Overloading Operators in the Time Class

1. Create a new Visual Studio project, empty console, named LastnameOverloadedOperators.
2. Add a new class, named Time. It will keep time in 24 hour context.
3. Add include guards to the .h file.
4. Add this class:

#ifndef TIME\_H

#define TIME\_H

class Time

{

private:

int hours{ 0 };

int minutes{ 0 };

public:

Time();

Time(int h, int m) : hours{ h }, minutes{ m }{};

int GetHours()const { return hours; }

int GetMinutes()const { return minutes; }

};

#endif

1. Let’s consider that we might want to compute the difference between two times. If we have two times:

Time morning{9, 30};

Time lunch{2, 0};

Then we’d like to say: mimnutesToLunch = lunch – morning;

1. To do that, we will overload the – operator for this class. Maybe the + operator, too. Good things come in pairs.
2. Add these lines to the .h file:

int operator-( const Time& other ) const;

int operator+( const Time& other) const;

1. Now open the .cpp file. Find the - and + methods. Add this code:

return hours \* 60 + minutes - (other.hours \* 60 + other.minutes);

return hours \* 60 + minutes +(other.hours \* 60 + other.minutes);

1. Let’s also just add /subtract some minutes to the current time and return a modified Time object. Open the .h file. Add:

int operator+(int mins);

int operator-(int mins);

1. Open the .cpp file and add:

int resultMinutes = hours \* 60 + minutes + mins;

return Time(resultMinutes / 60, resultMinutes % 60);

and

int resultMinutes = hours \* 60 + minutes - mins;

return Time(resultMinutes / 60, resultMinutes % 60);

1. What about checking to see if it’s the same time? Open the .h file and enter:

bool operator==( const Time& other) const;

Probably also should add != : bool operator!=( const Time& other) const;

1. The in the implementations:

return \*this - other == 0;

return \*this - other != 0;

1. And what about sorting time objects? How do we handle that? We need to use > and < operators to do that.
2. Open the .h file and add:

bool operator>(const Time& other) const;

bool operator<(const Time& other)const ;

1. Then open the .cpp file and add into the method stubs:

return(minutes + hours \* 60 > other.minutes \* other.hours \* 60);

return(minutes + hours \* 60 < other.minutes \* other.hours \* 60);

1. What else can we do? Increment Operators
2. Open the .h file and add:

Time operator++();

Time operator++(int dummy);

1. Now open the .cpp file and add:

\*this = \*this + 1;

return \*this;

and

Time original = \*this;

\*this = \*this + 1;

return original;

1. Add a new .cpp file and name it Driver.
2. Add the usual program header, program purpose and file name.
3. Add #include <iostream> and using namespace std;
4. Add int main()
5. Add this code:

cout << "\n Let's say you are scheduling your day";

cout << "\n On Tuesday, you get up at 7:45 am. You have a class at 9:30 am.";

cout << "\nHow long do you have to get up and to class?";

Time awake{ 7, 45 };

Time getToClass{ 9, 30 };

int timeToGetThrere = getToClass - awake;

cout << "\n The time to get ready for class is " << timeToGetThere

<< " minutes.";

1. When do you have to get up if you need to be there 15 minutes early?

cout << "\n The time to get up and be there 15 minutes early for class is " << getUpEarly.GetFormattedString();

1. cout << "\n I need to organize my appointments for Friday. There are four. ";
2. They are:

Walk 3 miles for exercise at 8 am for 45 minutes.

Online meeting on Zoom with my mentor at 3pm

Lunch date at 12 with my friends

Dinner at home at 6:30pm with my dog, Leia.

1. Sort the appointments by time so I know when to get ready for the next one.
2. I can do this using the Bubble sort
3. First, put the appointments into an array, create objects
4. Time walk{ 8, 0 };
5. Time meeting{ 15, 0 };
6. Time lunch{ 12, 0 };
7. Time dinner{ 18, 30};
8. Time appts[] = { meeting, lunch, dinner, walk };
9. Start the Bubble Sort
10. Time temp;
11. int total{ 4 };
12. for (int i = 0; i < total - 1; ++i)
13. {
14. for (int j = 1; j < total; ++j)
15. {
16. if (appts[j - 1] > appts[j])
17. {
18. temp = appts[j];
19. appts[j] = appts[j - 1];
20. appts[j - 1] = temp;
21. }
22. }
23. }
24. cout << "\n My Friday schedule:\n";
25. for (int x = 0; x < total; ++x)
26. {
27. cout << appts[x].GetFormattedString() << endl;
28. }
29. cout << "\n But I have to walk today. My car is not running!";
30. cout << "\n That will take me twice as long to get to school.";
31. cout << "\n I hope I can get there in time if I get up at 6:30am";
32. Time getUp{ 6, 30 };

Time itWillTake{getUp + timeToGetThere\*2};

1. if (itWillTake < getToClass)
2. {
3. cout << "\n I made it!
4. }
5. else
6. {
7. cout<<"\n Whoops! tardy again.";
8. }
9. cout<< ”\n I will be there at " << itWillTake.GetFormattedString();

Run your project.