**Problem 2:**

#include <iostream>

#include <string>

using namespace std;

class ModInt{

private:

int mod;

int value;

//value should be one less than the mod

public:

//bool operator ==(ModInt lhs, ModInt rhs);

ModInt(){

mod = 1;

value = 0;

}

ModInt(int modInput, int upperInput){

mod = modInput;

value = upperInput;

}

int getMod(){

return mod;

}

int getvalue(){

return value;

}

};

bool operator == (ModInt lhs, ModInt rhs){

if(lhs.getMod() == rhs.getMod() && lhs.getvalue() == rhs.getvalue()){

return true;

}

else{

return false;

}

}

int main(){

ModInt test = ModInt(2000, 1999);

ModInt test2 = ModInt(2000, 1999);

ModInt test3 = ModInt(3000, 1999);

cout << test.getMod() << endl;

cout << test.getvalue() << endl;

if(test==test2){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

if(test==test3){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

return 0;

}

**Problem 3:**

#include <iostream>

#include <string>

using namespace std;

class ModInt{

private:

int mod;

int value;

//value should be one less than the mod

public:

//bool operator ==(ModInt lhs, ModInt rhs);

ModInt(){

mod = 1;

value = 0;

}

ModInt(int modInput, int upperInput){

mod = modInput;

value = upperInput;

}

int getMod(){

return mod;

}

int getvalue(){

return value;

}

void operator += (ModInt rhs){

int tempSum;

if(mod == rhs.getMod()){

tempSum = value + rhs.getvalue();

tempSum = tempSum%mod;

value = tempSum;

}

else{

mod = -1;

value = -1;

}

}

};

bool operator == (ModInt lhs, ModInt rhs){

if(lhs.getMod() == rhs.getMod() && lhs.getvalue() == rhs.getvalue()){

return true;

}

else{

return false;

}

}

int main(){

ModInt test = ModInt(2000, 1999);

ModInt test2 = ModInt(2000, 1999);

ModInt test3 = ModInt(3000, 1999);

/\*cout << test.getMod() << endl;

cout << test.getvalue() << endl;

if(test==test2){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

if(test==test3){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

\*/

test+=test2;

cout << test.getvalue() << endl;

cout << test.getMod() << endl;

test2+=test3;

cout << test2.getvalue() << endl;

cout << test2.getMod() << endl;

return 0;

}

**Problem 3:**

#include <iostream>

#include <string>

using namespace std;

class ModInt{

private:

int mod;

int value;

//value should be one less than the mod

public:

//bool operator ==(ModInt lhs, ModInt rhs);

ModInt(){

mod = 1;

value = 0;

}

ModInt(int modInput, int upperInput){

mod = modInput;

value = upperInput;

}

int getMod(){

return mod;

}

int getvalue(){

return value;

}

void operator += (ModInt rhs){

int tempSum;

if(mod == rhs.getMod()){

tempSum = value + rhs.getvalue();

tempSum = tempSum%mod;

value = tempSum;

}

else{

mod = -1;

value = -1;

}

}

};

bool operator == (ModInt lhs, ModInt rhs){

if(lhs.getMod() == rhs.getMod() && lhs.getvalue() == rhs.getvalue()){

return true;

}

else{

return false;

}

}

ModInt operator + (ModInt lhs, ModInt rhs){

int tempSum;

ModInt toReturn;

if(lhs.getMod() == rhs.getMod()){

tempSum = rhs.getvalue() + lhs.getvalue();

tempSum = tempSum%rhs.getMod();

toReturn = ModInt(lhs.getMod(), tempSum);

}

else{

toReturn = ModInt(-1, -1);

}

return toReturn;

}

ModInt operator \* (ModInt lhs, ModInt rhs){

int tempMult;

ModInt toReturn;

if(lhs.getMod() == rhs.getMod()){

tempMult = rhs.getvalue() \* lhs.getvalue();

tempMult = tempMult%rhs.getMod();

toReturn = ModInt(lhs.getMod(), tempMult);

}

else{

toReturn = ModInt(-1, -1);

}

return toReturn;

}

int main(){

ModInt test = ModInt(2000, 10);

ModInt test2 = ModInt(2000, 20);

ModInt test3 = ModInt(3000, 1999);

/\*cout << test.getMod() << endl;

cout << test.getvalue() << endl;

if(test==test2){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

if(test==test3){

cout << "They are equal" << endl;

}

else{cout << "They are NOT equal" << endl;}

test+=test2;

cout << test.getvalue() << endl;

cout << test.getMod() << endl;

test2+=test3;

cout << test2.getvalue() << endl;

cout << test2.getMod() << endl;

\*/

ModInt multMod = test2\*test;

ModInt addMod = test2+test;

cout << multMod.getvalue() << endl;

cout << addMod.getvalue() << endl;

return 0;

}