函数指针是函数的地址，第一条执行指令的地址（入口地址）

保存地址：指向函数的指针变量

表示方法：函数名代表入口地址，调用函数就是c=p(a,b);（过去是(\*p)）

给调用函数提供第二个途径

定义：int (\*p) (int, int);

指向函数的指针变量和指向数组区别：p++，p--，p+n都无意义（不指向下一语句）

调用函数就要声明，和指针无关

间接调用函数的好处：提高灵活性（函数名赋予同一指针，从而编写通用函数）

程序开发需求的理念，主程序要调用很多函数，编一个统一的框架（与函数名无关）

返回指针的函数

return只返回一个值（或状态，所以不需要返回也返回一下，看看是否正常运行）

返回多个值需要指针，把这些值所在的地址返回（行指针）

声明可以把形参都写上

指针难点：表达形式太多、使用原因

自定义类型数据：结构体、枚举型、共用体（几乎不用了，教材有）

模具-月饼 —— 数据类型-变量

自制“模具”——结构体

需求：一组信息紧密关联

用多个数组保存这些信息——组间相关性被破坏，算法复杂了（如按某类排序）

数据类型不一样，算法差异大！

结构体类型：用户自定义组合数据类型，由若干不同数据类型的有序组合（画表头）

先定义类型，再定义变量

自定义类型：结构体、共用体、枚举型、（C++类）

struct是关键词

类型声明：struct [结构体类型名（标识符）]{ [基本类型] [成员（分量）名];…;}:

不要漏最后一个分号！

改变基本类型名：定义只有一个分量的结构体，或typedef

意义：区分系统函数和自定义函数，兼容不同高级语言习惯

变量定义：struct [标识符] [变量名];

结构体变量（存放结构体数据）是分配空间的，标识符只是类型

先声明后定义

成员可以是另一个结构体类型，即可以嵌套定义

类似于表格！声明——表头；定义——数据行

#define STUDENT struct student可以少写些struct

定义方式：

1（基本方式）：先声明类型，再定义变量（定义时struct可写可不写）；

2：声明同时定义变量struct [类型名]{}[变量名];

3：直接定义变量（一次性，如2省去类型名）

变量名与类型名相同没有关系

存储规则：有对齐问题（空出奇数字节，保证每次取偶数）等等

结构体变量不同分量之间不是连续存储的！

结构体成员偏移量是大小的整数倍，其变量大小是最大成员的整数倍

定义同时赋初值[变量名]={,,,};

但是不能在声明时赋值

一般不能直接使用整体，只能使用分量

格式：结构体变量名.成员名（.：分量运算符）

相同类型结构体赋值、参数传递、取地址（首地址），可以整个结构体/数组使用

结构体嵌套，应一级一级使用

“%x”按照16进制输出

结构体数组：[结构体类型名] [数组名][[长度]]

按元素而不是分量连续存储

初始化是花括号套花括号，或者把内层花括号去掉（如果可以区分）

起始地址用指针变量保存：结构体指针变量

struct [结构体类型] \*p; p=&[结构体变量]

引用成员：指向运算符 p -> [成员名] 或 (\*p).[分量名]

在main函数里声明结构体变量，则其他函数用不了

所以一般把结构体声明写在全局/头文件

字符串赋值用strcpy，不能直接赋值

结构体指针数组，与一般指针数组一样

结构体成员作函数参数：单向传递，值传送

结构体变量作函数参数：单向传递，值传送

结构体指针作函数参数：实参地址传给形参，指向同一存储空间，地址结合，双向传递

+1代表加整个结构体变量的空间

结构：头文件、源文件

结构体声明写在头文件

自定义函数独立按顺序写在前面，最后写main函数

链表（笔试有！）

数据表达、数据计算（编程两件事）

计算：建立、插入、删除、拷贝、排序……（不考）

下棋，记录每一步每个点横纵坐标

点数是动态的，不确定！

给每个点一个空间，互相连续不可能，但是要互相关联：在每一个点里还保存指针

动态内存分配和有序组合的问题得到解决

struct 链表名 {坐标（这一点）;结构体指针[指针域]（指向下一点）};

最后一个结点指针为空指针

头指针指向第一个结点

寻址：头指针，取坐标，再根据指针查找下一个，直到空指针（尾指针）

效率低，引入双链表！

双链表：两头查找，每个结点包含两个指针，需要两个头指针

链表需要动态存储分配malloc（alloc.h/stdlib.h都定义了）

void \* malloc分配地址，void free释放地址（必须在合适时候释放，不然死机！）

calloc是malloc进一步封装

C++使用NEW和delete进行封装

计算：建立、删除、插入结点、排序结点……（数据结构课中学习）

建立链表：定义结构体保存内容和指针，输入为0建立过程结束

首先：表头，新建节点和表尾结点

表头/新节点/表尾指针指向第一个地址（先申请）

第二个结点：malloc创建，新节点和表尾指针指向它

达到最后一个，取尾指针取NULL，free新节点空间，结束

#define LEN sizeof (struct [结构体类型])，表示结构体变量长度

枚举类型

一个变量只有几种可能的值，switch分别每种值情况对应的操作

switch用短整型表示可能值，无语义而且很多空间被浪费

把取值范围已知的变量枚举取值

enum [数据类型] {[枚举元素1],…};

为类型分配空间会小（按照枚举数量确定）

定义：enum [类型] [变量名]; enum可以不写

计算机按照0，1，2，……给枚举元素常量赋值，输出按照整数输出

枚举变量不能赋值，但是定义枚举类型时可以规定被赋的值

枚举值可以作判断

不能把整型赋给枚举变量，但是可以通过强制转换(enum [类型])介导

增强可读性、节省空间！

typedef：定义类型标识符，即已有数据类型别名

typedef [原名] [自定义别名];

专用于某种类型的变量，使程序更清晰；简化数据类型书写（结构体）

16周机考，三道题，20分，1.5h

下次课讲文件

笔试考链表表示，不考链表操作