**C/C++/STL/Python(熟练)**

1. 静态成员

解析：类的静态成员属于整个类 而不是某个对象，可以被类的所有方法访问，子类当然可以父类静态成员；静态成员无多态性！静态方法属于整个类，在对象创建之前就已经分配空间，类的非静态成员要在对象创建后才有内存，所有静态方法只能访问静态成员，不能访问非静态成员；静态成员可以被任一对象修改，修改后的值可以被所有对象共享。

静态成员可以实现多个对象之间的数据共享。

静态成员函数和静态数据成员一样，它们都属于类的静态成员，它们都不是对象成员。因此，对静态成员的引用不需要用对象名。

在静态成员函数的实现中不能直接引用类中说明的非静态成员，可以引用类中说明的静态成员（这点非常重要）。如果静态成员函数中要引用非静态成员时，可通过对象来引用。从中可看出，调用静态成员函数使用如下格式：<类名>::<静态成员函数名>(<参数表>);

1. 多态，虚函数理解（虚表指针）
2. STL迭代器失效
3. 字节对齐

解析：对于**单层结构体**要遵循下面2个原则:

（1）结构体变量中每个成员的偏移量都要是其自身大小的整数倍（第一个成员直接不用看可以,因为其偏移量为0,而任何数的0倍都是0）,不是整数倍要在上一个成员后面补字节.

（2）结构体大小(包含补上的字节大小)必须是所有成员大小(成员自身大小,不包含在其后面补上的字节)的整数倍，也即所有成员大小的公倍数。不是整数倍要在最后一个成员后面补字节

对于**嵌套的结构体**，**需要将其展开**。对结构体求sizeof时，上述两种原则变为：

（1）展开的子结构体中的第一个成员的偏移量应当是父结构体中除子结构体外最大成员大小的整数倍。

（2）结构体大小必须是所有成员大小的整数倍，这里所有成员计算的是展开后的成员，而不是将子结构体当做一个整体。

Sizeof()类，虚函数？？

1. 指针++：

main()

{

    char\*a[]={"work","at","alibaba"};

    char\*\*pa=a;

    pa++;

    printf("%s",\*pa); //at

}

1. C/C++四种类型转换

**数据结构+图像处理+机器学习和深度学习在计算机视觉中的应用（熟练）**

数据结构：

1. 链表：（链表反转，插入排序，归并排序，）
2. 栈：（辅助实现DFS，Get Max Stack, 两个栈实现队列，Sort Stack, Validate parenthesis,二叉树中序遍历，Evaluate Expression,环判断）：
3. 队列：（生产者-消费者,Queue with Max）
4. 树的先，中，后非递归遍历
5. 二叉排序树的查找，插入，删除操作
6. 二叉树节点的下一个节点
7. Dijkstra算法

图像处理：

1. 对比度拉伸，直方图均衡，直方图匹配

2. 仿射变换，透视变换

3. 连通域求解

机器学习：

1.SVM理解

2.LR推导

3.最小二乘法

深度学习：

1. BP算法
2. 各类优化算法

**操作系统+数据库+网络+linux（知道重要的）**

1. 页面置换算法
2. TCP/IP协议

解析：TCP建立连接时，首先客户端和服务器处于close状态。然后客户端发送SYN同步位，此时客户端处于SYN-SEND状态，服务器处于lISTEN状态，当服务器收到SYN以后，向客户端发送同步位SYN和确认码ACK，然后服务器变为SYN-RCVD，客户端收到服务器发来的SYN和ACK后，客户端的状态变成ESTABLISHED(已建立连接)，客户端再向服务器发送ACK确认码，服务器接收到以后也变成ESTABLISHED。然后服务器客户端开始数据传输

1. linux的I/O复用接口select和epoll: http://blog.csdn.net/tennysonsky/article/details/45745887