Overloading operators/Friends/This

You can overload these operators in C++ :

+ - * / % ^ & | ~ ! = < > += -= += /= %= ^= &= |= <<
>> >>= <<= == != <= >= && || ++ -- ->* -> [] () new delete

You cannot overload these operators in C++ :

:: ?: . sizeof .*

These are the overloaded operators in this example.

>=	In this screen shot example, this comparison returns the houses which have least 3 bedrooms and 2 baths. The >= operator is comparing multiple fields.
<	In this example, this returns the houses which have a price which is less or equal than the price entered by the user. The screen shots use a value of 200,000.
==	In this screen shot example, this comparison returns the houses which are in the city of Washington and the state of MS.

This is an example of a friend function, which is allowed to access a private variable.

A practical example could be that a realtor is allowed to change the price if it needs to be lowered to sell.

The fields such as address, city, state, zip, bedrooms and baths would not be normally changed since they are fixed values.

F:\housesclassesaoverlading.cpp

```
1   using namespace std;
3   #include <fstream>
4   #include <iostream>
5   #include <string>
6   #include <iomanip>
7   class houserecord
8   {
9   private :
10      double price;
11
12  public:
13      string address;
14      string city;
15      string state;
16      string zip;
17      double bathrooms;
18      int bedrooms;
19
20      void printhouse();
21      void addhouse();
22      void printform();
23      void setprice(double newprice)
24      { price=newprice;
25      }
26
27      bool operator>= (const houserecord &house) const;
28      bool operator< (const houserecord &house) const;
29      bool operator== (const houserecord &house) const;
30
31     friend void changeprice(houserecord & house);
32
33  };
34
```

```
35    bool houserecord::operator>= (const houserecord &house) const
36
37    {
38      if ((house.bedrooms  >= this->bedrooms) &&
39          (house.bathrooms >= this->bathrooms)
40          )
41          return true;
42          else return false;
43            };
44
45    bool houserecord::operator< (const houserecord &house) const
46
47    {  if (house.price  <= this->price)
48          return true;
49          else return false;
50            };
51
52    bool houserecord::operator== (const houserecord &house) const
53
54    {  if ((house.city == this->city) &&
55        (house.state == this->state))
56          return true;
57          else return false;
58            };
59
60  //---------------------------------------------------------------------
61
62    void printheader();
63    void changeprice(houserecord & house)
64    {
65      cout <<"Enter new price for house "<<house.address<<" ";
66      cin >>house.price;
67    };
68
69  void loadarray(houserecord house[], int size);
70
```

The ***this*** keyword is normally a hidden pointer and refers to the object which invoked the function.

You can explicitly make this reference and is sometimes return the pointer to the object.

These are the overloaded operators in this example.

>=	In this screen shot example, this comparison returns the houses which have least 3 bedrooms and 2 baths. The >= operator is comparing multiple fields.
<	In this example, this returns the houses which have a price which is less than or equal the price entered by the user. The screen shots use a value of 200,000. Notice that the operator of < is not the exact logic in the function. This type is permitted but is stupid to write code like this.
==	In this screen shot example, this comparison returns the houses which are in which are in the city of Washington and the state of MS.

This is an example of a friend function, which is allowed to access a private variable.

Notice that this function is not a member of the class ***houserecord.***

The ***this*** keyword is normally a hidden pointer and refers to the object which invoked the function. It is sometimes needed to be explicitly referenced to avoid confusion as shown in this example.

```
void time::sethour (int hour)
{ if (hour>=0  && hour <24)
    this->hour=hour;  //this is needed to avoid confusion between the variable hour in the object and the parameter hour
}
```

More overloading operators (input >> and output<<)

You can also the input and output operators >> and << with and example as follows:

```
friend istream &operator>>(istream &input, houserecord &house);
friend ostream &operator<<(ostream &output, const houserecord &house);
```

You generally will make the >> and << input/output operators a friend of the class.

