The Zen of C++

Chapter 2: Fundamental Data Types

Adnan Zejnilovic

C++ Programming Process

- Write the source code
  - Notepad
  - IDEs
- Compile
  - Command Line
  - IDEs
- Execute

Compilation

Command Line Compilers

- GNU C++ (GCC)
- Pros: Free
- Cons:
  - Need to remember commands
  - Error prone due to typing

IDEs

- MS Visual Studio
- Code Blocks
- DevC++
- Pros:
  - No need to memorize commands – only a few buttons
- Cons:
  - Each IDE is specific
  - Learning curve

Your 1st C++ Program

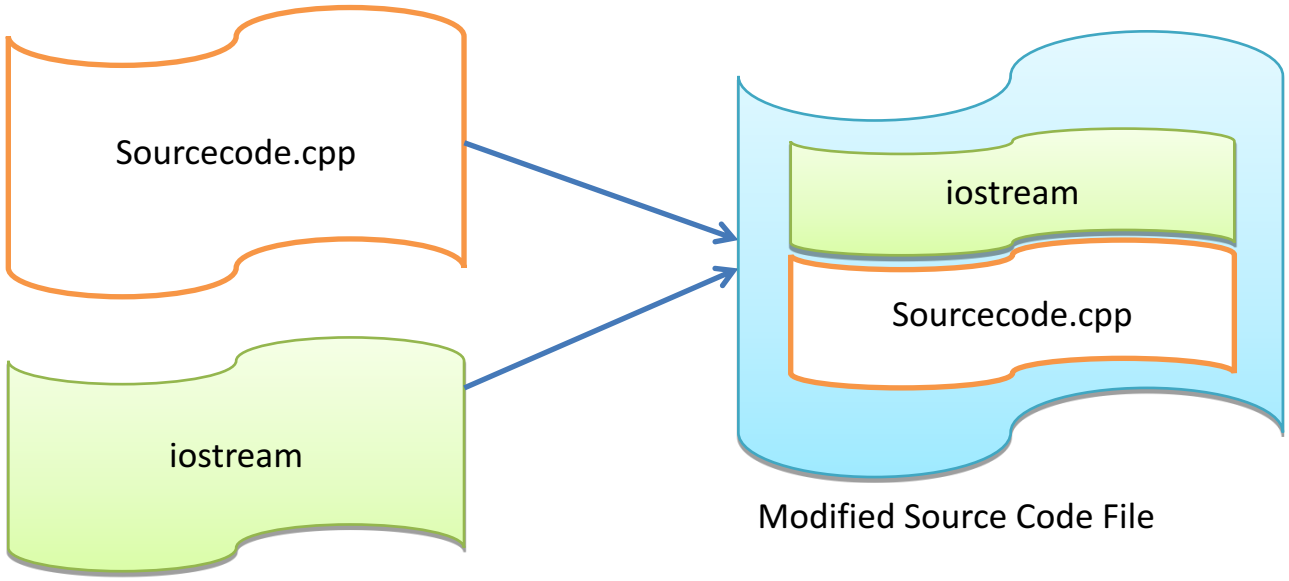
```
1   // my first C++ program -Hello World
2   #include <iostream>
3
4   using namespace std;
5
6   int main()
7   {
8       cout << "Hello world!" << endl;
9       return 0;
10  }
```

