CSCI60 Lab 2

Benjamin Xiong

Mike Dong

10/5

1.

```
#include <iostream>
using namespace std;


class ModInt{
private:
        int num;
        int mod;
public:
        ModInt(int a, int b);
        ModInt();
        int getNum(){
                return num;
        }
        int getMod(){
                return mod;
        }
        void plusEq(ModInt rhs);
};
```

```
ModInt :: ModInt(int a, int b){
        num = a;
        mod = b;
}
ModInt :: ModInt(){
        num = 0;
        mod = 1;
}
int main(){
        return 0;
}
```

(all testing codes are at the bottom)

2.

```
bool equals(ModInt lhs, ModInt rhs);
bool equals(ModInt lhs, ModInt rhs){
        if(lhs.getNum() == rhs.getNum() && lhs.getMod() == rhs.getMod()){
                return true;
        }
        else{
                return false;
        }
}
```

3.

It has to be a member function because there should already be a ModInt and that ModInt will use this function in the form of .plusEq(ModInt rhs). As a member function, plusEq can be used that way.

```
void ModInt :: plusEq(ModInt rhs){
        if(getMod() == rhs.getMod()){
                num = (getNum() + rhs.getNum()) % getMod();
        }
        else{
                num = -1;
                mod = -1;
        }
}
```

4.

(Because if the function is named “plus”, it might result in “reference to ‘plus’ is ambiguous” problem, and as we tested, naming it slightly different than “plus” can solve the problem, so our answer function is called “plus1”)

```
ModInt plus1(ModInt lhs, ModInt rhs);
```

(this has to be put under the ModInt class to let the program knows what “ModInt” is)

```
ModInt plus1(ModInt lhs, ModInt rhs){
        lhs.plusEq(rhs);
```

```
    return lhs;
}
```

Testing codes:

```
int main(){
    ModInt a;
    ModInt b(175, 180);
    cout << a.getNum() << endl; //0
    cout << a.getMod() << endl; //1
    cout << b.getNum() << endl; //175
    cout << b.getMod() << endl; //180
    ModInt c(175, 180);
    cout << equals(a, c) << endl; //0
    cout << equals(b, c) << endl; //1
    ModInt d(6, 180);
    c.plusEq(d);
    cout << c.getNum() << endl; //1
    cout << c.getMod() << endl; //180
    ModInt e(175, 180);
    ModInt f = plus1(e, d);
    cout << f.getNum() << endl; //1
    cout << f.getMod() << endl; //180
    return 0;
}
```