CS 124 Test 1 Spring 2008 Thursday, Jan 31

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ CWID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Section: (circle one) 11:00 12:30 2:00

1. What is output by the following program?

#include <iostream>

using namespace std;

class Rational {

int num, den;

public:

Rational() {cout << "default" << endl; num = 0; den = 1; }

Rational(int e) {cout << "top" << endl; num = e; den = 1; }

Rational(int a, int b){cout << "both" << endl; num = a; den = b;}

Rational(const Rational& f) {cout << "copy" << endl;

num = f.num; den = f.den; }

};

int main(){

Rational x;

Rational y = 'a';

Rational w = 3.14159;

Rational g = w;

Rational z('1', 2);

return 0;}

2. Write the definition (outside of the declaration) of the following method for the Rational class:

**IsReduced** – returns *true* if the Rational object is reduced, else returns *false*

3. Write a short main program that illustrates proper and meaningful usage of the **IsReduced** method from above.

4. What are the implications of a class method being private?

5. For class Point write the prototype that would go in the class declaration for a friend function SameQuad that returns true if two Points are in the same quadrant and returns false otherwise.

class Point {

double x, y;

public:

Point() : x(0), y(0) { }

Point(double a, double b) : x(a), y(b) { }

double getX() { return x; }

double getY() { return y; }

void set(double a, double b) { x = a; y = b; }

void print(){cout << "(" << x << "," << y << ")" << endl;}

double dist(const Point& p)const {

double d = sqrt( pow((x-p.x),2) + pow((y-p.y),2) );

return d;

}

};

6. For the friend function from #5 above, write the definition.

7. Write the definition (outside of the class) for a class method that overloads the post decrement operator for class Point.

8. What should the declaration prototype for overloaded input look like for class Point?

9. Write the definition (outside of the class Point declaration) for **operator bool( );** where true is returned if the distance between the origin and the point is greater than 10 and false is returned otherwise.

10. Explain why the following code may work if + is overloaded as a friend but fails to work if + is overloaded as a class method.

**Rational a(1,2), c;**

**c = 2 + a;**

11. Why do we often use & with object parameters?

12. Why do we sometimes prefer to use a friend function instead of a class method?

13. . Explain the purposes of the first and second occurrences of **const** in

**bool compareGenre(const Movie&) const;**

14. Below is the class declaration for class Weight. Tell what should replace the repeated letters in all five cases.

class Weight {

int pounds;

int ounces;

public:

Weight() : pounds(0), ounces(0) { }

Weight(int p, int o) : pounds(p), ounces(o) { }

Weight(int p) : pounds(p), ounces(0) { }

Weight& operator++() {

++ounces;

if (ounces == 16) { ++pounds; AAAAAAAA; }

return \*this;

}

bool operator>(const Weight& w) const {

int t1, t2;

t1 = pounds \* 16 + ounces;

t2 = w.pounds \* 16 + w.ounces;

if (t1 > t2) return true;

return BBBBBBBB;

}

CCCCCCCC ostream& operator<<(ostream& os, const DDDDDDDD);

EEEEEEEE

AAAAAAAA 🡪 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

BBBBBBBB 🡪 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CCCCCCCC 🡪 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DDDDDDDD 🡪 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

EEEEEEEE 🡪 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

15. Writethe definition of method **bool compareDirector(const Movie&, const Movie& )** (from project 1) outside of the class Movie declaration.

16. Illustrate a meaningful usage of compareDirctor (from #15) in a short main program.

17. Describe what you did or should have done in Project 1 to know where in the Cast array you needed to put a new cast member when the method **addCastMember** was used.

18. What is output by the following program?

#include <iostream>

#include <string>

using namespace std;

class Rational {

int num, den;

public:

Rational( ):num(0), den(1){cout << "default" << endl;}

Rational(int n):num(n), den(1){cout << "int" << endl;}

Rational(double d):num(d), den(3){cout << "double/three" << endl;}

friend Rational operator+ ( const Rational& a, const Rational& b );

friend ostream& operator<<(ostream&, const Rational&);

};

Rational operator + (const Rational& a, const Rational& b){

Rational temp;

temp.num=a.num\*b.den+b.num\*a.den;

temp.den=a.den\*b.den;

return temp;}

ostream& operator<<(ostream& os, const Rational& r){

cout << r.num << "/" << r.den << endl; return os;}

int main( ) {

Rational a, b(37);

a = b + 72.56;

cout << a;

return 0;}

19. What would the **typecast to an integer** prototype look like for the class in #18?

20. If class Rational contains both of – a typecast to an integer and a one integer parameter constructor – what problems might arise with the following code?  
 Rational r1(12), r2;  
 int x = 5, y;  
 r2 = r1 + x; // line 1

Y = r1 + x; // line 2