Arithmetic Operators and Variable Typing

April 20, 2018

1 Arithmetic Operators

Arithmetic operations in C++ are similar to other languages.

Operator	Opearation
+	Addition
-	Substraction
	Multiplication
/	Division
%	Modulo

An useful module for arithmetic operations is `cmath` which contains a few cool functions. For example std::pow(3, 5) for 3^5 and M_PI which is equivalent to pi.
Exercise:
Compute the volumes of: a cube, sphere, cone.
Hint:
$V_{cube} = side^3$
$V_{Sphere} = (4/3) * \pi * radius^3$
$V_{Cone} = \pi * radius^2 * \frac{height}{3}$

```
In [1]: float cubeSide = 3;
        float volCube = 0;
        volCube = std::pow(cubeSide, 3);
        std::cout<<"The volume of the cube is: "<<volCube;
```

```
The volume of the cube is: 27
```

```
In [2]: float volSphere = 0;
        float sphereRadius = 1;
        volSphere = (4.0/3.0) * M_PI * std::pow(sphereRadius, 3);
        std::cout<<"The volume of the sphere is: "<<volSphere;
```

```
The volume of the sphere is: 4.18879
```

```
In [3]: float coneRadius = 1;
        float coneHeight = 5;
        float volCone = 0;
        volCone = M_PI * std::pow(coneRadius, 2) * (coneHeight/3);
        std::cout<<"The volume of the cone is: "<<volCone;
```

```
The volume of the cone is: 5.23599
```

2 Variable assignment

C++ is a language that requires variable types to be known at compile time.

But, C++ does allow some implicit conversions, for example an integer can be assigned to a float or an integer can be treated as a char.

```
In [4]: int intA = 10;
        std::cout<<"intA = "<<intA;
```

```
intA = 10
```

```
In [5]: float floatA = 10.0;
        std::cout<<"floatA = "<<floatA;
```

```
floatA = 10
```

```
In [6]: char charA = 10;
        std::cout<<"charA = "<<charA;
```

```
charA = 

```

```
In [7]: char charB = 'B';
        std::cout<<"charB = "<<charB;
```

```
charB = B
```

We can assign an integer to a float

```
In [8]: float floatB = 0.0;
        floatB = intA;
        std::cout<<"integer = "<<intA<<"\n\n";
        std::cout<<"floatNumber = integer = "<<floatB;
```

```
integer = 10

floatNumber = integer = 10
```

We can assign a char to a float

```
In [9]: floatB = charB;
        std::cout<<"floatB = charB = "<<floatB;
```

```
floatB = charB = 66
```

```
In [10]: float floatC = 0;
         floatC = floatB/4;
         std::cout<<"floatC = floatB/4 = "<<floatC;
```

```
floatC = floatB/4 = 16.5
```

But assigning a float to a char doesn't quite work

```
In [11]: char charC = "C";
         charC = floatC;
         std::cout<<"charC = floatC = "<<charC;
```

```
input_line_17:2:7: error: cannot initialize a variable of type 'char' with an lvalue of type
 char charC = "C";
      ^       ~~~


        Interpreter Error:
```

Assigning a float to an interger, results in the float being truncated

```
In [12]: intA = floatC;
         std::cout<<"intA = floatC = "<<intA;
```

```
intA = floatC = 16
```

Exercise 1:
What is the output of this program?

```
In [13]: float width = 4.5;
         float height = 5.5;

         int area = width * height;
```

Answer 1:

```
In [14]: std::cout << "area = " << area;
```

```
area = 24
```

Exercise 1:
What is the output of this program?

```
In [15]: int numerator = 4;
         int denominator = 5;

         int answer = numerator / denominator;
```

Answer 1:

```
In [16]: std::cout<<"answer = "<<answer;
```

```
answer = 0
```

2.1 Prefix and Postfix

Incrementing

- prefix: ++a
- postfix: a++

Decrementing

- prefix: –a
- postfix: a–

The difference between prefix and postfix is subtle, but crucial.

Prefix operators increment the value of the variable, then return the reference to the variable.

Postfix operators create a copy of the variable and increments the value of the variable. Then it returns a copy from **BEFORE** the increment.

```
In [17]: int a, b = 0;
         int post, pre = 0;
         std::cout<<"Inital values: \t\t\tpost = "<<post<<" pre= "<<pre;
```

```
Inital values:                         post = 0 pre= 0
```

```
In [18]: post = a++;
         pre = ++b;
         std::cout<<"After one postfix and prefix: \tpost = "<<post<<" pre= "<<pre;
```

```
After one postfix and prefix:         post = 0 pre= 1
```

```
In [19]: post = a++;
         pre = ++b;
         std::cout<<"After two postfix and prefix: \tpost = "<<post<<" pre= "<<pre;
```

```
After two postfix and prefix:         post = 1 pre= 2
```

2.2 Variable Assignment Operators

The table below shows the most often used assignment operators.

Operator	Example	Equivalence
+=	A += B	A = A + B
-=	A -= B	A = A - B
=	A *= B	A = A * B
/=	A /= B	A = A / B
%=	A %= B	A = A % B

```
In [20]: int add = 0;
         std::cout<<"Variable\t\tOperation\tResult\n";
         std::cout<<"add = "<<add;
         add += 5;
         std::cout<<"\t\t\tadd += 5 \t\t add = "<<add<<"\n";
```

```
Variable                Operation        Result
add = 0                        add += 5                  add = 5
```

```
In [21]: int sub = 0;
         std::cout<<"Variable\t\tOperation\tResult\n";
         std::cout<<"sub = "<<sub;
         sub -= 5;
         std::cout<<"\t\t\tsub -= 5 \t\t sub = "<<sub<<"\n";
```

```
Variable                Operation        Result
sub = 0                        sub -= 5                  sub = -5
```

```
In [22]: int mul = 1;
         std::cout<<"Variable\t\tOperation\tResult\n";
         std::cout<<"mul = "<<mul;
         mul *= 5;
         std::cout<<"\t\t\tmul *= 5 \t\t mul = "<<mul<<"\n";
```

```
Variable                Operation        Result
mul = 1                        mul *= 5                  mul = 5
```

```
In [23]: float divi = 1;
         std::cout<<"Variable\t\tOperation\tResult\n";
         std::cout<<"divi = "<<divi;
         divi /= 0.5;
         std::cout<<"\t\t\tdivi /= 5 \t\t divi = "<<divi<<"\n";
```

```
Variable                Operation        Result
divi = 1                        divi /= 5                  divi = 2
```

```
In [25]: int mod = 5;
         std::cout<<"Variable\t\tOperation\tResult\n";
         std::cout<<"mod = "<<mod;
         mod %= 2;
         std::cout<<"\t\t\tmod %= 2 \t\t mod = "<<mod<<"\n";
```

```
Variable                Operation        Result
mod = 5                        mod %= 2                  mod = 1
```