When an input or and output file is passed as a parameter, it normally should have the & as a reference. This because the file marker position is normally changed. Output files are normally changed as part of the process.

```
36
37  ostream &operator<<(ostream &output, const houserecord &house)
38  {
39    output <<"\n\n-------------------------------------------------------------------\n";
40    output << "Address "<< setw(23) << left << house.address
41           << "\nCity "<< setw(15) << left << house.city
42           << "State "<< setw(10) << left << house.state
43           << "Zip "<< setw(10) << left << house.zip
44           << "\nPrice "<< setw(8) << left << house.price
45           << "\nBathrooms "<< setw(6) << house.bathrooms
46           << "Bedrooms "<< setw(6) << house.bedrooms
47           << endl;
48    output <<"\n\n-------------------------------------------------------------------
49    return output;
50  };
```

```
217 //-------------------------------------------------------------------
218
219 void houserecord :: printform()
220
221 {
222   cout <<"\n\n-------------------------------------------------------------------
223   cout << "Address "<< setw(23) << left << address
224        << "\nCity "<< setw(15) << left << city
225        << "State "<< setw(10) << left << state
226        << "Zip "<< setw(10) << left << zip
227        << "\nPrice "<< setw(8) << left << price
228        << "\nBathrooms "<< setw(6) << bathrooms
229        << "Bedrooms "<< setw(6) << bedrooms
230        << endl;
231   cout <<"\n\n-------------------------------------------------------------------\n";
232 }
```

In this example, the function ***printform()*** has been modified to be an overloaded operator <<.

The advantage to this method is that the form can be printed to any output specified by the parameter output.

It could be to the

- Console
- Printer device
- Datafile
- A port connecting to the internet or any device.
- Other devices as specified.

When an input or and output file is passed as a parameter, it normally should have the & as a reference. This because the file marker position is normally changed. Even though an input file does not change since it was being read from, where the file position marker in the file does change as part of the process.

```
51
52  istream &operator>>(istream &input, houserecord &house)
53  {
54    cout << "In overloaded operator\nEnter Address ";
55    getline(input, house.address);
56    cout << "Enter city ";
57    getline(input, house.city);
58    cout << "Enter state ";
59    getline(input, house.state);
60    cout << "Enter zip ";
61    getline(input, house.zip);
62    cout << "Enter price ";
63    input >> house.price;
64    cout << "Enter bathrooms ";
65    input >> house.bathrooms;
66    cout << "Enter bedrooms ";
67    input >> house.bedrooms;
68    cout << "\n\n";
69
70  };
```

```
196  void houserecord :: addhouse()
197
198  { cout << "Enter Address ";
199    getline(cin, address);
200    cout << "Enter city ";
201    getline(cin, city);
202    cout << "Enter state ";
203    getline(cin, state);
204    cout << "Enter zip ";
205    getline(cin, zip);
206    cout << "Enter price ";
207    cin >> price;
208    cout << "Enter bathrooms ";
209    cin >> bathrooms;
210    cout << "Enter bedrooms ";
211    cin >> bedrooms;
212    cout << "\n\n";
213  }
```

In this example, the function ***addhouse()*** has been modified to be an overloaded operator >>.

The advantage to this method is that the information can be retrieved from any input specified by the parameter input.

It could be to the

- Console
- Datafile
- A port connecting to the internet or any device.
- Other sensors or devices as specified.

The user prompts which appear on the screen will look pretty stupid if the input is coming from anywhere than user input.

