Lab 3 Report

Group 25: Randy Hucker and Sam Graler

Instructions:

.cpp files for all Tasks are located in the same visual studio project. Open / run the visual studio project as normal.

A description of the objectives/concepts explored in this assignment including why you think they are important to this course and a career in CS and/or Engineering.

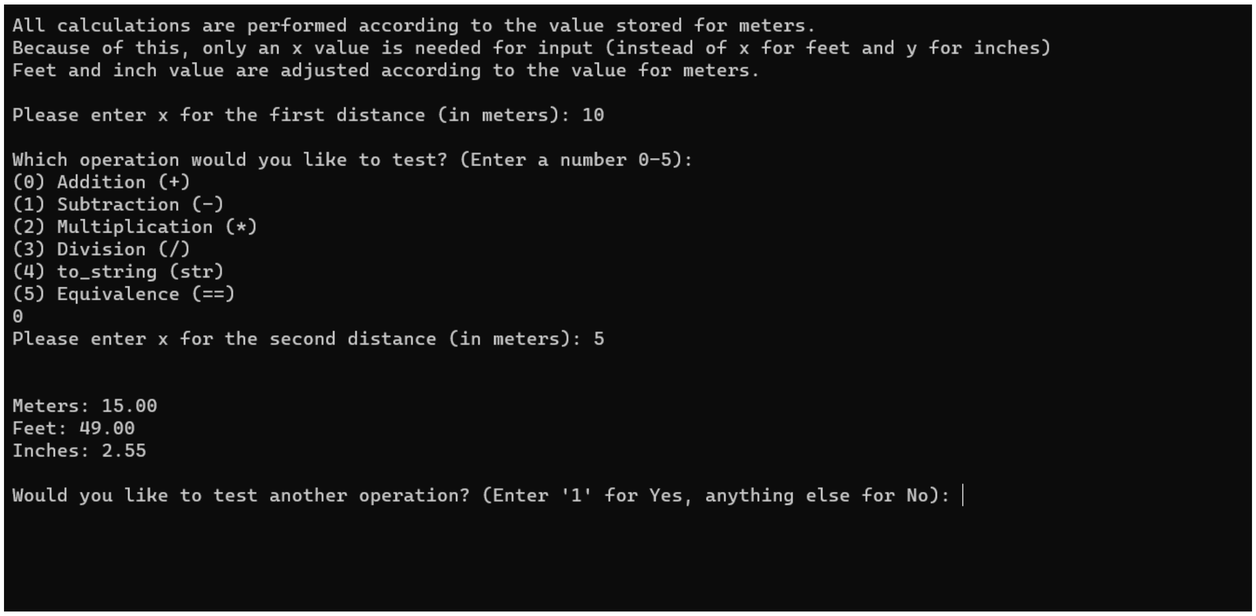
The main concepts we explored in this lab were class design and operator overload. To reiterate the importance of classes, binding data together is an essential concept in computer science. Encapsulation makes code easier to follow, faster to write, and allows for easy re-use in the future. It is also an important way of thinking to develop even if you aren't in CS. In engineering, it is important to think through the future needs / features of products you design, and this is very similar to trying to design a class from the ground up. Operator overloading is important specifically relating to classes because it allows you to simplify complex operations into shorter calls. It is also useful because you can design a more intuitive version of the operator you're using (+ switches to adding the individual members of the class), or make it something completely different.

Why you designed the class the way you did initially, what changes you made because of each task and what considerations you consider important when designing classes.

Our class was designed to only hold 3 private variables - our three distances, and the other member functions were all to be public. We decided on this approach in Task 1 and continued with it through task 4. We made the three distance variables private per the lab instructions. This seemed obvious to us as we would need getters/setters later in the lab to be able to perform calculations. We also elected to put our operator overloads in public for the same reasons. We didn't make many changes to the class structure - just appended to it as needed as the lab continued and more functions were required. We hypothesized that having all variables available when needed would probably be a good approach which led us to designing the class in this manner. The only changes we made that were unique to us and not required by the lab were the creation of our two "Distance Update" functions. This was added in task 2 and wasn't required, however, we found them necessary to perform updates when doing overloaded operations.

