1. 关系运算符重载
   1. 对于自定义数据类型，编译器不知道如果进行比较
   2. 重载 == ！=号
   3. bool operator==( Person & p)
   4. bool operator!=(Person & p)
2. 函数调用运算符重载
   1. 重载 （）
   2. 使用时候很像函数调用，因此称为仿函数
   3. void operator()(*string* text)
   4. int operator()(int a,int b)
   5. 仿函数写法不固定，比较灵活
   6. *cout* << MyAdd()(1, 1) << *endl*; // 匿名函数对象 特点：当前行执行完立即释放
3. 不要重载 && 和 ||
   1. 原因是无法实现短路特性
   2. 建议：将<< 和 >>写成全局函数，其他可重载的符号写到成员即可
4. 强化训练-字符串类封装
   1. myString类 实现自定义的字符串类
   2. 属性
      1. char \* pString; 维护 在堆区真实开辟的字符数组
      2. int m\_Size; 字符串长度
   3. 行为
      1. 有参构造 MyString(char \* str)
      2. 拷贝构造 MyString(const MyString & str);
      3. 析构 ~MyString();
   4. 重载<< 运算符
   5. 重载 >> 运算符
   6. 重载 = 赋值运算
   7. 重载 [] str[0] 按照索引位置设置获取字符
   8. 重载 + 字符串拼接
   9. 重载 == 对比字符串
5. 继承基本语法
   1. 继承优点：减少重复的代码，提高代码复用性
   2. // 语法： class 子类 ： 继承方式 父类
   3. // News 子类 派生类
   4. // BasePage 父类 基类
6. 继承方式
   1. 公共继承
      1. 父类中公共权限，子类中变为公共权限
      2. 父类中保护权限，子类中变为保护权限
      3. 父类中私有权限，子类访问不到
   2. 保护继承
      1. 父类中公共权限，子类中变为保护权限
      2. 父类中保护权限，子类中变为保护权限
      3. 父类中私有权限，子类访问不到
   3. 私有继承
      1. 父类中公共权限，子类中变为私有权限
      2. 父类中保护权限，子类中变为私有权限
      3. 父类中私有权限，子类访问不到
7. 继承中的对象模型
   1. 父类中的私有属性，子类是继承下去了，只不过由编译器给隐藏了，访问不到
   2. 可以利用开发人员工具查看对象模型
   3. C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\Tools\Shortcuts
   4. 打开开发人员命令工具
   5. 跳转盘符 E:
   6. 跳转文件路径 cd到文件路径下
   7. c**l** /d**1** reportSingleClassLayout类名 文件名
8. 继承中的构造和析构
   1. 先调用父类构造，再调用其他成员构造， 再调用自身构造 ，析构的顺序与构造相反
   2. 利用初始化列表语法 显示调用父类中的其他构造函数
   3. 父类中 构造、析构、拷贝构造 、operator= 是不会被子类继承下去的
9. 继承中的同名成员处理
   1. 我们可以利用作用域 访问父类中的同名成员
   2. 当子类重新定义了父类中的同名成员函数，子类的成员函数会 隐藏掉父类中所有重载版本的同名成员，可以利用作用域显示指定调用
10. 继承中的同名 静态成员处理
    1. 结论和 非静态成员 一致
    2. 只不过调用方式有两种
       1. 通过对象
       2. 通过类名
          1. 通过类名的方式 访问 父类作用域下的m\_A静态成员变量
          2. Son::Base::m\_A
11. 多继承基本语法
    1. class 子类 ： 继承方式 父类1 ， 继承方式 父类2
    2. 当多继承的两个父类中有同名成员，需要加作用域区分
12. 菱形继承
    1. 两个类有公共的父类 和共同的子类 ，发生菱形继承
    2. 菱形继承导致数据有两份，浪费资源
    3. 解决方案：利用虚继承可以解决菱形继承问题
       1. class Sheep : **virtual** public Animal{};
    4. //当发生虚继承后，sheep和tuo类中 继承了一个 vbptr指针 虚基类指针 指向的是一个 虚基类表 vbtable
    5. //虚基类表中记录了 偏移量 ，通过偏移量 可以找到唯一的一个m\_Age
    6. 利用地址偏移找到 vbtable中的偏移量 并且访问数据