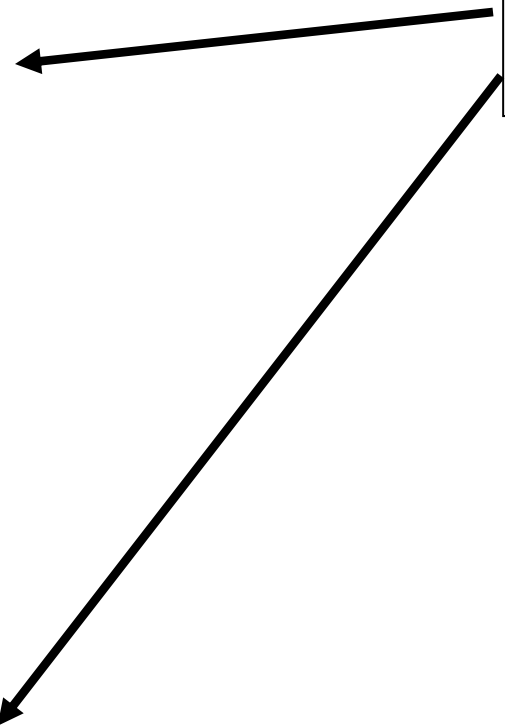
```
111 //-----------------------------------------------------------------
112 void houserecord::printhouse ()
113 {cout << setw(23) << left << address
114     << setw(15) << left << city
115     << setw(10) << left << state
116     << setw(10) << left << zip
117     << setw(8) << right << price
118     << setw(6) << right << bathrooms
119     << setw(6) << right << bedrooms << endl
120     << endl;
121     return;
122 }
123 //-----------------------------------------------------------------
124 void loadarray(houserecord house[], int size)
125 {   ifstream fin;
126     int i; double price;
127     fin.open("house.txt");
128     for (i = 0; i < size; i++)
129     {   getline(fin, house[i].address);
130         getline(fin, house[i].city);
131         getline(fin, house[i].state);
132         getline(fin, house[i].zip);
133         fin >> price;
134         house[i].setprice(price);
135         fin >> house[i].bathrooms;
136         fin >> house[i].bedrooms;
137         fin.ignore();
138     }
139 }
140 //-----------------------------------------------------------------
141 void houserecord :: addhouse()
142 { cout << "Enter Address ";
143   getline(cin, address);
144   cout << "Enter city ";
145   getline(cin, city);
146   cout << "Enter state ";
147   getline(cin, state);
148   cout << "Enter zip ";
149   getline(cin, zip);
150   cout << "Enter price ";
151   cin >> price;
152   cout << "Enter bathrooms ";
153   cin >> bathrooms;
154   cout << "Enter bedrooms ";
155   cin >> bedrooms;
156   cout << "\n\n";
157 }
158
```

