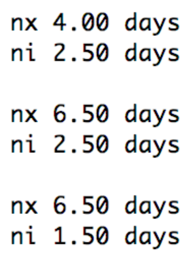
CSC 122 001 Computer Science II
Julius Ranoa

Chapter 11 Programming Challenge 4 Number of Days Worked
Design a class called *NumDays* that stores a value that represents a number of work hours and convert it to a number of days. It should have a constructor that accepts a number of hours, as well as member functions for storing and retrieving the hours and days. The class should also have the following overloaded operators: addition, subtraction, prefix and postfix increment and decrement operators.

Screenshot of runtime.



```
nx 4.00 days
ni 2.50 days

nx 6.50 days
ni 2.50 days

nx 6.50 days
ni 1.50 days
```

Files included: (1) main.cpp, (2) NumDays.h, (3) NumDays.cpp

main.cpp

```cpp
#include <iostream>
#include "NumDays.h"
using namespace std;

int main() {
    NumDays nx(32), ni(20);

    cout << "nx " << nx << "\n";
    cout << "ni " << ni << "\n\n";

    nx += ni;

    cout << "nx " << nx << "\n";
    cout << "ni " << ni << "\n\n";

    ni -= 8;

    cout << "nx " << nx << "\n";
    cout << "ni " << ni << "\n\n";

    return 0;
}
```

NumDays.h

```
#ifndef CH11_PR4_NUM_DAYS_WORKED_NUMDAYS_H
#define CH11_PR4_NUM_DAYS_WORKED_NUMDAYS_H

#include <iostream>

class NumDays {

public:
    static const int hoursPerDay;

private:
    int hours;
    double days;

    void calcDays();

public:
    NumDays();
    NumDays(int);

    int getHours() const { return hours; };
    double getDays() const { return days; };
    void setHours(const int);
    void addHours(const int);

    // All variants of addition
    friend NumDays operator+(NumDays, const NumDays&);
    friend NumDays operator+(int, NumDays);
    friend NumDays operator+(NumDays, int);
    NumDays& operator+=(const NumDays&);
    NumDays& operator+=(const int);

    // All variants of subtraction
    friend NumDays operator-(NumDays, const NumDays&);
    friend NumDays operator-(int, NumDays);
    friend NumDays operator-(NumDays, int);
    NumDays& operator-=(const NumDays&);
    NumDays& operator-=(const int);

    // Printing because I'm lazy. lol
    friend std::ostream& operator<<(std::ostream&, const NumDays&);
};


#endif //CH11_PR4_NUM_DAYS_WORKED_NUMDAYS_H
```

NumDays.cpp

```cpp
#include <iomanip>
#include "NumDays.h"

// Calculation-relevant members and methods.

const int NumDays::hoursPerDay = 8;

void NumDays::calcDays() {
    days = static_cast<double>(hours) / hoursPerDay;
}

// Constructors

NumDays::NumDays(int h) {
    setHours(h);
}

NumDays::NumDays() : NumDays(0) { }

// Setters

void NumDays::setHours(const int h) {
    hours = ( h >= 0 ) ? h : 0;
    calcDays();
}

void NumDays::addHours(int hx) {
    hours += hx;
    if (hours < 0) hours = 0;
    calcDays();
}

// Addition Operations

NumDays operator+(NumDays n1, const NumDays& n2) {
    n1.addHours( n2.getHours() );
    return n1;
}

NumDays operator+(NumDays n1, int n2) {
    n1.addHours(n2);
    return n1;
}

NumDays operator+(int n1, NumDays n2) {
    n2.addHours(n1);
    return n2;
}

NumDays& NumDays::operator+=(const NumDays& n2) {
    this->addHours( n2.getHours() );
    return *this;
}
```

```cpp
NumDays& NumDays::operator+=(const int n2) {
    this->addHours( n2 );
    return *this;
}

// Subtraction Operations

NumDays operator-(NumDays n1, const NumDays& n2) {
    n1.addHours( -n2.getHours() );
    return n1;
}

NumDays operator-(NumDays n1, int n2) {
    n1.addHours(-n2);
    return n1;
}

NumDays operator-(int n1, NumDays n2) {
    n2.addHours(-n1);
    return n2;
}

NumDays& NumDays::operator-=(const NumDays& n2) {
    this->addHours( -n2.getHours() );
    return *this;
}

NumDays& NumDays::operator-=(const int n2) {
    this->addHours( -n2 );
    return *this;
}

// Stream insertion operator.

std::ostream& operator<<(std::ostream& out, const NumDays& nx) {
    static std::ios state(NULL);
    state.copyfmt(out);
    out << std::showpoint << std::fixed << std::setprecision(2);
    out << nx.getDays() << " day" << (nx.getDays() == 1 ? "" : "s");
    out.copyfmt(state); // Removes iomanip formatting.
    return out;
}
```