20220601-元编程

1.学习内容

1.1 元编程

常见的编译期常量

有些东西编译器要求编译期间就确定。除了变量的类型外，最频繁出现的地方就是数组、switch的case标签和模板。

- 数组中的编译期常量

创建一个静态数组时，需要设定一个size。这个size是编译期常量

C++ 复制代码

```
int some[520];
有时候不用显式地指明数组大小，编译器会帮我们自动计算数组的大小
int some[]={5,2,0};
char charArray="Hello";
```

- 模板中的编译期常量

除了type以外，数值也可以作为模板的参数。这些数值变量包括int, long, short, bool, char和弱枚举enum

```cpp
enum color {RED,GREEN,BLUE};
template <unsigned long N,char ID,color c>
struct someStruct{};

someStruct<42ul,'e',GREEN> theStruct;
```

- case labels

switch语句的分支判断必须是编译期常量

```cpp
void comment(int phrase) {
  switch(phrase) {
  case 42:
  std::cout << "You are right!" << std::endl;
  break;
  case BLUE:
  std::cout << "Don't be upset!" << std::endl;
  break;
  case 'z':
  std::cout << "You are the last one!" << std::endl;
  break;
  default:
  std::cout << "This is beyond what I can handle..." << std::endl;
  }
}
```

使用编译期常量的好处

- 更安全的程序

编译期常量能让我们写出更有逻辑的代码——在编译期就体现出逻辑。以矩阵相乘为例。

C++ 复制代码

```cpp
//一般情形下，对于矩阵相乘的判断
class Martix
{
    unsigned rowCount;
    unsigned columnCount;
};

Matrix operator*(Matrix const& lhs,Matrix const& rhs)
{
    if(lhs.getColumnCount()!=rhs.getRowCount())
    {
        throw ohWeHaveAProblem();
    }
}

//如果在编译期就直到了矩阵的size，则可以把上边的判断放在模板中完成。编译器本身就阻止了错误的发生
template <unsigned Rows,unsigned Columns>
class Matrix
{
};
template<unsigned N,unsigned M,unsigned P>
Matrix<N,P> operator*(Matrix<N,M> const& lhs,Matrix<M,P> const& rhs)
{

}

Matrix<1,2> m12=...;
Matrix<2,3> m23=...;
auto m13=m12*m23;//OK
auto mX=m23*m13;//compile error

```

- 编译优化

编译器能根据编译期常量来实现各种不同的优化。比如，如果在一个if判断语句中，其中一个条件是编译期常量，编译器知道在这个判断句中一定会走某一条路，那么编译器就会把这个if语句优化掉，留下只会走的那一条路。在下例中，编译器就会直接利用其中某一个cout语句来替换掉整个if代码块。
另一个可以优化的地方在空间优化。总体来说，如果我们的对象利用编译期常数来存储数值，那么我们就不用在这个对象中再占用内存存储这些数。
》someStruct结构中包含一个‘unsigned long’，一个‘char’，和一个‘color’，尽管如此他的实例对象却只占用一个byte左右的空间。

》矩阵相乘的时候，我们在矩阵中也没必要花费空间去存储矩阵的行数和列数了

C++ 复制代码

```cpp
if (sizeof(void*) == 4) {
  std::cout << "This is a 32-bit system!" << std::endl;
} else {
  std::cout << "This is a 64-bit system!" << std::endl;
}
```

2.结果描述

今天开始详细看元编程的内容。明天继续。