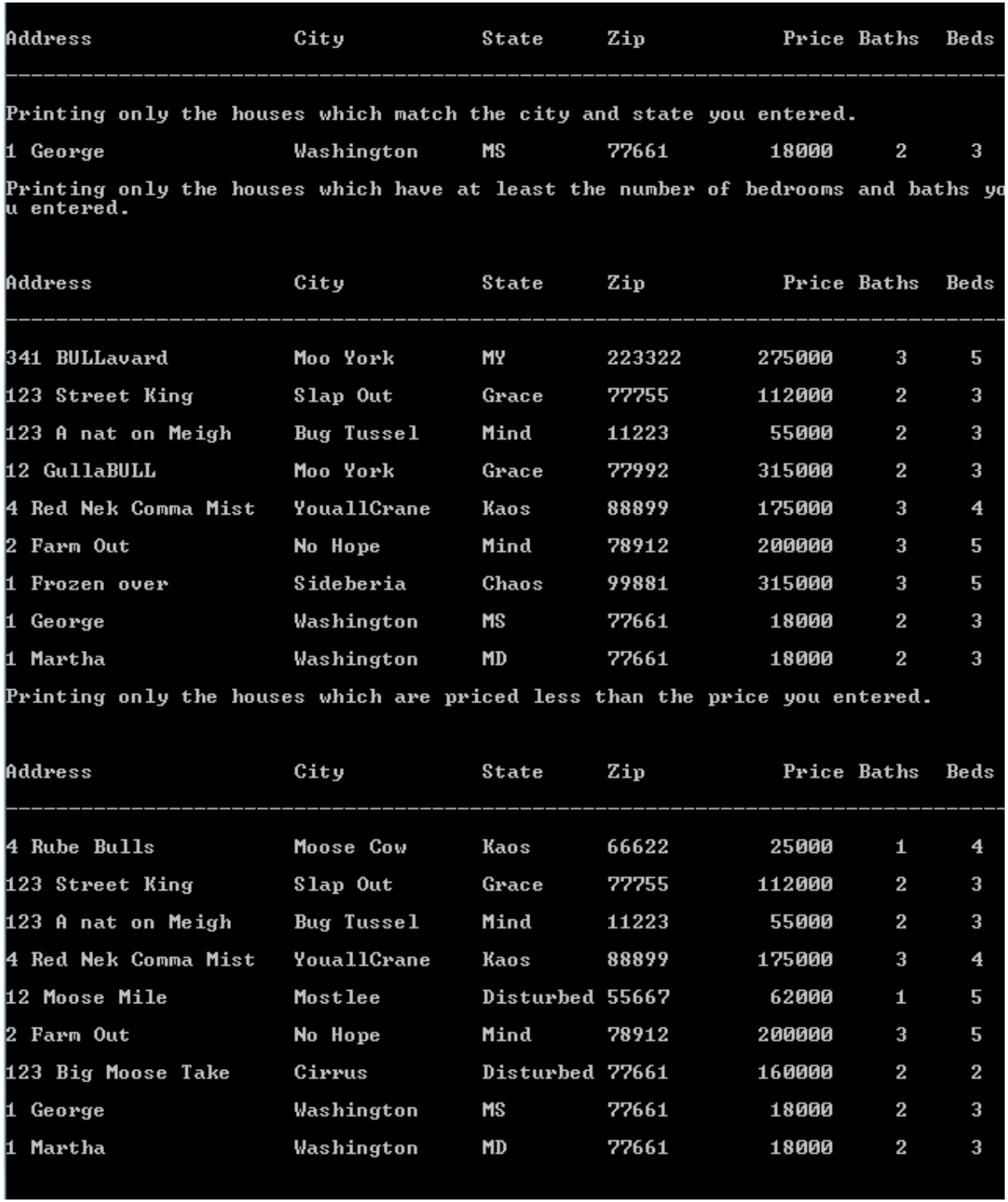
```
164  //-------------------------------------------------------------------------
165  void houserecord :: printform()
166
167  { cout <<"\n\n-------------------------------------------------------------------------\n";
168    cout << "Address "<< setw(23) << left << address
169         << "\nCity "<< setw(15) << left << city
170         << "State "<< setw(10) << left << state
171         << "Zip "<< setw(10) << left << zip
172         << "\nPrice "<< setw(8) << left << price
173         << "\nBathrooms "<< setw(6) << bathrooms
174         << "Bedrooms "<< setw(6) << bedrooms
175         << endl;
176    cout <<"\n\n-------------------------------------------------------------------------\n";
177  }
178
179  //-------------------------------------------------------------------------
180     void printheader()
181     { int i;
182         cout <<"\n\n"<< setw(23) << left << "Address"
183              << setw(15) << left << "City"
184              << setw(10) << left << "State"
185              << setw(10) << left << "Zip"
186              << setw(9) << right << "Price"
187              << setw(6) << right << "Baths"
188              << setw(6) << right << "Beds" << endl
189              << endl;
190         for (i = 0; i < 80; i++)
191             cout << "-";
192         cout << endl;
193     }
```

```
Enter Address 123 Main
Enter city Washington
Enter state MS
Enter zip 22334
Enter price 200000
Enter bathrooms 2
Enter bedrooms 3


Address                City           State     Zip           Price Baths  Beds
-------------------------------------------------------------------------------

341 BULLavard          Moo York       MY        223322       275000     3     5
4 Rube Bulls           Moose Cow      Kaos      66622         25000     1     4
123 Street King        Slap Out       Grace     77755        112000     2     3
123 A nat on Meigh     Bug Tussel     Mind      11223         55000     2     3
12 GullaBULL           Moo York       Grace     77992        315000     2     3
4 Red Nek Comma Mist   YouallCrane    Kaos      88899        175000     3     4
12 Moose Mile          Mostlee        Disturbed 55667         62000     1     5
2 Farm Out             No Hope        Mind      78912        200000     3     5
1 Frozen over          Sideberia      Chaos     99881        315000     3     5
123 Big Moose Take     Cirrus         Disturbed 77661        180000     2     2
1 George               Washington     MS        77661         18000     2     3
1 Martha               Washington     MD        77661         18000     2     3
Enter new price for house 123 Big Moose Take 160000

Printing all the houses; notice the price for one of the houses (10) has changed
 due to the friend function.


Address                City           State     Zip           Price Baths  Beds
-------------------------------------------------------------------------------

341 BULLavard          Moo York       MY        223322       275000     3     5
4 Rube Bulls           Moose Cow      Kaos      66622         25000     1     4
123 Street King        Slap Out       Grace     77755        112000     2     3
123 A nat on Meigh     Bug Tussel     Mind      11223         55000     2     3
12 GullaBULL           Moo York       Grace     77992        315000     2     3
4 Red Nek Comma Mist   YouallCrane    Kaos      88899        175000     3     4
12 Moose Mile          Mostlee        Disturbed 55667         62000     1     5
2 Farm Out             No Hope        Mind      78912        200000     3     5
1 Frozen over          Sideberia      Chaos     99881        315000     3     5
123 Big Moose Take     Cirrus         Disturbed 77661        160000     2     2
1 George               Washington     MS        77661         18000     2     3
1 Martha               Washington     MD        77661         18000     2     3
```

