Distance.h

#ifndef DISTANCE\_H

#define DISTANCE\_H

#include <istream>

#include <ostream>

class Distance //English Distance class

{

private:

int feet;

float inches;

public: //constructor (no args)

Distance() : feet(0), inches(0.0)

{ } //constructor (two args)

Distance(int ft, float in) : feet(ft), inches(in)

{ }

//Distance(float allc);

Distance operator+(const Distance & dis2) const;

friend Distance operator-(Distance d1, Distance d2);

friend std::ostream& operator<<(std::ostream& out, const Distance& d);

friend std::istream & operator>>(std::istream & in, Distance & d);

};

#endif // !DISTANCE\_H

Distance.cpp

#include "Distance.h"

#include <iostream>

#include <istream>

#include <ostream>

using namespace std;

Distance Distance::operator+(const Distance& dis2) const

{

float total = (feet \* 12 + inches) + (dis2.feet \* 12 + dis2.inches);

int ft = std::floor(total / 12);

float in = total - 12 \* ft;

Distance local(ft, in);

return local;

}

Distance operator-(Distance d1, Distance d2)

{

Distance d;

float result = (d1.feet \* 12 + d1.inches) - (d2.feet \* 12 + d2.inches);

if ((d1.feet \* 12 + d1.inches) >= (d2.feet \* 12 + d2.inches)) {

d.feet = std::floor(result / 12);

}

else {

d.feet = std::ceil(result / 12);

}

d.inches = result - 12 \* d.feet;

return d;

}

ostream& operator<<(ostream& out, const Distance& d)

{

out << d.feet << "'" << d.inches << "\"";

return out;

}

istream& operator>>(istream& in, Distance& d)

{

in >> d.feet >> d.inches;

return in;

}

Main.cpp

#include "Distance.h"

#include <iostream>

using namespace std;

int main()

{

Distance dist1, dist3, dist4; //define distances

cin >> dist1;

Distance dist2(11, 6.25); //define, initialize dist2

dist3 = dist1 + dist2; //single '+' operator

dist4 = dist1 - dist2; //friend '-' operators

//display all lengths

cout << "dist1 = ";

cout << dist1 << endl;

cout << "dist2 = ";

cout << dist2 << endl;

cout << "dist3 = ";

cout << dist3 << endl;

cout << "dist4 = ";

cout << dist4 << endl;

return 0;

}

Demonstrated at 11:26 am on 10/14/2021

