信手相连

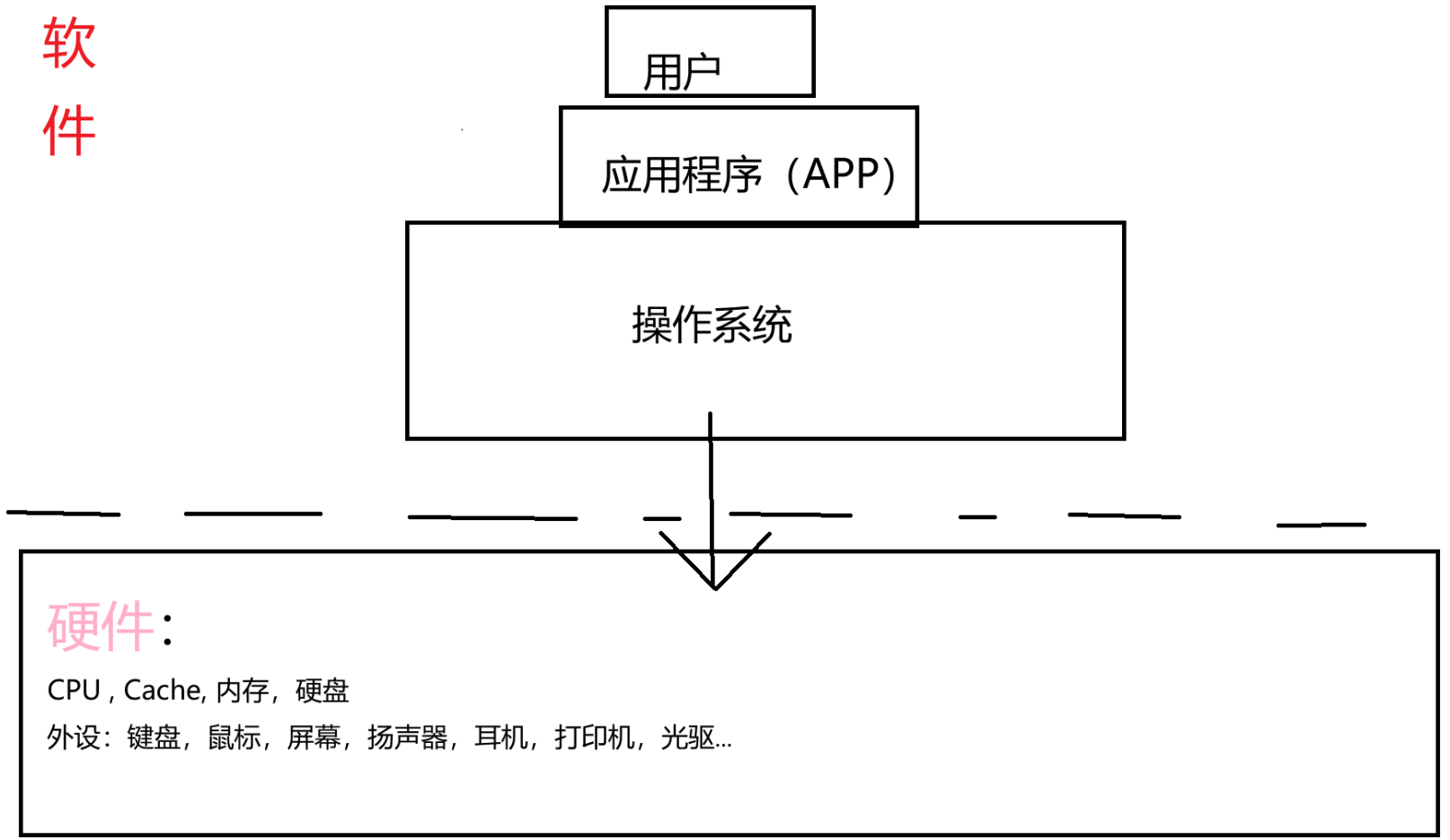
Lecture 2

元雨桐

2024.11.22

一、计算机体系结构

软件

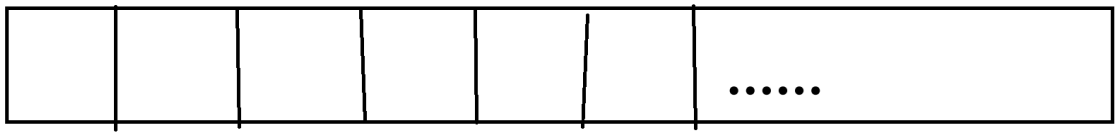


一、计算机体系结构

- 操作系统 是 特殊的程序
- Operating system 非常强大
- Windows, Android, MacOS, Linux

一、计算机体系结构

- 此课程 主要关注 内存
- 内存 可以 想象成 一个 超大 一维数组
- 内存的地址 是 房间号



二、地址

- 数组的每一格 有自己的 下标 —— 内存的每一格有自己的 地址

- 64位的机器（地址是64位的，即 64 bit）
- 1 Byte = 8 bit
- 32位机器：0x00000000 到 0xFFFFFFFF

- 演示：VS2022看内存
  - 内存保护机制：每次运行时，变量地址都是不一样的，
    甚至你看到的地址都只是虚拟地址罢了

三、指针变量

- 指针就是个变量，类似于int, double

  int 存整数，double存小数，指针 存 别人的地址

- 地址是什么？ 64位/32位的2进制数

- 指针：无论你往我里面放什么，我都认为它是个地址（强制类型转换）
  - 指针跟其他变量是一样的，就是用来存东西的，没有什么神奇的

三、指针变量

- 指针变量的大小：8 Byte/4 Byte
- int —— 4 Byte double —— 8 Byte

- 简单提一下：表示范围，原反补码，大小端存储

- 演示：sizeof(char*); x86,x64

三、指针变量

- 指针是一个类型，它有不同 种
  - 类似于float,double，都是小数类型，但根据表示范围的不同，分为了这两种

- 指针区分不同种的标准：存的地址 指向的地方 是 存什么的
  - void*, char*, int*, double*, struct Candidate*, int (*)[], int (*)(int, int)

- 定义： int* p = NULL;
  - p是指针变量，int* 是 p的类型

