析构构造与成员函数

类的访问限制是编译期间的限制

没有显示构造函数，并且没有虚函数、没有继承关系下，默认构造函数在底层其实是没有的

成员函数

在类声明中直接写函数实现，默认函数属性带内联

`__thiscall` 通过 `ecx` 传递对象首地址，其他特性跟 `__stdcall` 一样

- 即使用 `ecx` 传递第一个参数this指针
- 从第二个参数开始使用 `push` 压参数入栈

且在函数内，`mov dword ptr [ebp - xxx], ecx` 将this指针存放在 `[ebp - xxx]` 处

注意：

如果成员函数是 `__cdecl`，this指针会通过压栈成第一个参数，函数外部会平栈

如果成员函数是 `__stdcall`，this指针会通过压栈成第一个参数，函数内部会平栈

如果成员函数是 `__fastcall`，this指针依然通过 `ecx` 传递，第二个参数通过 `edx`，其余通过栈，函数内部会平栈

- 无参的 `__thiscall` 和单参的 `__fastcall` 的识别：判断 `ecx` 是否是类对象的指针

构造析构

对于构造和析构，编译器强制 `__thiscall`

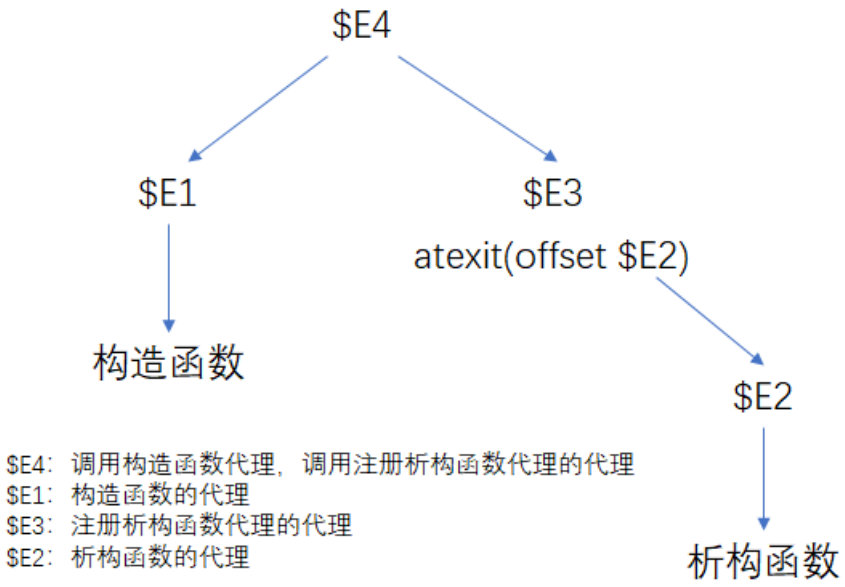
- 构造函数
    1. 构造函数使用 `__thiscall`
    2. 构造函数返回this指针，且通常不会被调用方使用
    3. **构造函数通常是这个对象在对应作用域内第一个调用的成员函数**
- 析构函数
    1. 构造函数使用 `__thiscall`
    2. 构造函数没有返回值定义，并且是无参
    3. **析构函数通常是这个对象在对应作用域内最后一个调用的成员函数**

作用域对构造析构的影响

全局对象

- 在第二个 `__initterm` 中

- `$E4`是构造函数代理的代理，其中包含`$E1`和`$E3`
  - `$E1`是构造函数的代理
  - `$E3`是注册析构函数代理的代理，其中调用`atexit(offset $E2)`来注册析构函数的代理