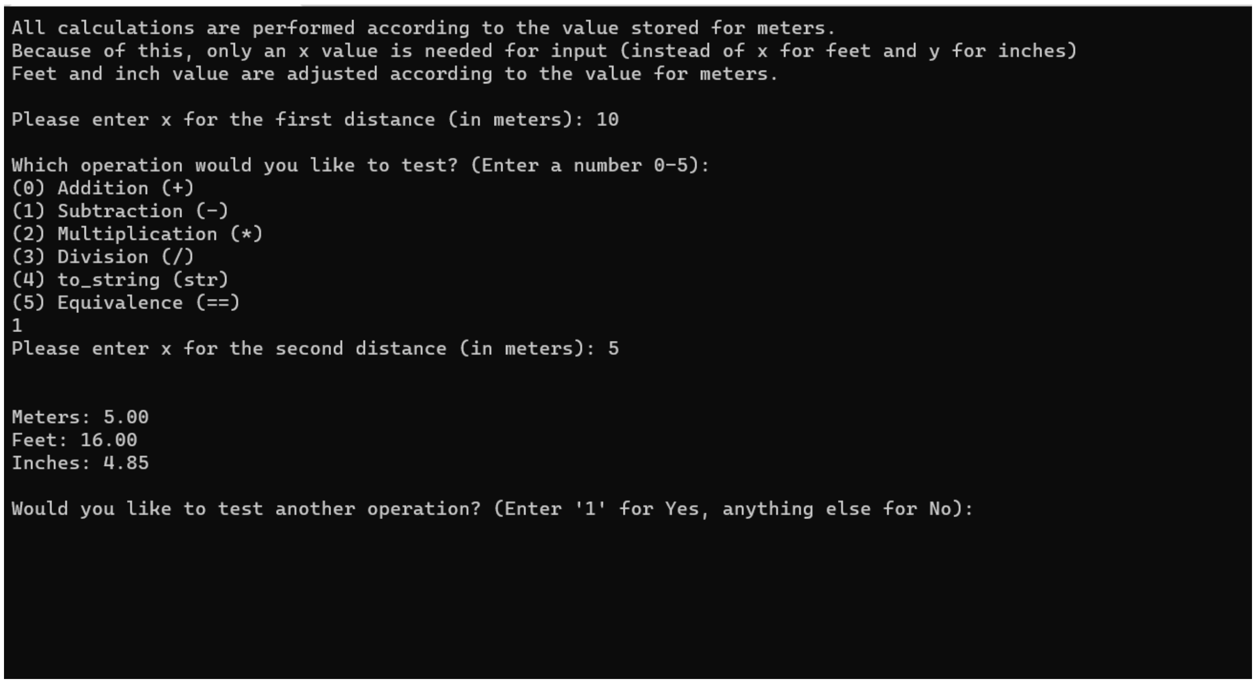
Output Screenshots:

Addition:



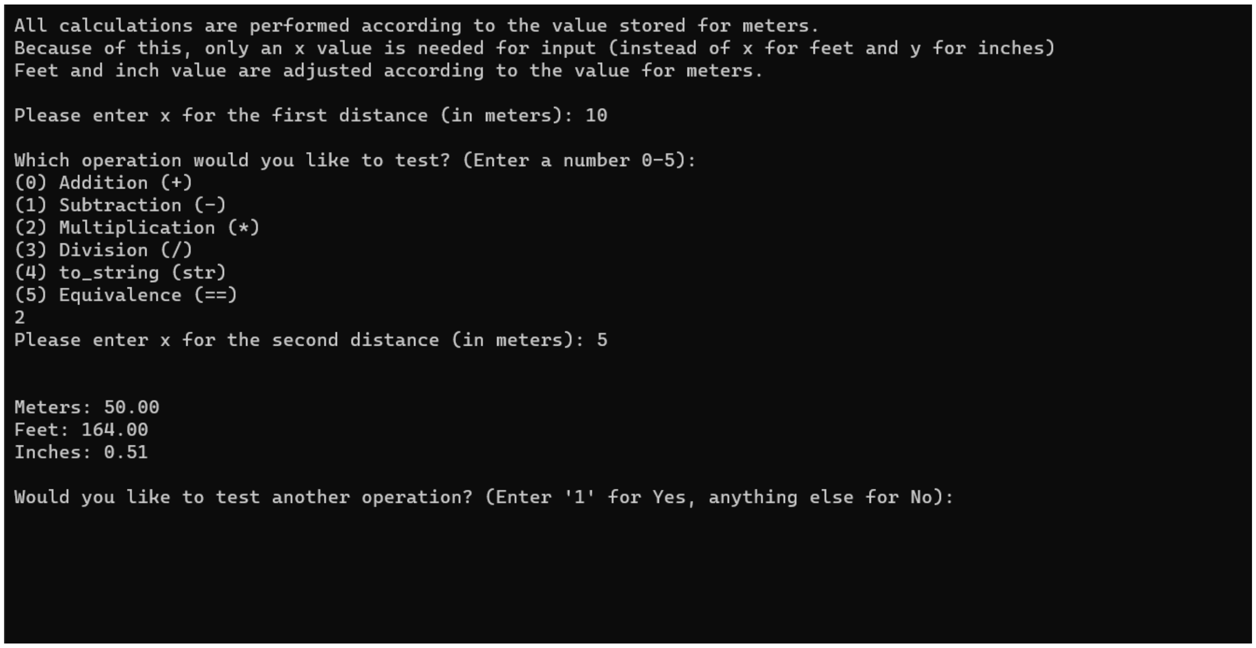
```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
0
Please enter x for the second distance (in meters): 5


Meters: 15.00
Feet: 49.00
Inches: 2.55

Would you like to test another operation? (Enter '1' for Yes, anything else for No): 
```

