Modern cpp 30 lectures — return object

Monday, December 21, 2020 2:14 PM

接口直接返回对象.

```
#include <armadillo>
#include <iostream>

using arma::imat22;
using std::cout;

int main()
{
  imat22 a{{1,1},{2,2}};
  imat22 b{{1,0},{0,1}};
  imat22 c{{2,2},{1,1}};
  auto r = a * b + c;
  cout << r;
}
```

imat22是元素类型为整数的大小固定为2x2的矩阵.

如何返回一个对象.

一个用来返回的对象,通常应当是可移动构造/赋值的,一般也同时是可拷贝构造/赋值的.如果这样一个对象同时又可以默认构造,我们就称其为一个半正则semiregular的对象.如果可能,应当尽量让我们的类满足半正则这个要求.

Eg.

```
class matrix {
public:
  matrix(size_t rows, size_t cols);      // 普通构造.

  matrix();
  matrix(const matrix&);
  matrix(matrix&&);                      // 半正则要求的构造

  matrix& operator=(const matrix&);
  matrix& operator=(matrix&&);           // 半正则要求的赋值.
};
```

在没有返回值优化的情况下:

```
matrix operator*(const matrix& lhs,
                 const matrix& rhs)
{
   if (lhs.cols() != rhs.rows()) {
      throw runtime_error("sizes mismatch");
   }
   matrix result(lhs.rows(), rhs.cols());

   //具体计算过程.

   return result;
}
```

对于一个本地变量,我们永远不应该返回其引用(或指针),这样的对象一般放在栈上可被调用者正常覆盖使用的部分.

返回非引用类型的表达式,结果是个纯右值(prvalue),在执行 auto r=… 的时候,编译器会认为我们实际是在构造 matrix r(…),而…的部分是一个纯右值,因此编译器会首先试图匹配 matrix(matrix&&),在没有时则匹配 matrix(const matrix&);也就是说,有移动支持时使用移动,没有移动支持时则拷贝.

返回值优化(拷贝消除).

```
#include <iostream>

using namespace std;
                 // can copy and move
class A {
public:
   A() { cout << "Create A\n"; }

   ~A() { cout << "Destroy A\n"; }

   A(const A&) { cout << "Copy A\n"; }

   A(A&&) { cout << "Move A\n"; }
};

A getA_unnamed()
{
  return A();
}

int main()
{
  auto a = getA_unnamed();
}
```

Output:

Create A
Destroy A.

没有Copy A, Move A.

改一下代码:

```
A getA_named()
{
  A a;
  return a;
}

int main()
{
  auto a = getA_named();
}
```

Output:

Create A
Move A
Destroy A
Destroy A

返回内容被移动构造了.

再改一下:

```
#include <stdlib.h>

A getA_duang()
{
  A a1;
  A a2;
  if (rand() > 42) {
     return a1;
  } else {
     return a2;
  }
}

int main()
{
  auto a = getA_duang();
}
```

这回所有编译器都被难倒了, output:

Create A
Create A
Move A
Destroy A
Destroy A
Destroy A.

如果把移动构造函数删除:

```
A(A&&) = delete;
```

Copy A出现在了结果中,说明变成拷贝构造了.