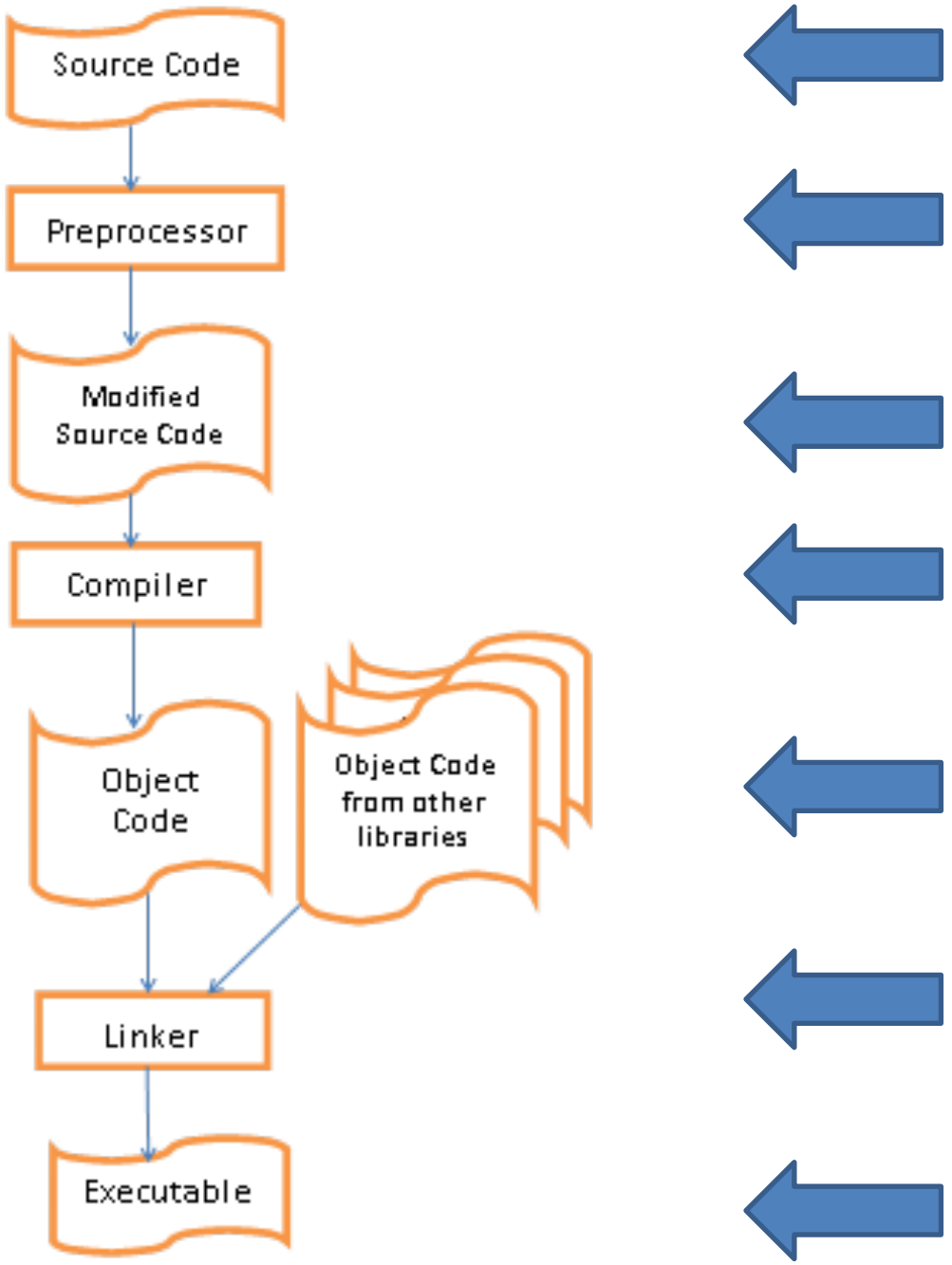
```
1   /*
2       Programmer    : Adi Zejnilovic
3       Date          : 07/28/2013
4       Description   : This program will display the following
    message:

5                       Hello world!
6   */
7   #include <iostream>
8
9   using namespace std;
10
11  int main()
12  {
13      cout << "Hello world!" << endl;
14      return 0;
15  }
```

Preprocessor Directive

```
1   // my first C++ program -Hello World
2   #include <iostream>
3
4   using namespace std;
5
6   int main()
7   {
8       cout << "Hello world!" << endl;
9       return 0;
10  }
```