注意：

`atexit`函数维护了一个函数指针的堆链表，添加的函数和会放在链表的尾部

`main`函数退出后，调用了`exit`，其中又调用了`doexit`，里面从函数指针链表的尾部到头部依次执行

识别：

1. 从`atexit`往上摸
2. 顺着摸
3. 高版本可能把代理函数给内联了，关键点是`__initterm`参数所指示的函数指针的范围

静态局部对象

跟静态局部变量相似，高级版依靠TLS做判定

同样存在**构造的代理**和**析构代理的代理**

堆对象

`new`操作：检查申请的空间是否为空，为空就不执行构造

`delete`操作：检查申请的空间是否为空，为空就不执行析构

- `delete`可能存在参数2，此参数为标志，按位赋予含义
  - 最低位是1，表示释放空间
  - 第二位是1，表示是数组
- 针对于数组
  - 在堆内存中，前面有4字节存放对象个数，
  - `delete[] ptr`先是`ptr - 4`拿到堆空间首地址，然后从计算出堆末尾，从末尾开始往前析构

- 支持异常处理的向量化构造函数迭代器需要5个参数
    1. 对象数组首地址
    2. 对象大小
    3. 数组元素个数
    4. 构造函数指针
    5. 析构函数指针

```
    Test *t = new Test[10];
003C1176 6A 54              push        54h
003C1178 E8 25 02 00 00     call        operator new[] (03C13A2h)
003C117D 83 C4 04           add         esp,4
    Test *t = new Test[10];
003C1180 89 45 F0           mov         dword ptr [ebp-10h],eax
003C1183 C7 45 FC 00 00 00 00 mov       dword ptr [ebp-4],0
003C118A 85 C0              test        eax,eax
003C118C 74 1F              je          main+5Dh (03C11ADh)
003C118E 68 50 10 3C 00     push        offset Test::~Test (03C1050h)
003C1193 68 00 10 3C 00     push        offset Test::Test (03C1000h)
003C1198 6A 0A              push        0Ah
003C119A 8D 58 04           lea         ebx,[eax+4]
003C119D C7 00 0A 00 00 00  mov         dword ptr [eax],0Ah
003C11A3 6A 08              push        8
003C11A5 53                 push        ebx
003C11A6 E8 77 00 00 00     call        `eh vector constructor iterator' (03C1222h)
```

- 支持异常处理的向量化析构函数迭代器需要4个参数
    1. 对象数组首地址
    2. 对象大小
    3. 数组元素个数
    4. 析构函数指针

```
    delete[] t;
003C11C5 85 DB              test        ebx,ebx
003C11C7 74 09              je          main+82h (03C11D2h)
003C11C9 6A 03              push        3
003C11CB 8B CB              mov         ecx,ebx
003C11CD E8 8E FE FF FF     call        Test::`vector deleting destructor' (03C1060h)
```

```
地址(A): Test::`vector deleting destructor'(unsigned int)
查看选项
003C1060 55                 push        ebp    已用时间<=1ms
003C1061 8B EC              mov         ebp,esp
003C1063 6A FF              push        0FFFFFFFFh
003C1065 68 40 21 3C 00     push        3C2140h
003C106A 64 A1 00 00 00 00  mov         eax,dword ptr fs:[00000000h]
003C1070 50                 push        eax
003C1071 53                 push        ebx
003C1072 56                 push        esi
003C1073 57                 push        edi
003C1074 A1 04 40 3C 00     mov         eax,dword ptr [__security_cookie (03C4004h)]
003C1079 33 C5              xor         eax,ebp
003C107B 50                 push        eax
003C107C 8D 45 F4           lea         eax,[ebp-0Ch]
003C107F 64 A3 00 00 00 00  mov         dword ptr fs:[00000000h],eax
003C1085 8B F1              mov         esi,ecx
003C1087 8B 5D 08           mov         ebx,dword ptr [ebp+8]
003C108A F6 C3 02           test        bl,2
003C108D 74 41              je          Test::`vector deleting destructor'+70h (03C10D0h)
003C108F 68 50 10 3C 00     push        offset Test::~Test (03C1050h)
003C1094 FF 76 FC           push        dword ptr [esi-4]
003C1097 8D 7E FC           lea         edi,[esi-4]
003C109A 6A 08              push        8
003C109C 56                 push        esi
003C109D E8 F4 01 00 00     call        `eh vector destructor iterator' (03C1296h)
```

优化

优化后，可能构造的代理会被内联，但一定存在析构的代理（因为`atexit`注册清理函数需要**统一**的接口）