3. Expressions

4. Expressions and Arithmetic

Expressions (1)

- Operator and operand
  - 5 + 10   **// binary**
  - -x   **// unary**
  - 5 + 10/2   **// precedence**
  - a = b = 2   **// associativity**

- Arithmetic operators
  - +, -, *, /, %(modulus)

- std::cin
  - Standard input stream
  - >>: extraction operator

  - int x;
  - std::cin >> x;

Expressions (2)

```cpp
#include <iostream>
int main() {
    int value1, value2, sum;
    std::cout << "Please enter two integer values: ";
    std::cin >> value1 >> value2;
    sum = value1 + value2;
    std::cout << value1 << " + " << value2 << " = " << sum << '\n';

    std::cout << 10%3 << " " << 3%10 << '\n';

    char lower = 'd', upper = lower - 32;
    // upper = lower - ('a' - 'A');
    std::cout << upper << '\n';
}

// sum = value1 + value2;
// value1 + value2
// sum = (value1 + value2)
```
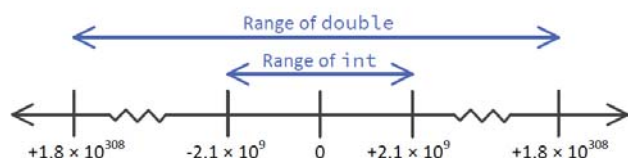
Mixed Type Expressions

```cpp
int x = 4;
double y = 10.2, sum;
int sum2;


sum = x + y;
sum2 = sum;       // warning C4244, sum2 = static_cast<int>(sum);
                  // (type) cast


enum class Shade { Dark, Dim, Light, Bright };
Shade color = Shade::Light;

std::cout << (int)color << std::endl;
std::cout << static_cast<int>(color) << std::endl;
```

Operator Precedence and Associativity

- Precedence
    - When an expression contains two different kinds of operators, which should be applied first?
- Associativity
    - When an expression contains two operators with the same precedence, which should be applied first?

Arity	Operator	Associativity
Unary	+, -	
Binary	*, /, %	Left
Binary	+, -	Left
Binary	=	Right

The operators in each row have a higher precedence than the operators below it.

Comments

- Comment
    - Annotation
    - Ignored by compilers and interpreters
- Single line comment
    - //
- Block comment
    - /*...*/

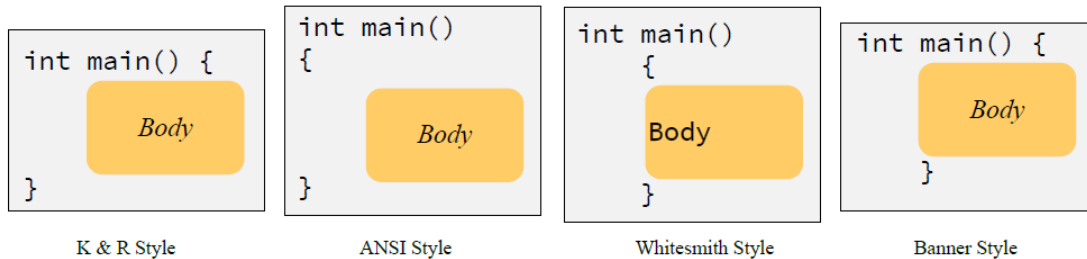# Formatting (1)

```
#include <iostream>
int
main
(){
int
x;x=
10
;
std
::
cout
<<
x
<<
'\n'
;}
```

```
#include <iostream>
int main(){int
x;x=10;std::cout<<x<<'\n';}
```

```
#include <iostream>
int main(){
    int x;
    x = 10;
    std::cout<<x<<'\n';
}
```

Formatting (2)


```
int main() {

    Body

}
```
K & R Style

```
int main()
{

    Body

}
```
ANSI Style

```
int main()
    {
    Body
    }
```
Whitesmith Style

```
int main() {

    Body

    }
```
Banner Style

Errors and Warnings (1)

- Compile-time error
  - Syntax error
  - Link error
- Run-time error
  - Invalid memory access
  - Memory leak
  - Division by zero
- Logic error (run-time error)
  - Wrong formula/output
  - Division by zero

Errors and Warnings (2)

- Compiler warnings
  - Uninitialized object (variable)
  - Narrowing conversion (`double` → `int`)

Arithmetic Examples (1)

```cpp
#include <iostream>
int main() {
   double degreesF, degreesC;
   // Prompt user for temperature to convert
   std::cout << "Enter the temperature in degrees F: ";
   // Read in the user's input
   std::cin >> degreesF;
   // Perform the conversion
   degreesC = 5/9*(degreesF - 32);
   // Report the result
   std::cout << degreesC << '\n';
}
```

Arithmetic Examples (2)

```cpp
#include <iostream>
int main() {
   int hours, minutes, seconds;
   std::cout << "Please enter the number of seconds:";
   std::cin >> seconds;

   hours = seconds / 3600;
   seconds = seconds % 3600;
   minutes = seconds / 60;
   seconds = seconds % 60;
   std::cout << hours << " hr, " << minutes << " min, "
       << seconds << " sec\n";
}
```