  演示：sizeof(char*) / sizeof(int*);

  转到定义：#define NULL (void*) 0

四、解引用*

- 访问 指针中 存的 内存地址，并告诉你 这个地址上 存的 内容

五、指针种类的作用

- 指针的种类是一种视角

- int* p;
- *p
- 从地址p开始，往后看4个格，并且把这4格里的东西解码为整数

- char* p;
- *p
- 从地址p开始，往后看1个格，并且把这1格里的东西解码为字符

五、指针种类的作用

- 指针的种类是一种视角

- *(p+1)

五、指针种类的作用

- 指针的种类是一种视角
- 指向数组的某个元素

五、指针种类的作用

- 指针的种类是一种视角
- 指针- 指针 = 两个地址之间的 元素 个数

五、指针种类的作用

- 指针的种类是一种视角

- 用指针迭代，遍历数组（基本不用，容易出错）

六、指针和别的变量类型的杂糅

- 指针的数组 int* p_arr[10]
- 数组的指针 int(*p)[10]
- 指针的指针 char** p
- void*
- 函数的指针
  - bool cmp(int x, int y);
  - bool (*p_func)(int, int) = &cmp;
  - 函数指针的数组

六、指针和别的变量类型的杂糅

- 指针与二维数组
  - 一维数组中，*(arr+2) 等价于 arr[2]
  - 二维数组中，int arr[3][5] 等价于 (int [3])[5]
  - (*(arr+2))+4 等价于 arr[2][4]

七、*指针有效性检查

- 初值
- 越界
- 内存空间已销毁

八、引用&

- 引用 相当于 一个固定的指针
  - 定义时必须给初值，且 一经定义，这个指针指向的地址 就 不许再变了

```
int a = 0;
int& reference = a;
```

- 引用 本质是 指针
  - 所以引用 也可以 像指针一样 有 很多种

八、引用&

- 引用可以用于传参，类似于地址传递
- 在自定义函数内，改变引用参数的值，在函数返回后，此修改依旧生效

九、指针的视角下:vector,map