String 类的用法：

结构体的基本用法：

共用体的基本概念以及用法

共用体只能存储 int long / double 类型

Union MyUnion{

Int intValue;

Long longValue;

Doubel doubleValue;

}

枚举的概念以及用法：

New 的基础用法

Oop技术在程序运行时分配内存。变量是在编译时分配的，有名称的内存，而指针只是可以通过名称直接访问的内存体用了一个别名。

指正的真正用途是，在运行时分配未命名的内存子存储值，只能通过指针访问。

Deleate 释放内存

Deleate 只能释放使用new分配的内存，

使用deleate的关键在于，将他用于new分配的内存。

C++ 实现通讯录管理功能

程序内存模型：

内存四区：

1. 代码区：存放函数的二进制代码，由操作系统进行管理
2. 全局区：存放全局变量和静态变量以及常量（const 修饰的全局常量）
3. 栈区：由编辑器自动分配释放，存放函数的参数值，局部变量等。
4. 堆区：由程序员分配和释放，若程序员不释放，程序结束时由操作系统回收。

全局区：

该区域的数据在程序结束后由操作系统释放。

引用：

引用一旦初始化后不能改变

引用函数的返回值：

1. 不要返回局部变量的引用。
2. 函数的调用可以作为左值存在。

引用的本质：

引用的本质在C++内部实现是一个指针常量 （地址不可修改， 值可修改）

int\* const ref;

常量引用：

常量引用用来修饰形参，防止形参改变实参

函数提高：

1. 函数的默认参数：

如果函数的声明有默认参数，函数的实现就不能有默认参数；

1. 函数的占位参数：
2. 函数重载

满足条件：

同一个作用域下

函数名称相同

函数参数类型不同或者个数不同或者顺序不同

注意事项：

引用作为重载条件

函数重载碰到默认参数

**类和对象**

* 1. 封装

封装的意义：

将属性和行为作为一个整体，表现生活中的事物

将属性和行为加以权限限制

* 1. 对象的初始化和清理

构造函数的分类和调用：

按参数分：有参构造 无参构造

按类型分： 普通构造 拷贝构造

调用：

1. 括号法
2. 显示法
3. 隐式转换法

注意事项：

调用默认构造函数时，不要使用 （）

类对象作为成员：

当其他类作为本类的成员 构造先构造其他构造，再调用本类构造。 析构函数与之相反。

静态成员变量：

1. 所有对象共享同一份数据
2. 在编译阶段分配内存
3. 类内声明，类外初始化

静态成员函数

1. 所有对象共享同一个函数
2. 静态成员函数只能访问静态成员变量。
3. 两种访问方式： 通过对象访问 ， 通过类名访问

### C++ 对象模型和 this指针

1. 成员变量和成员函数分开存储
   1. 在C++中，类内成员变量和成员函数分开存储
   2. 只有非静态成员变量才属于类的对象上

1. This指针概念

This指针指向被调用的成员函数所属于的对象

This指针是隐含每一个非静态成员函数内的一种指针

This指针不需要定义，可以直接使用

用处：

当形参合成员变量同名时，可以使用this指针来区分

在类的非静态成员函数中返回对象本身，可以 return \*this

1. 空指针访问成员函数
2. 对象特性 const const修饰成员函数
   1. 常函数：

成员函数后加 const 我们称为这个函数为常函数

常函数不可以修改成员属性

成员属性声明时加关键在 mutable后，依然可以在常函数中修改

* 1. 常对象

声明对象前 加 const 称该对象为常对象

常对象只能调用常函数

This指针的本质是指针常量//不可以修改指针的指向，可以修改指向对象的值

等价于 Person\* const this;

1. 友元
2. 全局函数作友元
3. 类作友元
4. 成员函数作友元