期末笔试时间地点：1-13晚7：00~9：00

笔试闭卷，今年题型有变化（10填空题，10给代码写结果，没定）

往年题意义不大

一教101（70人），201（96人）随便选

不上课也要考试！（及格！）

机考16周！

指针（最难的，不好理解，不好写程序）

程序员最挚爱的武器，功能强大（很多不可能的任务可以完成）

bug大多数来自指针和数组（金矿，生财有道）

灵活，所以麻烦！转了一个弯——寻址方式的变化

指针和指针变量的概念、指针变量作为函数参数、数组指针及指针变量

指针和指针变量口语上不区分，实际应当区分

学指针虽然难，但有更难的生活！

直接寻址：变量名-变量内容；间接寻址：指针变量名-指针变量内容=地址-地址对应内容

指针：内存中某存储空间的地址（编号，一般十六进制）（高级语言中的称呼）

一个变量占多个字节：

小端字节顺序——第一个空间的地址作为变量地址（常用）

大端字节顺序——最后一个空间的地址作为变量地址（不常用）

类型不同，地址意义不同

地址和指针是同义词

指针变量：存放另一个变量的指针（地址）的变量

意义：本来变量分配的地址是动态分配的，事先未知，指针变量提前设好地址

对数据的寻址都是直接寻址

访问：直接访问、间接访问（微机原理有七种访问方式）

间接访问优势：

统一：不同数据类型（存储类型）都可以统一表示指针

灵活：一条语句有不同作用（面向不同寻址目标）

跨界：跨越变量作用范围限定（全局、文件内、函数内、复合语句）

main函数定义的变量a在函数max里作用不了

作用范围不在，但是生命期在——不能通过变量名访问，但是可以间接访问！

作用范围的定义：安全性考虑，但是可以被间接访问打破，出现安全性问题

指针变量两个属性：

指针变量指向的变量类型（汇编：地址）

指向哪一个变量，即指针变量的值（地址值）

也可以指向数组的起始/元素地址，或者函数的地址

函数是一串代码，也要占空间，也有起始地址（函数的地址）

指针的指针：n级指针

定义指针变量：基类型（指向的变量类型）\*变量名（指针变量名）

int \*pointer\_1, \*pointer\_2;：只能保存整型变量的地址

指针变量初始化：定义同时给初值

int \*p = &a; p是指向a的（&：取地址）

基类型确定保存地址的类型

p=&a;把a的地址赋予p

q=p;把p保存的地址给q，使q也指向a

指针变量只能存储地址，不能将非地址类型数据赋给指针变量

解决办法：pointer\_1=(char\*)100，（“告诉房号”）但很危险，可能和其它系统重要代码冲突！

黑客：通过地址找到重要代码进行攻击

指针变量的引用：

&：取地址运算符

\*：指针运算符——定义指针变量、取指针变量对应的指针值对应的空间、还是乘法运算符

绝对找不到混淆三种语义的场合！

定义指针变量未指定指向哪个变量就被用了——访问悬挂指针（野指针）（很容易犯的错误）

没有赋值的变量就是垃圾变量！

永远要清楚每个指针指向哪里！

函数形参使用指针，可以改变未被传递的变量内容——地址结合，双向传递

解决方案窗口看调用函数关系

swap有两种调用方式（pointer\_1/&a）

只有用指针才能实现变量的交换

双向传递需要传递改变的内容，而不能仅仅是指针形参

原理：实参与形参共用存储空间（只是可能发生，不是一定发生）

跨领域访问是主要意义

形参定义写法：（声明）int\*，（定义）int \* 变量名

一维数组指针及其指针变量

定义数组：找一片连续的区域

数组的指针：数组空间起始地址=第一个字节地址值（但是意义不同）

数组名代表其起始地址

数组第一个元素的地址意义与数组的起始地址不同，虽然等值！

可以保存数组元素的指针 &a[i]，或者数组的指针

可以使用指针法代替数组下标法访问数组元素\*(p+1)表示p下一位的空间

数组元素实型，p+1代表p起始的四个字节之后

\*(a+i)表示a数组的第i位的空间（基类型一维数组和元素相同）

scanf后面加&是为了取地址，指针就不管了！

超越作用范围和未定义在调试时无差别

a是指针，&a是二级指针（取a的地址）

p指向a数组，p+i与a+i值同而含义不同，数组的起始地址是常量，不能++

数组访问越界不出语法错误，循环时指针变量指向谁要小心！

避免越界是程序员自己的责任！（C++有报警类）

自学自增自减相关结合性问题！

实参和形参可以混合使用数组名（声明int[]）、指针，形参数组名要额外定义长度

数组名是指针，但不是指针变量

指针可读性差！注释！！！

对程序整体的解释：文档和函数解释说明

有了指针变量名还有用处！

若int\*p，指针p+1时后面的1指的是(int\*)1

可以用char\*p=&a，a为int