CSDN 学院

C++语言基础

迂者 - 贺利坚

http://blog.csdn.net/sxhelijian/

http://edu.csdn.net

本节主题：

案例：MyVector类的设计

MyVector类要做什么？

- 表示一组相关的数据
  - 数学中的向量：（x, y) , (x, y , z), ...
  - 物理中的矢量：（x, y) , (x, y , z), ...
  - 数据库理论中的元组：（d1, d2, ..., dn）
  - 程序设计语言中的数组：a[5]={...}
- 这组数据上的操作
  - 加
  - 减
  - 数乘
  - 笛卡尔积
  - ...

标准模板库(STL)中的Vector模板类

- 数据成员
  - Member type、value_type、allocator_type、size_type、const_reference、pointer...
- 成员函数及友元函数
  - (constructor)、(destructor)、operator=、assign、get_allocator
  - 元素存取：at、operator[]、front、back
  - 数据容量：empty、size、max_size、reserve、capacity、shrink_to_fit
  - 数据更新：clear、insert、emplace、erase、push_back、emplace_back、pop_back、resize、swap
  - 判断：operator==、operator!=、operator<、operator<=、operator>、operator>=

MyVector类的设计

```
class MyVector  //定义向量类
{
public:
   MyVector(int m);
   MyVector(const MyVector &v);
   ~MyVector();
   friend istream &operator>>(istream &input, MyVector &d);
   friend ostream &operator<<(ostream &output, const MyVector &d);
   friend MyVector operator+(const MyVector &d1,const MyVector &d2);
   friend MyVector operator-(const MyVector &d1,const MyVector &d2);
   bool operator==(const MyVector &d);
   int operator[](int i);
private:
   int *Array;
   int num;
};
```

MyVector类的设计

```
class MyVector  //定义向量类
{
public:
    MyVector(int m);
    MyVector(const MyVector &v);
    ~MyVector();
    friend istream &operator>>(...);
    friend ostream &operator<<(...);
    friend MyVector operator+(...);
    friend MyVector operator-(...);
    bool operator==(...);
    int operator[](int i);
private:
    int *Array;
    int num;
};
```

```
MyVector::MyVector(int m)
{
    num = m;
    Array = new int[num];
    for(int i=0; i<num; ++i)
        Array[i]=0;
}
```

```
MyVector::MyVector(const MyVector &d)
{
    num=d.num;
    if(Array!=NULL) delete []Array;
    Array = new int[num];
    for(int i=0; i<num; ++i)
        Array[i]=d.Array[i];
}
```

```
MyVector::~MyVector()
{
    delete [] Array;
}
```

MyVector类的设计

```
class MyVector  //定义向量类
{
public:
   MyVector(int m);
   MyVector(const MyVector &v);
   ~MyVector();
   friend istream &operator>>(istream &input, MyVector &d);
   friend ostream &operator<<(ostream &output, const MyVector &d);
   friend MyVector operator+(...);
   friend MyVector operator-(...);
   bool operator==(...);
   int operator[](int i);
private:
   int *Array;
   int num;
};
```

```
istream &operator>>(istream &input, MyVector &d){
   cout<<d.num<<"个元素的向量: ";
   for(int i=0; i<d.num; ++i)
      cin>>d.Array[i];
   return input;
}
```

```
int main()
{
   MyVector d1(5),d2(5);
   cin>>d1;
   cin>>d2;
   cout<<"d1="<<d1<<endl;
   cout<<"d2="<<d2<<endl;
   ...;
   return 0;
}
```

```
ostream &operator<<(ostream &output, const MyVector &d){
   cout<<"(";
   if(d.num>0)
      cout<<d.Array[0];
   for(int i=1; i<d.num; ++i)
      cout<<", "<<d.Array[i];
   cout<<")";
   return output;
}
```

MyVector类的设计

```
class MyVector  //定义向量类
{
public:
   MyVector(int m);
   MyVector(const MyVector &v);
   ~MyVector();
   friend istream &operator>>(...);
   friend ostream &operator<<(...);
   friend MyVector operator+(const MyVector &d1,const MyVector &d2);
   friend MyVector operator-(const MyVector &d1,const MyVector &d2);
   bool operator==(...);
   int operator[](int i);
private:
   int *Array;
   int num;
};
```

```
MyVector operator+(const MyVector &d1,const MyVector &d2){
   //未保证大小相同
   MyVector d(d1.num);
   for(int i=0; i<d1.num; ++i)
   {
      d.Array[i]=d1.Array[i]+d2.Array[i];
   }
   return d;
}
```

```
int main()
{
   MyVector d1(5),d2(5);
   ...;
   cout<<"d1+d2="<<d1+d2<<endl;
   cout<<"d1-d2="<<d1-d2<<endl;
   return 0;
}
```

```
MyVector operator-(const MyVector &d1,const MyVector &d2){
   MyVector d(d1.num);
   for(int i=0; i<d1.num; ++i)
   {
      d.Array[i]=d1.Array[i]-d2.Array[i];
   }
   return d;
}
```

MyVector类的设计

```
class MyVector  //定义向量类
{
public:
   MyVector(int m);
   MyVector(const MyVector &v);
   ~MyVector();
   friend istream &operator>>(...);
   friend ostream &operator<<(...);
   friend MyVector operator+(...);
   friend MyVector operator-(...);
   bool operator==(const MyVector &d);
   int operator[](int i);
private:
   int *Array;
   int num;
};
```

```
bool MyVector::operator ==(const MyVector &d)
{
   if(num!=d.num) return false;
   bool eq = true;
   for(int i=0; i<num; ++i)
   {
      if (Array[i]!=d.Array[i])
      {
         eq=false;
         break;
      }
   }
   return eq;
}
```

```
int main()
{
   MyVector d1(5),d2(5);
   ...;
   if (d1==d2) ...;
   cout<<"d1[2]="<<d1[2]<<endl;
   return 0;
}
```

```
int MyVector::operator[](int i)
{
   return Array[i];  //未实施越界保护
}
```

CSDN 学院

THANKS

本课程由 迂者-贺利坚 提供

CSDN网站：www.csdn.net
企业服务：http://ems.csdn.net/
人才服务：http://job.csdn.net/
CTO俱乐部：http://cto.csdn.net/
高校俱乐部：http://student.csdn.net/
程序员杂志：http://programmer.csdn.net/

CODE平台：https://code.csdn.net/
项目外包：http://www.csto.com/
CSDN博客：http://blog.csdn.net/
CSDN论坛：http://bbs.csdn.net/
CSDN下载：http://download.csdn.net/