目录

[1. 内存分区模型 3](#_Toc128322621)

[代码区 3](#_Toc128322622)

[全局区 3](#_Toc128322623)

[栈区 3](#_Toc128322624)

[堆区 4](#_Toc128322625)

[2.三大特性 5](#_Toc128322626)

[封装 5](#_Toc128322627)

[继承 5](#_Toc128322628)

[多态 5](#_Toc128322629)

1. 内存分区模型

不同区域存放的数据，赋予不同的生命周期给我们更大的灵活编程

在程序编译后，生成exe可执行程序，未执行该程序前分为两个区域，代码区和全局区

代码区

存放函数体的二进制代码，由操作系统进行管理

存放CPU执行的机器指令

代码区是共享的，共享的目的是对于频繁被执行的程序，只需要在内存中有一份代码即可

代码区是只读的，使其只读的原因是防止程序意外的修改了它的指令

全局区

存放全局变量和静态变量（static）以及常量

全局变量和静态变量存放在此

全局区还包含了常量区，字符常量和其它常量也存在此

该区域的数据在程序结束后由操作系统回收

|  |  |
| --- | --- |
| 不在全局区中 | 全局区 |
| 局部变量  const修饰的局部变量（局部常量） | 全局变量  静态变量（static关键字）  字符串常量  const修饰的全局变量（全局常量） |

栈区

由编译器自动分配释放，存放函数的参数值（形参），局部变量等

注意：不要返回局部变量的地址，会由编译器自动释放

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

主要利用new在堆区开辟内存，释放内存使用delete

释放数组的时候，要加 [ ] （delete[] arr;）

|  |  |
| --- | --- |
| 栈区 | 堆区 |
| int \* func() {  int a = 10;  return &a;  } | int \* func() {  int \*p = new int(10);  int \*arr = new int[10];  return p;  }  int main() {  int \*p = func();  delete p;  } |
| 使用函数返回的地址时非法操作 | 返回堆区存放数据的地址  释放后再调用为非法操作 |

2.三大特性

封装

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控制

|  |  |
| --- | --- |
| 都可以表示类 | |
| struct | 默认权限为公共 |
| class | 默认权限为私有 |

|  |  |  |
| --- | --- | --- |
| public | 公共权限 | 类内可以访问，类外可以访问 |
| protected | 保护权限 | 类内可以访问，类外不可以访问，继承时儿子可以访问 |
| private | 私有权限 | 类内可以访问，类外不可以访问，继承时儿子不可以访问 |

**构造函数：**

构造函数，没有返回值也不写void

函数名称与类名相同

构造函数可以有参数，因此可以发生重载

程序在调用对象的时候会自动调用构造，无需手动调用，而且只会调一次

|  |  |  |  |
| --- | --- | --- | --- |
| class Person {  public:  /\* code \*/  } | | | |
| 默认构造函数 | Person() | |  |
| 有参构造函数 | Person(int a) | |  |
| 拷贝构造函数 | Person(const Person &p) | |  |
| 注意：调用时Person p1() 会被解释为p1函数的返回值是Person类型 | | | |
| 括号法调用 | | Person p1(10) | |
| 显示法调用 | | Person p1 = Person(10) | |
| 隐式转换法 | | Person p1 = 10 | |
| 注意：Person(10) 为定义匿名对象，当前行执行结束会被系统回收  Person(p1) 不要利用拷贝构造函数，初始化匿名对象，系统会解释为  Person(p1) === Person p1 即报错p1重定义 | | | |

**--------------------------108--------------------------**

**析构函数：**

析构函数，没有返回值也不写void

函数名称与类名相同，在名称前加符号~

析构函数不可以有参数，因此不可以发生重载

程序在对象销毁前会自动调用析构，无需手动调用，而且只会调一次

继承

多态