**2016年秋：**计算机高级语言程序设计(C++)答案：

**一、单项选择题**（每题2分，共20分）

**(1) B (2) A (3) B (4) B (5) A**

**(6) C (7) C (8)A (9) C (10) B**

**二、程序阅读题**（每题4分，共40分）

(1) #@ （2）6 4 3

（3） 2,G (4) 3,6

(5) 3 (6) abccba

(7)4 5

(8)6

X operator+(const X &rhs) {

return X(m\_x + rhs.m\_x);

}

程序改进后输出结果 4

(9) 54321，3

(10) A1 A1 B1 A2 B3 A3 B2 A3 A3

三、编程题

class Fraction{

int m\_numerator; // 分子

int m\_denominator; //分母

public:

static int ms\_cnt;

public:

Fraction(int a=0,int b=1);

Fraction(const Fraction&);

void set(int top, int bottom);

friend ostream &operator<<(ostream & out, const Fraction& f);

Fraction & operator=(const Fraction&);

~Fraction();

Fraction operator/(const Fraction&)const;

Fraction &operator/=(const Fraction&);

bool operator<(const Fraction&)const;

void reduction();

};

int Fraction::ms\_cnt = 0;

void Fraction::reduction(){

int r=m\_numerator,s=m\_denominator,max,min;

max=r>s?r:s;

min=r<s?r:s;

while(max%min!=0){

r=max%min;

max=min;

min=r;

}

m\_numerator/=r;

m\_denominator/=r;

}

Fraction::Fraction(int a,int b):m\_numerator(a),m\_denominator(b){}

Fraction::Fraction(const Fraction& rhs):m\_numerator(rhs.m\_numerator),

m\_denominator(rhs.m\_denominator){}

Fraction::~Fraction() {

Fraction::ms\_cnt++;

}

Fraction& Fraction::operator=(const Fraction& rhs){

if(&rhs==this) return \*this; //do not assign itself, e.g. a=a;

m\_numerator=rhs.m\_numerator;

m\_denominator=rhs.m\_denominator;

return \*this;

}

Fraction Fraction::operator/(const Fraction&rhs)const {

return Fraction(m\_numerator\*rhs.m\_denominator, m\_denominator\*rhs.m\_numerator);

}

Fraction &Fraction::operator/=(const Fraction& rhs) {

m\_numerator = m\_numerator\*rhs.m\_denominator ;

m\_denominator = m\_denominator\*rhs.m\_numerator;

return \*this;

}

bool Fraction::operator<(const Fraction&rhs)const{

return m\_numerator\*rhs.m\_denominator / (m\_denominator\*rhs.m\_numerator) < 1;

}

void Fraction::set(int top, int bottom) {

m\_numerator = top;

m\_denominator = bottom;

}

ostream &operator<<( ostream & out, const Fraction& f) {

out<< f.m\_numerator << "/" << f.m\_denominator;

return out;

}

Fraction &max\_element(Fraction \*begin, Fraction \* last) {

Fraction \*max = begin;

for (int i = 1; i < last-begin; ++i) {

if (\*max<begin[i]) {

max = begin + i;

}

}

return \*max;

}

void loadData(Fraction \*f, int size) {

ifstream in("data.txt");

for (int i = 0; i < size; ++i) {

int t, b;

in >> t >> b;

f[i].set(t, b);

}

in.close();

}

void normalize(Fraction \*begin, Fraction \* last) {

Fraction \*max = &max\_element(begin,last);

for (Fraction \*it = begin; it <= last;++it) {

if(it!=max) \*it /= \*max;

}

max->set(1, 1);

}

#define NUM 10

int main(){

{

Fraction arr[NUM];

loadData(arr, NUM);

Fraction &f = max\_element(arr, arr + NUM);

cout << f << endl;

normalize(arr, arr + NUM);

}

cout << Fraction::ms\_cnt << endl;

return 0;

}