c++运算符重载总结

class A{

public:

A(int d):data(d){}

A operator+(A&){return A(data+a.data);};//成员函数

A operator-(A&){return A(data-a.data);};

A operator\*(A&){return A(data\*a.data);};

A operator/(A&){return A(data/a.data);};

A operator%(A&){ return A(data%a.data);};

friend A operator+(A&,A&){return A(a1.data+a2.data);};//友元函数

friend A operator-(A&,A&){return A(a1.data-a2.data);};

friend A operator\*(A&,A&){return A(a1.data\*a2.data);};

friend A operator/(A&,A&){return A(a1.data/a2.data);};

friend A operator%(A&,A&){return A(a1.data%a2.data);};

private:

int data;

};

//关系运算符重载

bool operator == (const A& );

bool operator != (const A& );

bool operator < (const A& );

bool operator <= (const A& );

bool operator > (const A& );

bool operator >= (const A& );

//逻辑运算符重载

bool operator || (const A& );

bool operator && (const A& );

bool operator ! ();

//单目运算符重载

A& operator + ();

A& operator - ();

A\* operator & ();

A& operator \* ();

//自增减运算符重载

A& operator ++ ();//前置++

A operator ++ (int);//后置++

A& operator --();//前置--

A operator -- (int);//后置—

//位运算符重载

A operator | (const A& );

A operator & (const A& );

A operator ^ (const A& );

A operator << (int i);

A operator >> (int i);

A operator ~ ();

//赋值运算符重载

A& operator += (const A& );

A& operator -= (const A& );

A& operator \*= (const A& );

A& operator /= (const A& );

A& operator %= (const A& );

A& operator &= (const A& );

A& operator |= (const A& );

A& operator ^= (const A& );

A& operator <<= (int i);

A& operator >>= (int i);

//内存运算符重载

void \*operator new(size\_t size);

void \*operator new(size\_t size, int i);

void \*operator new[](size\_t size);

void operator delete(void\*p);

void operator delete(void\*p, int i, int j);

void operator delete [](void\* p);

//特殊运算符重载

A& operator = (const A& );

char operator [] (int i);//返回值不能作为左值

const char\* operator () ();

T operator -> ();

//类型转换符

operator char\* () const;

operator int ();

operator const char () const;

operator short int () const;

operator long long () const;

friend inline ostream &operator << (ostream&, A&);//输出流

friend inline istream &operator >> (istream&, A&);//输入流