Subtraction:



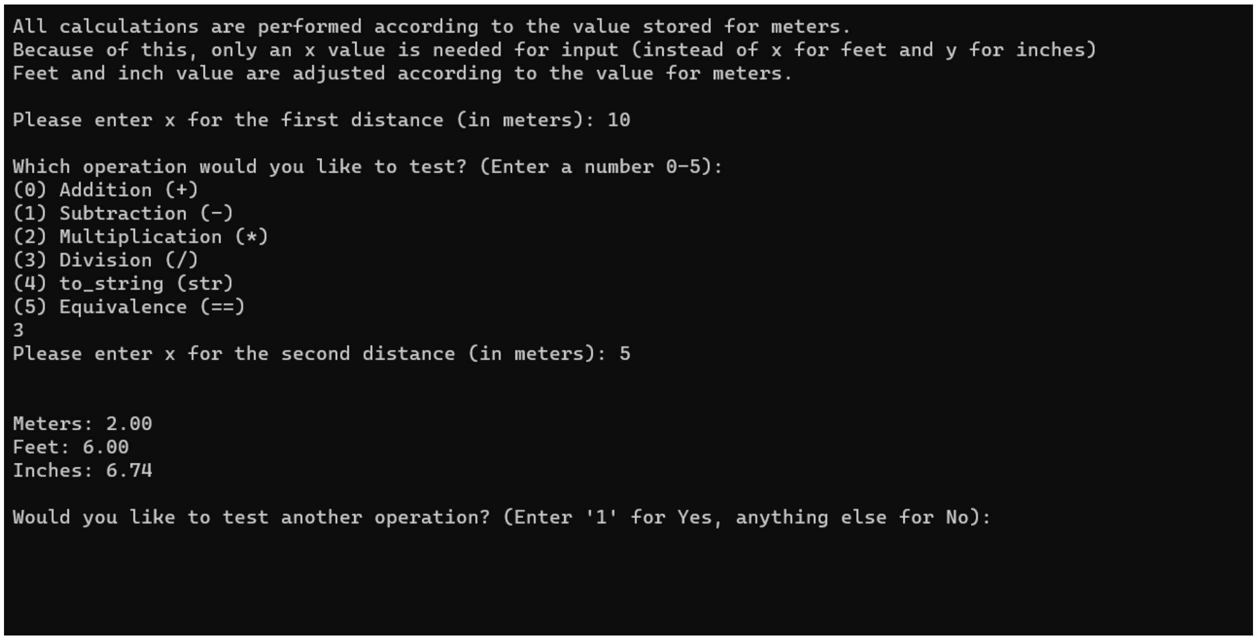
```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
1
Please enter x for the second distance (in meters): 5


Meters: 5.00
Feet: 16.00
Inches: 4.85

Would you like to test another operation? (Enter '1' for Yes, anything else for No):
```

Multiplication:



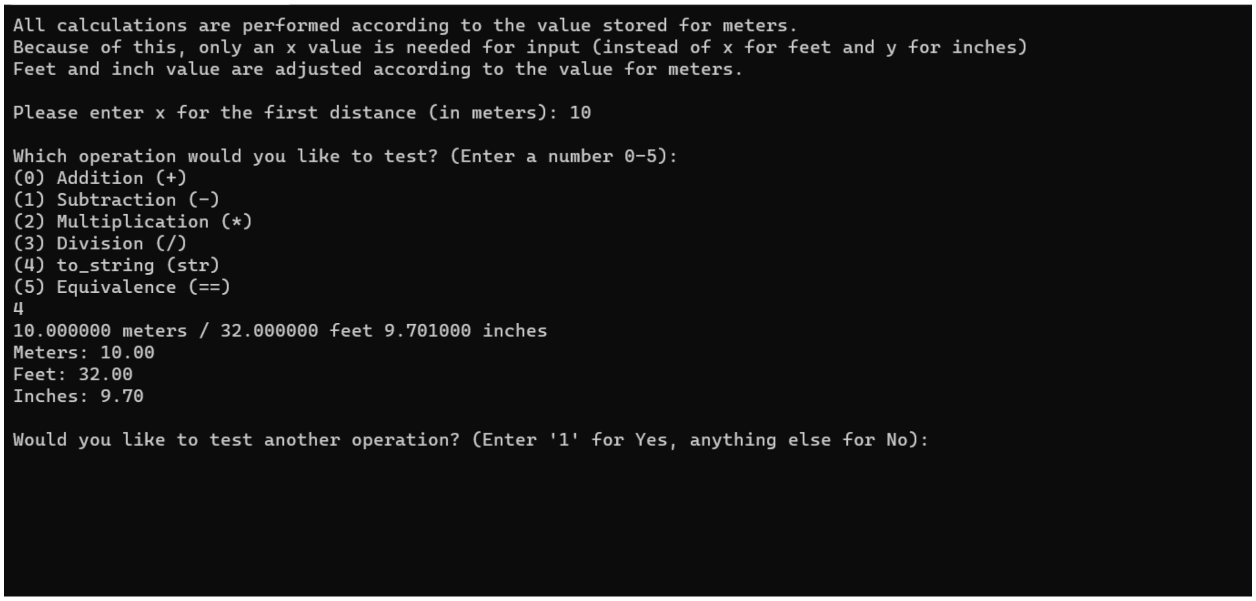
```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
2
Please enter x for the second distance (in meters): 5


Meters: 50.00
Feet: 164.00
Inches: 0.51

Would you like to test another operation? (Enter '1' for Yes, anything else for No):
```

