In this exercise we implement overloaded operators to a simple class.

Excercise A (4 p) Implement operators

Improve class you wrote in exercise 4 by adding overloaded operators. The operators to add are:

1. Output operator (<<) that outputs the time in two character fields with leading zeros and separates the fields with a colon.
2. Comparison operator less than (<) that compares two times
3. Addition operator (+) that adds two times
4. Subtract operator (-) that subtracts two times.
5. Pre and post increment operators (++). Both operators increment the time by one minute

Your class should work with the test program below. Note that your class must have a default constructor that initializes time to 0:00.
Addition must make times to roll over to “next day” but doesn’t have to keep track of days. For example, adding 14:30 and 13:45 should result in 4:15 or adding 18:30 and 5:37 should yield 0:07.

The program below should work with your class:

```
void print(const vector<Time> &v)
{
	for(auto &t : v) {
		cout << t << endl;
	}
}



int main() {
	Time time1, time2, duration;

	time1.read("Enter time 1");
	time2.read("Enter time 2");
	if (time1<time2) {
		duration = time2 - time1;
		cout << "Starting time was " << time1 << endl;
	} else {
		duration = time1 - time2;
		cout << "Starting time was " << time2 << endl;
	}
	cout << "Duration was " << duration << endl;

	vector<Time> tv(5);
	for(auto &t : tv) {
		t.read("Enter time:");
	}

	cout << "Times: " << endl;
	print(tv);

	Time sum;
	for(auto t : tv) {
		sum = sum + t;
	}

	cout << "Sum of times: " << sum << endl;

	cout << "Post-increment: " << endl;
	print(tv);
	for(auto &t : tv) {
		cout << t++ << endl;
	}

	print(tv);

	cout << "Pre-increment: " << endl;
	for(auto &t : tv) {
		cout << ++t << endl;
	}

	sort(tv.begin(), tv.end());

	cout << "Sorted times: " << endl;
	print(tv);

	return 0;
}
```