**CS 37 - Exam #4**  Name: Minh Anh Bui

**20 Points Possible**

You are to write some overloaded operator methods for Restaurant Survey (Rest) objects. A Rest class will be used with the following private data members for one restaurant:

integer number of people who say the restaurant is good **good** (eg 200)

integer number of people who say the restaurant is fair **fair** (eg 75)

integer number of people who say the restaurant is bad **bad** (eg 25)

integer number of total people surveyed **total** (good + fair + bad eg 300)

float percentage of good results **pergood** (good / total \* 100 eg 66.66667)

(3) 1. Overload the >= operator in the public section. Example: if (r1 >= r2) Your code should return true if the percentage of good results for r1 is greater than or equal to the percentage of good results for r2.

bool operator >= (const Rest &rhs) const

{

return rhs.good >= good;

}

(3) 2. Overload the + operator. Example: sum = r1 + r2; Your code should sum the good, fair, bad, and total data members of r1 and r2 and place the results into the sum object along with calculating the pergood data member for the sum object.

Rest operator + (const Rest &rhs) const

{

rhs.pergood = rhs.good/rhs.total\*100

return Rest(good + rhs.good, fair + rhs.fair, bad + rhs. Bad, total + rhs.total, pergood + rhs.pergood);

}

(3) 3. Overload the ++ operator (preincrement). Example: ++r1; Your code should add one to good in the object. Do not forget to recalculate the necessary data members.

Student &Student::operator ++()

{

good++;

total = good + fair + bad;

pergood = (good / total \* 100)

return \*this;

}

(3) 4. Overload the ++ operator (postincrement). Example: r1++; Your code should add one to good in the object. Do not forget to recalculate the necessary data members.

Rest Rest::operator ++(int)

{

Rest temp = \*this;

good++;

total = good + fair + bad;

pergood = (good / total \* 100)

return temp;

}

(4) 5. Overload the stream insertion operator. Example: cout <<r1; (No formatting is required.)

ostream &operator<<( ostream &output, const Rest &r)

{

output << r.good << r.fair << r.bad << r.total << r.pergood << endl;

return output;

}

(4) 6. Overload the stream extraction operator. Example: cin >> r1;

istream &operator>>( istream &input, Rest &r)

{

input >> r.good >> r.fair >> r.bad;

r.total = r.good + r.fair + r.bad;

r.pergood = (r.good / r.total \* 100)

return input;

}