Exercises week 3

Klaas Isaac Bijlsma s2394480

David Vroom s2309939

December 6, 2017

Exercise 20

Learn the implications of using friends

We used the following code,

binops/binops.h

```
1  #ifndef EX20_BINOPS_H
2  #define EX20_BINOPS_H
3
4  #include "../addition/addition.h"
5  #include "../subtraction/subtraction.h"
6
7  class Binops: public Addition, public Subtraction
8  {
9      friend class Addition;
10     friend class Subtraction;
11
12     private:
13         void binopsAdd(Operations const &rhs);
14         void binopsSub(Operations const &rhs);
15 };
16
17 #endif
```

addition/addition.h

```
#ifndef EX20_ADDITION_H
#define EX20_ADDITION_H

class Operations;

class Addition
{
    friend Operations operator+(
        Operations const &lhs, Operations const &rhs);   // 1
    friend Operations operator+(
        Operations &&lhs, Operations const &rhs);        // 2

    public:
        Operations &operator+=(Operations const &rhs) &; // 1
        Operations operator+=(Operations const &rhs) &&; // 2
};

#endif
```

subtraction/subtraction.h

```
#ifndef EX20_SUBTRACTION_H
#define EX20_SUBTRACTION_H

class Operations;

class Subtraction
{
    friend Operations operator-(
        Operations const &lhs, Operations const &rhs);   // 1
    friend Operations operator-(
        Operations &&lhs, Operations const &rhs);        // 2

    public:
        Operations &operator-=(Operations const &rhs) &; // 1
        Operations operator-=(Operations const &rhs) &&; // 2
};

```

```
18 #endif
```

Exercise 21

Learn to implement a class hierarchy using friends in the final derived class

We used the following code,

binops/binops.ih

```
1 #include "binops.h"
2 #include "../operations/operations.h"
```

binops/binopsadd.cc

```
1 #include "binops.ih"
2
3 void Binops::binopsAdd(Operations const &rhs)
4 {
5     static_cast<Operations &>(*this).add(rhs);
6 }
```

binops/binopssub.cc

```
1 #include "binops.ih"
2
3 void Binops::binopsSub(Operations const &rhs)
4 {
5     static_cast<Operations &>(*this).sub(rhs);
6 }
```

addition/addition.ih

```
1 #include "addition.h"
2 #include "../operations/operations.h"
3
4 #include <utility>
5
6 using namespace std;
```

addition/operatoraddis1.cc

```
#include "addition.ih"

Operations &Addition::operator+=(Operations const &rhs) &
{
    cout << "operatoraddis1 calls: ";
    static_cast<Binops &>(*this).binopsAdd(rhs);
    return static_cast<Operations &>(*this);
}
```

addition/operatoraddis2.cc

```
#include "addition.ih"

Operations Addition::operator+=(Operations const &rhs) &&
{
    cout << "operatoraddis2 calls: ";
    static_cast<Binops &>(*this).binopsAdd(rhs);
    return move(static_cast<Operations &>(*this));
}
```

subtraction/subtraction.ih

```
#include "subtraction.h"
#include "../operations/operations.h"

#include <utility>

using namespace std;
```

subtraction/operatorsubis1.cc

```
#include "subtraction.ih"

Operations &Subtraction::operator-=(Operations const &rhs) &
{
    cout << "operatorsubis1 calls: ";
    static_cast<Binops &>(*this).binopsSub(rhs);
```

```
7       return static_cast<Operations &>(*this);
8   }
```

subtraction/operatorsubis2.cc

```
1   #include "subtraction.ih"
2
3   Operations Subtraction::operator-=(Operations const &rhs) &&
4   {
5       cout << "operatorsubis2 calls: ";
6       static_cast<Binops &>(*this).binopsSub(rhs);
7       return move(static_cast<Operations &>(*this));
8   }
```

Exercise 22

Learn to use a class hierarchy using friends in the final derived class

We used the following code,

addition/operatoradd1.cc

```
1  #include "addition.ih"
2
3  Operations operator+(Operations const &lhs, Operations const &rhs)
4  {
5      cout << "operatoradd1 calls: ";
6      Operations ret(lhs);
7      ret += rhs;
8      return ret;
9  }
```

addition/operatoradd2.cc

```
1  #include "addition.ih"
2
3  Operations operator+(Operations &&lhs, Operations const &rhs)
4  {
5      cout << "operatoradd2 calls: ";
6      Operations ret(move(lhs));
7      ret += rhs;
8      return ret;
9  }
```

subtraction/operatorsub1.cc

```
1  #include "subtraction.ih"
2
3  Operations operator-(Operations const &lhs, Operations const &rhs)
4  {
5      cout << "operatorsub1 calls: ";
6      Operations ret(lhs);
7      ret -= rhs;
```

```
8       return ret;
9   }
```

subtraction/operatorad2.cc

```
1   #include "subtraction.ih"
2
3   Operations operator-(Operations &&lhs, Operations const &rhs)
4   {
5       cout << "operatorsub2 calls: ";
6       Operations ret(move(lhs));
7       ret -= rhs;
8       return ret;
9   }
```

Exercise 23

Learn to use a class hierarchy using friends in the final derived class

Exercise 24

Learn to initialize ***string*** *objects with* ***new*** We used the following code,

main.ih

```
1  #include <string>
2  #include <iostream>
3
4  using namespace std;
5
6  string *factory(string const &str, size_t count);
```

main.cc

```
1  #include "main.ih"
2
3  int main()
4  {
5      string str = "test";
6      size_t count = 3;
7      string *sp = factory(hoi, count);
8      for (size_t idx = 0; idx != count; ++idx)
9          cout << sp[idx] << '\n';
10 }
```

factory.cc

```
1  #include "main.ih"
2
3  string *factory(string const &str, size_t count)
4  {
5      static string inputStr = str;       // made static s.t. Xstr has access
6
7      class Xstr: public string
8      {
9
10         public:
```

```
11              Xstr()
12              :
13                  string(inputStr)
14              {}
15      };
16
17      return new Xstr[count];
18  }
```