Notice that the last two cities in this screen shot are the city of Washington but they are in different states.

Notice that the friend function has changed the price of the house 123 Big Moose Take





```
Address                City           State     Zip           Price Baths  Beds
-------------------------------------------------------------------------------
Printing only the houses which match the city and state you entered.
1 George               Washington     MS        77661        18000     2     3
Printing only the houses which have at least the number of bedrooms and baths yo
u entered.

Address                City           State     Zip           Price Baths  Beds
-------------------------------------------------------------------------------
341 BULLavard          Moo York       MY        223322       275000     3     5
123 Street King        Slap Out       Grace     77755        112000     2     3
123 A nat on Meigh     Bug Tussel     Mind      11223         55000     2     3
12 GullaBULL           Moo York       Grace     77992        315000     2     3
4 Red Nek Comma Mist   YouallCrane    Kaos      88899        175000     3     4
2 Farm Out             No Hope        Mind      78912        200000     3     5
1 Frozen over          Sideberia      Chaos     99881        315000     3     5
1 George               Washington     MS        77661         18000     2     3
1 Martha               Washington     MD        77661         18000     2     3
Printing only the houses which are priced less than the price you entered.

Address                City           State     Zip           Price Baths  Beds
-------------------------------------------------------------------------------
4 Rube Bulls           Moose Cow      Kaos      66622         25000     1     4
123 Street King        Slap Out       Grace     77755        112000     2     3
123 A nat on Meigh     Bug Tussel     Mind      11223         55000     2     3
4 Red Nek Comma Mist   YouallCrane    Kaos      88899        175000     3     4
12 Moose Mile          Mostlee        Disturbed 55667         62000     1     5
2 Farm Out             No Hope        Mind      78912        200000     3     5
123 Big Moose Take     Cirrus         Disturbed 77661        160000     2     2
1 George               Washington     MS        77661         18000     2     3
1 Martha               Washington     MD        77661         18000     2     3
```

Notice that the report printed the house in Washington MS but not Washington MD due to the overloaded operator ==

Notice that the report printed the houses which have at least three bedrooms and two baths due to the overloaded operator >=

Notice that the report printed the houses which are less than or equal $200,000 due to the overloaded operator <. Notice that the operator implies less than but actually is equivalent to <= due the actual definition in the function.

As discussed in class, you can define an operator in such a way that it can cause confusion.