Arithmetic Examples (3)

```cpp
#include <iostream>
int main() {
    int hours, minutes, seconds;
    std::cout << "Please enter the number of seconds:";
    std::cin >> seconds;

    hours = seconds / 3600;
    seconds = seconds % 3600;
    minutes = seconds / 60;
    seconds = seconds % 60;
    std::cout << hours << ":";
    int tens = minutes / 10;
    std::cout << tens;
    int ones = minutes % 10;
    std::cout << ones << ":";
    tens = seconds / 10;
    std::cout << tens;
    ones = seconds % 10;
    std::cout << ones << '\n';
}
```
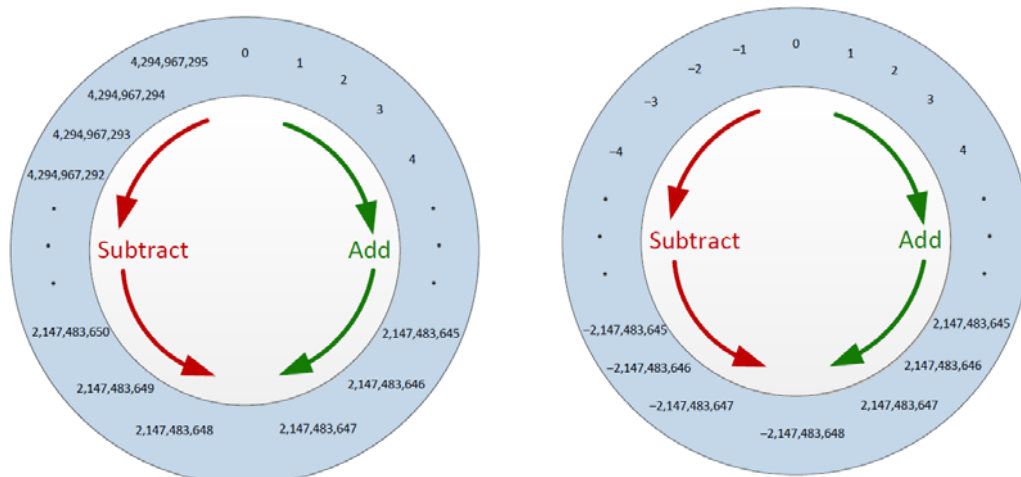
Integers vs. Floating-point Numbers (1)

- Computers store all data internally in binary form
- Integers
  - Overflow/underflow

Integers vs. Floating-point Numbers (2)

- Floating-point numbers
  - Sign, mantissa, exponent


```cpp
#include <iostream>
#include <iomanip>
int main() {
   double d1 = 2000.5;
   double d2 = 2000.0;
   std::cout << std::setprecision(16) << (d1 - d2) << '\n';

   double d3 = 2000.58;
   double d4 = 2000.0;
   std::cout << std::setprecision(16) << (d3 - d4) << '\n';
}
```

Integers vs. Floating-point Numbers (3)

```cpp
#include <iostream>
int main() {
  double one = 1.0, one_eighth = 1.0/8.0,
  zero = one - one_eighth - one_eighth - one_eighth
     - one_eighth - one_eighth - one_eighth - one_eighth - one_eighth;
  std::cout << "one = " << one << ", one_eighth = " << one_eighth
     << ", zero = " << zero << '\n';
}
```

```cpp
#include <iostream>
int main() {
   double one = 1.0, one_fifth = 1.0/5.0,
   zero = one - one_fifth - one_fifth - one_fifth - one_fifth
     - one_fifth;
   std::cout << "one = " << one << ", one_fifth = " << one_fifth
     << ", zero = " << zero << '\n';
}
```

More Arithmetic Operators (1)

- Increment/decrement
  - `++x, x++, --x, x—`
  - Prefix, postfix

```cpp
#include <iostream>
int main() {
   int x1 = 1, y1 = 10, x2 = 100, y2 = 1000;
   std::cout << "x1=" << x1 << ", y1=" << y1
      << ", x2=" << x2 << ", y2=" << y2 << '\n';
   y1 = x1++;
   std::cout << "x1=" << x1 << ", y1=" << y1
      << ", x2=" << x2 << ", y2=" << y2 << '\n';
   y2 = ++x2;
   std::cout << "x1=" << x1 << ", y1=" << y1
      << ", x2=" << x2 << ", y2=" << y2 << '\n';
}
```

More Arithmetic Operators (2)

- Assignment operators
  - `+=, -=, X=, /=, %=`
  - `x = 5;`
    `x += 2;`
    `x *= 4+6;`

Bitwise Operators

- Bitwise operators
  - `&`, bitwise and
  - `|`, bitwise or
  - `^`, bitwise exclusive or
  - `~`, bitwise negation, unary operator
  - `>>`, shift right
  - `<<`, shift left

  - `&=`, `|=`, `^=`, `>>=`, `<<=`

Algorithms

```cpp
#include <iostream>
int main() {
   double degreesF = 0, degreesC = 0;
   degreesC = 5.0/9*(degreesF - 32);
   std::cout << "Enter the temperature in degrees F: ";
   std::cin >> degreesF;
   std::cout << degreesC << '\n';
}
```

```cpp
#include <iostream>
int main() {
   int x, y, t;

   t = x;
   x = y;
   y = t;
}
```