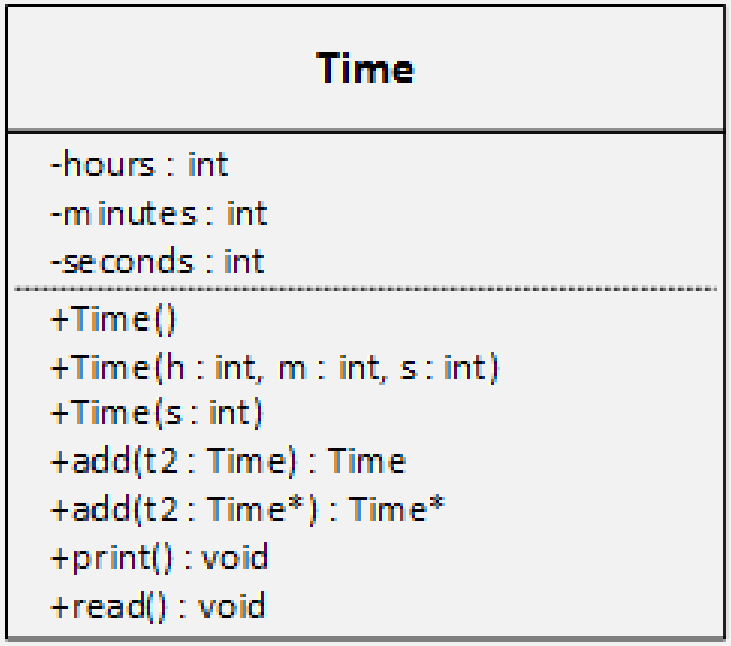
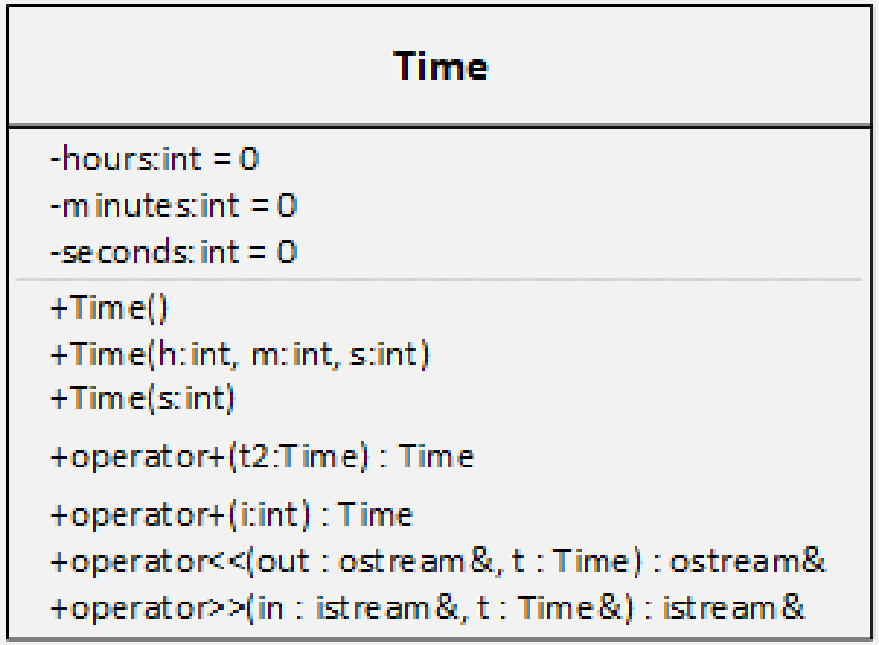
TIME

Overloaded Operators Version

Delroy A. Brinkerhoff

UML TIME CLASS DIAGRAMS





C++ Time CLASS

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

class Time
{
    private:
        int     hours = 0;
        int     minutes = 0;
        int     seconds = 0;

    public:
        Time() {}
        Time(int h, int m, int s) : hours(h), minutes(m), seconds(s) {}
        Time(int s);
        Time    operator+(Time t2);
        Time    operator+(int i);
        friend  ostream& operator<<(ostream& out, Time& t);
        friend  istream& operator>>(istream& in, Time& t);
};
```

ADDING TIME

```
Time Time::operator+(Time t2)
{
    int i1 = hours * 3600 + minutes * 60 + seconds;
    int i2 = t2.hours * 3600 + t2.minutes * 60 + t2.seconds;

    return Time(i1 + i2);
}

Time Time::operator+(int i)
{
    int i1 = hours * 3600 + minutes * 60 + seconds;

    return Time(i1 + i);
}
```

Time I/O

```
ostream& operator<<(ostream& out, Time& t)
{
    out.fill('0');

    out << t.hours << ":"
        << setw(2) << t.minutes << ":"
        << setw(2) << t.seconds;

    out.fill(' ');

    return out;
}
```

```
istream& operator>>(istream& in, Time& t)
{
    cout << "Please enter the hours: ";
    cin >> t.hours;

    cout << "Please enter the minutes: ";
    cin >> t.minutes;

    cout << "Please enter the seconds: ";
    cin >> t.seconds;

    return in;
}
```

A SINGLE friend FUNCTION

```
class Time
{
    private:

    public:
        Time(int s);
        friend Time operator+(Time t1, Time t2);
};

Time operator+(Time t1, Time t2)
{
    int i1 = t1.hours * 3600 + t1.minutes * 60 + t1.seconds;
    int i2 = t2.hours * 3600 + t2.minutes * 60 + t2.seconds;

    return Time(i1 + i2);
}
```