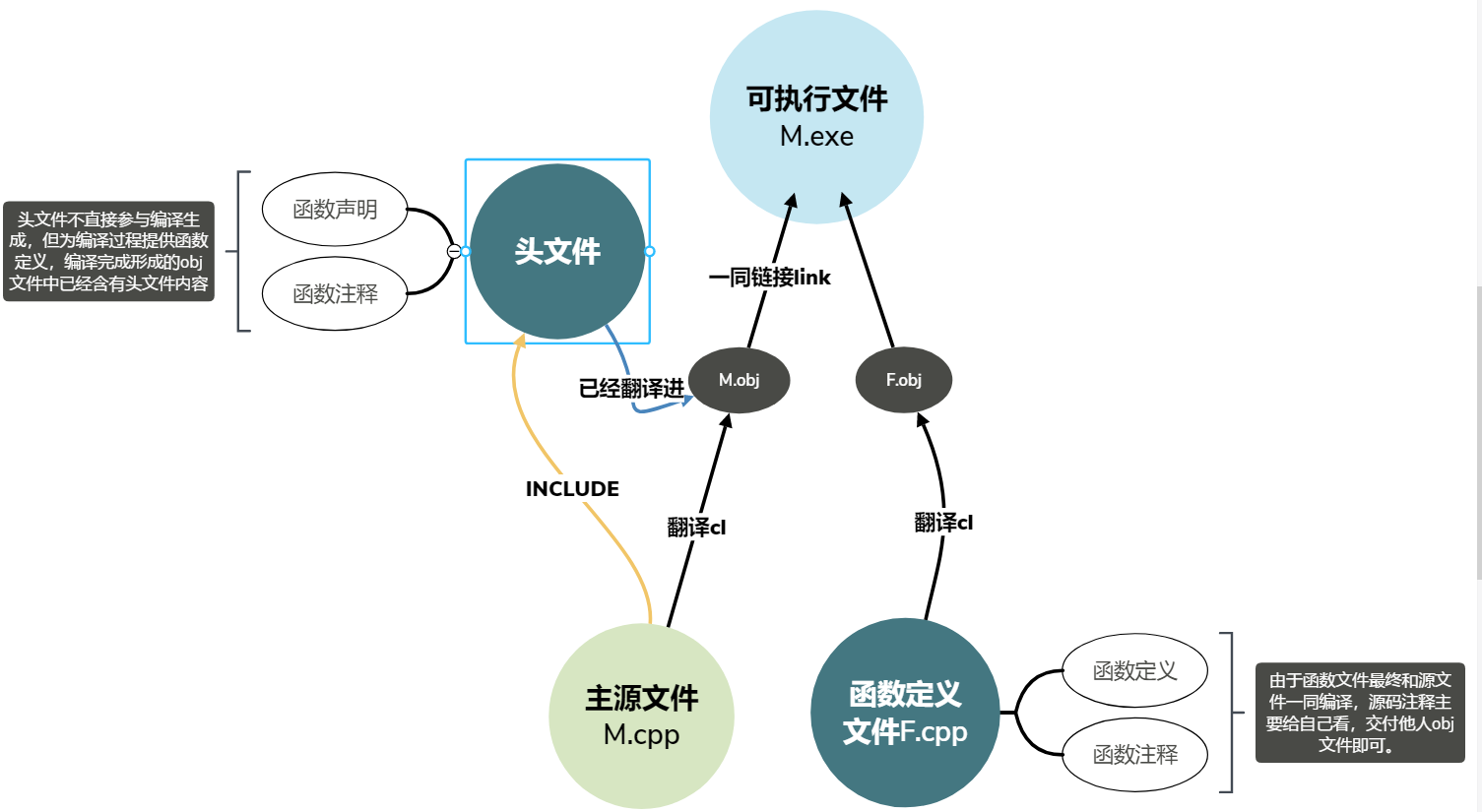
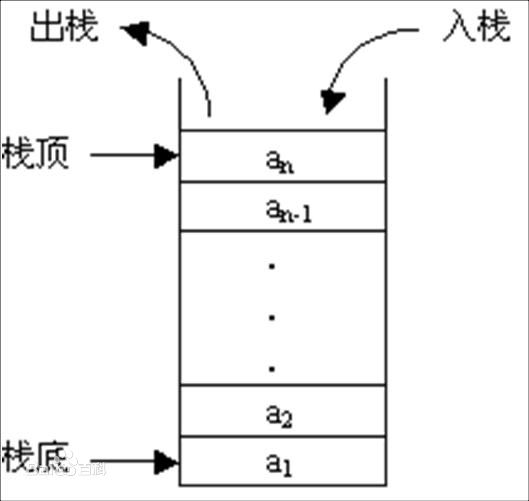
# 课堂笔记

1. 自创函数文件和头文件



1. 栈



栈数据关系：后进先出，编程语言的函数执行步骤的依赖关系用栈模型可以很好表达，所以一般将他们的依赖关系存储在内存的栈空间中。X86平台中栈顶在上，地址空间向下生长，

1. 函数调用原理

当函数调用时：

1. 根据调用约定传递参数

1.1调用约定：包括传参顺序，传参媒介，返回值位置，清理函数空间（参数平衡）

一般的约定：

\_cdecl: 从右往左传参到栈顶，返回值在寄存器，调用方负责清理参数。

\_stdcall: 从右往左传参到栈顶，返回值在寄存器，被调方负责清理参数。

\_fastcall: 左数两个参数通过寄存器传递，其他参数从右往左传参到栈顶， 返回值在寄存器，被调方清理参数。

\_thiscall:以后讲

2．保存返回地址到栈顶

3．转移流程到被调函数地址处

4.记录调用方的栈底部（方便切换）

5.更新栈底到当前位置，作为被调用方的栈底位置上，现在访问栈时是被调方的栈了

6.为局部变量开辟栈空间

6.1：debug模式中抬高的变量空间会大于等于所有局部变量之和，并将局部变量的空间初始化为0Xcccc cccc

6.2:release模式中，抬高的变量空间会小于等于有局部变量之和

7.保存将要修改的寄存器值

8.执行函数

9.从栈顶弹出数据，恢复到寄存器

10.清理局部变量

11．回复调用方的栈底

12．A：\_cdecl约定时候，直接取出栈顶数据作为返回地址，并且流程转移到返回地址处

12.B：其他约定的时候，直接取出栈顶数据作为返回地址，并清理参数空间后流程转移到返回地址处

13.\_ cdecl约定的时候,此时清理参数空间（此时在调用方）