内存管理

内存分布

1. 栈又叫堆栈，非静态局部变量/函数参数、返回值等，栈是向下增长的
2. 内存映射段是高效的IO映射方式，用于装载一个共享的动态内存库。用户可以使用系统接口创建共享内存，做进程间通信
3. 堆用于程序运行时动态内存分配，堆是可以上增长的
4. 数据段存储全局数据和静态数据
5. 代码段可执行的代码/只读常量

```
int globalVar = 1;//数据段
static int staticGlobalVar = 1;//数据段
void Test()
{
    static int staticVar = 1;//数据段
    int localVar = 1;//栈
    int num1[10] = {1, 2, 3, 4};//栈
    char char2[] = "abcd";//栈
    char* pChar3 = "abcd";//代码段
    int* ptr1 = (int*)malloc(sizeof (int)*4);//堆
    int* ptr2 = (int*)calloc(4,sizeof(int));
    int* ptr3 = (int*)realloc(ptr2,sizeof(int)*4);
    free (ptr1);
    free (ptr3);
}
```

C语言动态内存管理方式

malloc/calloc/realloc函数

```
int main()
{
    //malloc申请一块空间
    int* p1 = (int*) malloc(sizeof(int));
    free(p1);
    //calloc申请一块空间，并初始化为0
    int* p2 = (int*)calloc(4, sizeof (int));
    //realloc对已有的空间进行扩容
    int* p3 = (int*)realloc(p2, sizeof(int)*10);
    free(p3);
}
```

C++内存管理方式

new/delete操作符

内置类型

```
int main()
{
    // 动态申请一个int类型的空间
    int* p4 = new int;
    // 动态申请一个int类型的空间并初始化为10
    int* p5 = new int(10);
    // 动态申请10个int类型的空间
    int* p6 = new int[3];
    delete p4;
    delete p5;
    delete[] p6
}
```

- 申请和释放单个元素的空间，使用new和delete操作符，申请和释放连续的空间，使用new[]和delete[]

自定义类型

```
class A
{
public:
    //构造函数
    A()
    {
        _a=0;
    }
    //析构函数
    ~A()
    {

    }
private:
    int _a;
};
int main()
{
    //申请空间+构造函数初始化
    A* p1=new A;
    //释放空间+析构函数
    delete p1;
}
```

- 在申请自定义类型的空间时，new会调用构造函数，delete会调用析构函数，而malloc与free不会。

operator new和operator delete

new和delete是用户进行动态内存申请和释放的操作符，operator new 和operator delete是系统提供的全局函数，new在底层调用operator new全局函数来申请空间，delete在底层通过operator delete全局函数来释放空间。

1. operator new和malloc的区别?
    - 使用方式都一样，处理错误的方式不一样，malloc失败返回0，operator new失败抛出异常(面向对象处理错误的方式)。

new和delete的实现原理

内置类型

如果申请的是内置类型的空间，new和malloc，delete和free基本类似，不同的地方是：new/delete申请和
释放的是单个元素的空间，new[]和delete[]申请的是连续空间，而且new在申请空间失败时会抛异常，malloc会返回NULL

自定义类型

- new
  - 调用operator new函数申请空间，在申请的空间上执行构造函数，完成对象的构造
- delete
  - 调用operator delete函数释放对象的空间，在空间上执行析构函数，完成对象中资源的清理工作
- new []
  - 调用operator new[]函数，在operator new[]中实际调用operator new函数完成N个对象空间的申请,在申请的空间上执行N次构造函数
- delete[]
  - 调用operator delete[]释放空间，实际在operator delete[]中调用operator delete来释放空间，在释放的对象空间上执行N次析构函数，完成N个对象中资源的清理

面试题

malloc/free和new/delete的区别

- malloc/free和new/delete 都是从堆上申请空间，并且需要用户手动释放
- 不同之处：
  1. malloc和free是函数，new/delete是操作符
  2. malloc不能对申请的空间初始化，new可以初始化
  3. malloc申请空间时需要手动计算空间的大小并传递，new只需在后面跟上申请空间的类型
  4. malloc的返回值是void*，使用时需要强转，new不需要
  5. malloc申请失败时返回NULL，使用时必须判空，new申请失败时抛出异常，需要捕获异常
  6. 对于自定义类型，malloc/free只会开辟空间，new/delete会调用构造函数/析构函数完成对象的初始化/空间中资源的清理

内存泄漏

内存泄露指因为疏忽或错误造成程序未能释放已经不再使用的内存。

内存泄漏并不是指内存在物理上的消失，而是应用程序分配某段内存后，因为设计错误，失去了对该段内存的控制，因而造成了内存的浪费。

如何在堆上申请4G内存

将程序编译成X64的进程