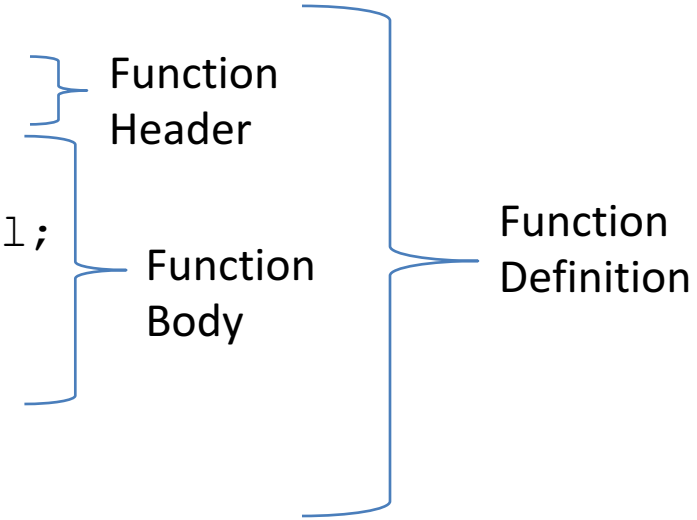

Source Code
Preprocessor
Modified Source Code
Compiler
Object Code
Object Code from other libraries
Linker
Executable

Namespaces

- using namespace std;
- namespace is like a directory
  - std::cout << "Hello world!" << std::endl;
  - cout << "Hello world!" << endl;

Function Definition, Function Header, and Function Body

```
int main()
{
    cout << "Hello World!" << endl;
    return 0;
}
```



int main cont.

```
int main()
{
    cout << "Hello World!" << endl;
    return 0;
}
```

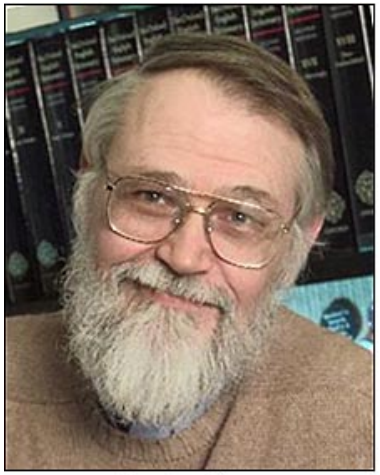
endl and Escape Sequences

Escape Sequence	Character	Comments
\’	Single quote	Prints a single quote
\”	Double quote	Prints a double quote
\\	Backslash	Prints a backslash
\0	Null character	Null Character (to terminate char arrays)
\b	Backspace	Moves the cursor back one space
\a	Audible bell	Plays a predetermined sound
\f	Form feed (new page)	Prints a new page
\n	Line feed (new line)	Moves the cursor to the new line
\r	Carriage return	Moves the cursor to the beginning of the existing line
\t	Horizontal tab	Moves the cursor to the next tab stop

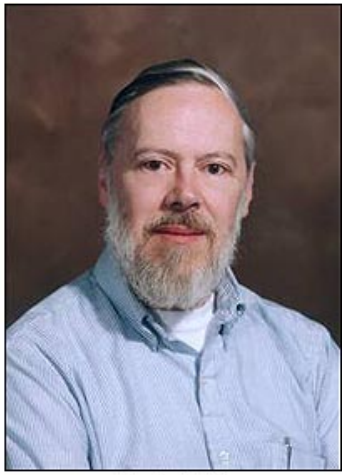
Programming Style – K & R

```
$ indent -kr source.c
$ cat source.c
#include <stdio.h>

int main()
{
    if (3 == 3) {
        printf("Yes\n");
    } else {
        printf("Neah\n");
    }
    return 0;
}
$
```

