田永铭-221900180 作业七

概念题：

1. lameda 表达式在编程中提供了什么便利？

答：对于一些临时用一下的简单函数，如果也要先给出这个函数的定义并为之取个名字，然后再通过这个函数的名字来使用它们，则会给编程带来不便。

Lameda 表达式解决了这个不便。

2. 试从静态成员函数作用的角度解释，为什么 new 操作符和 delete 操作符必须作为静态的成员函数来重载。

答：因为 new 操作符需要在内部调用构造函数来创建新的对象，而 delete 操作符则需要在内部调用析构函数来销毁对象。由于 new 和 delete 操作符必须在对象创建之前和销毁之后进行调用，因此它们无法访问任何对象的成员。而静态成员函数是与类本身关联的，它们不依赖于任何特定的对象实例，因此可以访问类的静态数据成员和静态成员函数，以及其他全局变量和函数。

3. 重载对象数组的创建操作 void *operator new 时，为什么如果类存在析构函 数则实际传入的参数 size 会比 对象数组需要的空间多 4 个字节。试从 void operator delete[] (void *p) 的操作过程这一角度解释。

答：用于存储元素个数！例如，假设类 A 有析构函数，则

- A *p=new A[10]; //size：sizeof(A)*10+4
- ……
- delete []p; //会根据存储的元素个数调用每个元素的析构函数

4. 自定义类型转换操作符可能会造成什么问题？如何解决？

答：会带来歧义问题。

```
class A
{           int x,y;
    public:
        A() { x = 0;   y = 0; }
        A(int i) { x = i; y = 0; }
      A(int i,int j) { x = i;   y = j; }
        operator int() { return x+y; }
    friend A operator +(const A &a1, const A &a2);
};
……
A a;
int i=1,z;
z = a + i;   //是 a 转换成 int 呢，还是 i 转换成 A？
```

编译器通不过。

对上面的问题，可以用显式类型转换来解决：

```
    z = (int)a + i;
```

或

```
    z = a + (A)i;
```

也可以通过给 A 类的构造函数 A(int i)加上一个修饰符 explicit，禁止把它当作隐式类型转换

符来用:

```
class A
{      ......
       explicit A(int i) //禁止把它当作隐式类型转换符来用。
      { x = i; y = 0;
       }
      operator int() { return x+y; }
       ......
};
A a;
int i=1,z;
z = a + i;    //a 转换成 int
```