Division:



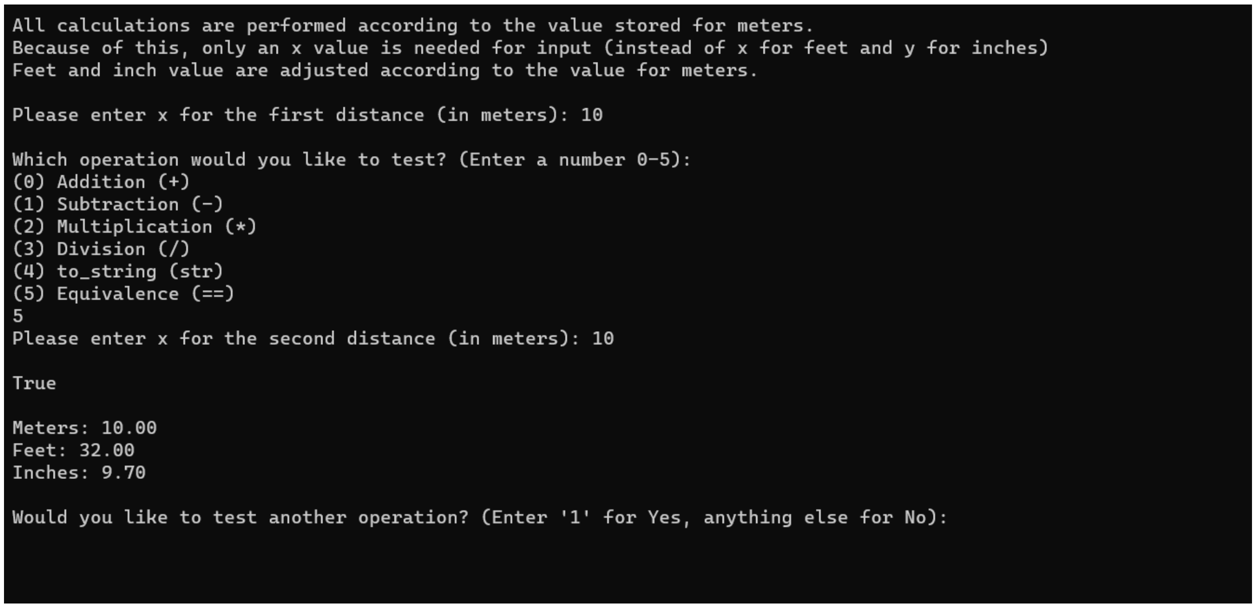
```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
3
Please enter x for the second distance (in meters): 5


Meters: 2.00
Feet: 6.00
Inches: 6.74

Would you like to test another operation? (Enter '1' for Yes, anything else for No):
```

ToString:



```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
4
10.000000 meters / 32.000000 feet 9.701000 inches
Meters: 10.00
Feet: 32.00
Inches: 9.70

Would you like to test another operation? (Enter '1' for Yes, anything else for No):
```

Equivalence:



```
All calculations are performed according to the value stored for meters.
Because of this, only an x value is needed for input (instead of x for feet and y for inches)
Feet and inch value are adjusted according to the value for meters.

Please enter x for the first distance (in meters): 10

Which operation would you like to test? (Enter a number 0-5):
(0) Addition (+)
(1) Subtraction (-)
(2) Multiplication (*)
(3) Division (/)
(4) to_string (str)
(5) Equivalence (==)
5
Please enter x for the second distance (in meters): 10

True

Meters: 10.00
Feet: 32.00
Inches: 9.70

Would you like to test another operation? (Enter '1' for Yes, anything else for No):
```