CS 368 Announcements
Wednesday, November 13, 2013

Program p4

- due Wednesday, November 27
- covers operator overloading
- do not change Complex.h

Last Time

- overloading arithmetic ops
- `explicit`
- overloading output operator (**<<**)
- overloading relational ops
- `friend`
- overloading increment (**++**) and decrement (**--**)

Today

- start Ch. 9 (Input and Output)
- console I/O
- overloading **>>**
- error states
- file I/O

Next Time

- continue Ch. 9 (Input and Output)
- string class
- C strings
- manipulators
- C I/O

Console I/O

Point Class

```
/*
 * The Point class represents integer points (x, y)
 */
class Point {

    friend ostream & operator<<(ostream & out, const Point & p);

    friend istream & operator>>(istream & in, Point & p);

    public:

    // constructors
    Point(): x(0), y(0) { }
    Point(int a, int b): x(a), y(b) { }

    // accessors
    int getX() const { return x; }
    int getY() const { return y; }

    // mutators
    void setX(int newX) { x = newX; }
    void setY(int newY) { y = newY; }

    private:

    int x, y;
};
```

Using console I/O

```
/*
 * Prompts user for two points and displays the results.
 */
int main() {
    Point p, q;

    cout << "enter a point: ";
    cin >> p;

    cout << "you entered: " << p << endl;

    cout << "enter another point: ";
    cin >> q;

    cout << "you entered: " << q << endl;

    return 0;
}
```

Overloading << for output

```
/*
 * Overload <<
 */
ostream & operator<<(ostream & out, const Point & p) {
    out << "(" << p.x << "," << p.y << ")";
    return out;
}
```

Overloading >> for input

```
istream & operator>>(istream & in, Point & p) {
    char ch;
    int x, y;

    in.get(ch);
    while (ch == ' ' || ch == '\n' || ch == '\t')
        in.get(ch);

    if (ch != '(') {

        return in;
    }

    in >> x;          // read in x value

    in.get(ch);
    while (ch == ' ' || ch == '\n' || ch == '\t')
        in.get(ch);

    if (ch != ',') {

        return in;
    }

    in >> y;          // read in y value

    in.get(ch);
    while (ch == ' ' || ch == '\n' || ch == '\t')
        in.get(ch);

    if (ch != ')') {

        return in;
    }

    // update p only after the entire point has been read
    p.x = x;
    p.y = y;
    return in;
}
```

Error States

File I/O

```
#include <fstream>
```

Input using `ifstream`

Output using `ofstream`

Useful functions for `istreams` (which includes `ifstreams` and `cin`)

- `in.eof()`

- `in.get(ch)`

- `in.unget()`

- `in.peek()`

- `in.ignore(n, delim)`

- `in.getline(myString, n)`

- `getline(in, myString)`

File I/O Example (from fileIO.cpp)

```
ifstream inFile("inFile.txt");
if (!inFile) {
    cerr << "Unable to open inFile.txt for reading" << endl;
    return 1;
}

ofstream outFile("outFile.txt");
if (!outFile) {
    cerr << "Unable to open outFile.txt for writing" << endl;
    return 1;
}

char ch;
while (inFile.get(ch)) {
    switch (ch) {
      case 'a':  case 'A':
      case 'e':  case 'E':
      case 'i':  case 'I':
      case 'o':  case 'O':
      case 'u':  case 'U':
         outFile << '*';
         break;
      default:
         outFile << ch;
    }
}

inFile.close();
outFile.close();
```