

## - Exercises

In this exercise, you will extend the MyDistance class from the example in the previous lesson.

### WE'LL COVER THE FOLLOWING ^

- Task 1
- Task 2

## Task 1 #

Extend `MyDistance` from the [previous example](#) to support the following units:

### 1. Feet (ft)

- $1 \text{ ft} = 0.3048 \text{ m}$

### 2. Mile (mi)

- $1 \text{ mi} = 1609.344 \text{ m}$

Tip: Define a good suffix for these units as well.

## Task 2 #

Your weekly drive with your car consists of many components: `work`, `workPerDay`, `workout`, `abbreviationToWork` and `shopping`.

```
workPerDay = 2 * work;
```

Extend `MyDistance` so you can calculate the total distance of your weekly drive. An example follows:

```
Distance::myDistance myDisPerWeek;
```

```
myDistPerWeek= 4 * workPerDay - 3 * abbreviationToWork + workout + shopping
;
```

```
//userDefinedLiteral.cpp
#include <iostream>
#include <ostream>

namespace Distance{
    class MyDistance{
    public:
        MyDistance(double i):m(i){}

        friend MyDistance operator +(const MyDistance& a, const MyDistance& b){
            return MyDistance(a.m + b.m);
        }
        friend MyDistance operator -(const MyDistance& a, const MyDistance& b){
            return MyDistance(a.m - b.m);
        }

        friend std::ostream& operator<< (std::ostream &out, const MyDistance& myDist){
            out << myDist.m << " m";
            return out;
        }
    private:
        double m;
    };

    namespace Unit{
        MyDistance operator "" _km(long double d){
            return MyDistance(1000*d);
        }
        MyDistance operator "" _m(long double m){
            return MyDistance(m);
        }
        MyDistance operator "" _dm(long double d){
            return MyDistance(d/10);
        }
        MyDistance operator "" _cm(long double c){
            return MyDistance(c/100);
        }
    }
}

using namespace Distance::Unit;

int main(){

    std::cout << std::endl;

    std::cout << "1.0_km: " << 1.0_km << std::endl;
    std::cout << "1.0_m: " << 1.0_m << std::endl;
    std::cout << "1.0_dm: " << 1.0_dm << std::endl;
    std::cout << "1.0_cm: " << 1.0_cm << std::endl;

    std::cout << std::endl;
    std::cout << "1.0_km + 2.0_dm + 3.0_dm - 4.0_cm: " << 1.0_km + 2.0_dm + 3.0_dm - 4.0_cm <
    std::cout << std::endl;

}
```



---

You can find the solution to these tasks in the next lesson.