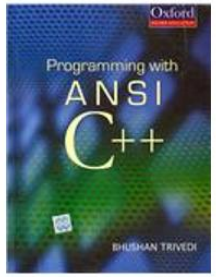


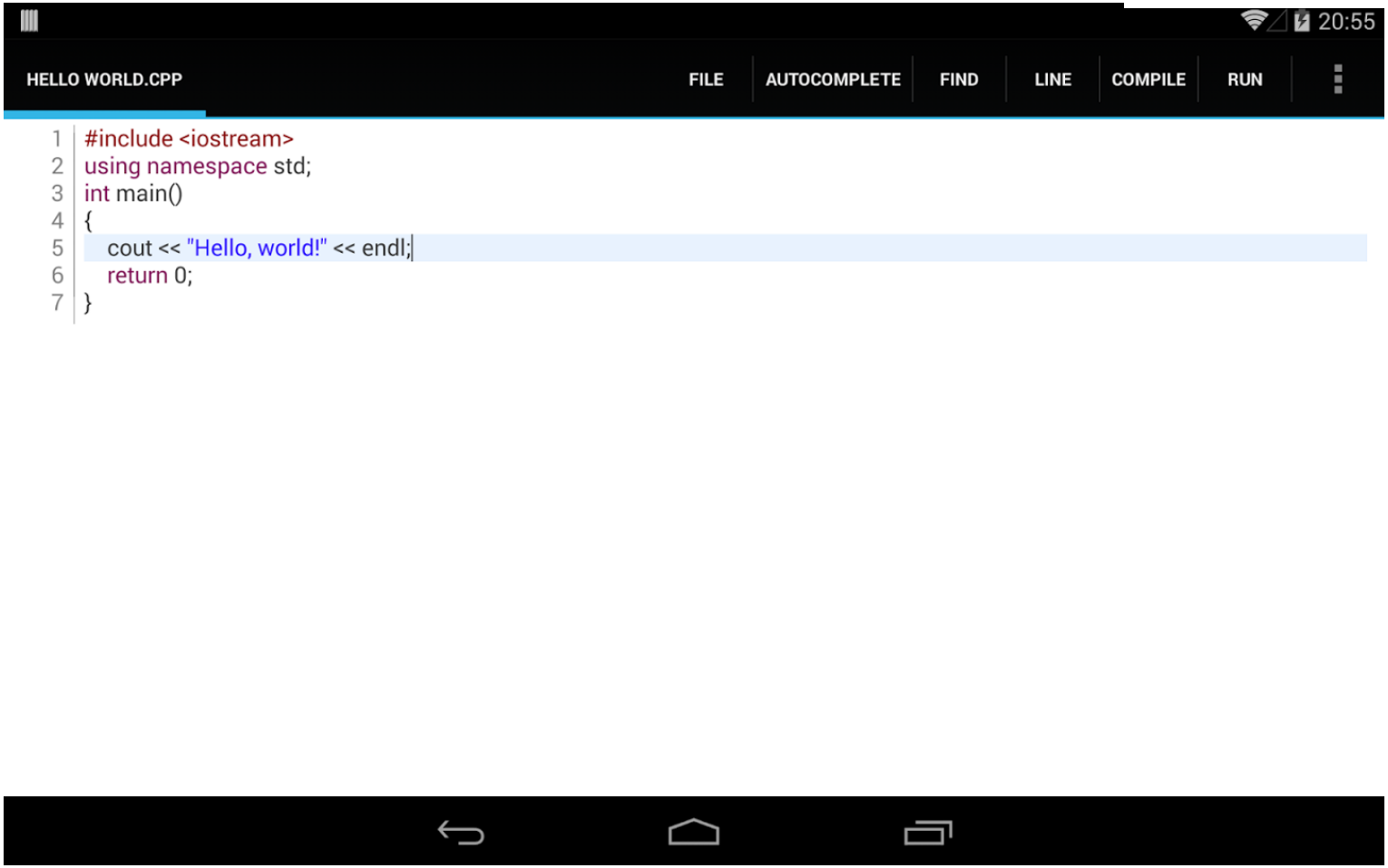
Brian Kernighan



Dennis Ritchie

Programming Style





Programming Style - GNU

```
if (a > 5)
  {
    // statement(s);
  }
```

Fundamental Data Types

- Reserved (“Key Words”) Bool, int, float, for, if, namespace
- Identifiers

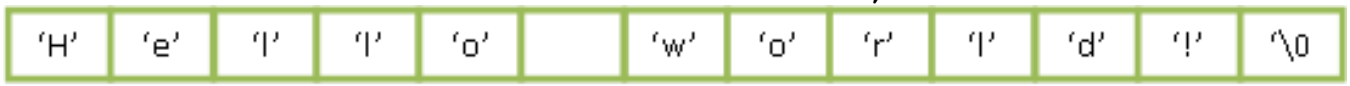
  constants, variables, types, templates, classes, functions, and namespaces

  - Must start with a letter or an underscore
  - Can contain characters and numbers
  - Cannot start with a number
- Operators
- Punctuation
- Grammar

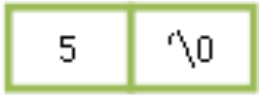
Literals

String Literal

```
cout << "Hello world!" << endl;
```




```
cout << "5" << endl;
```

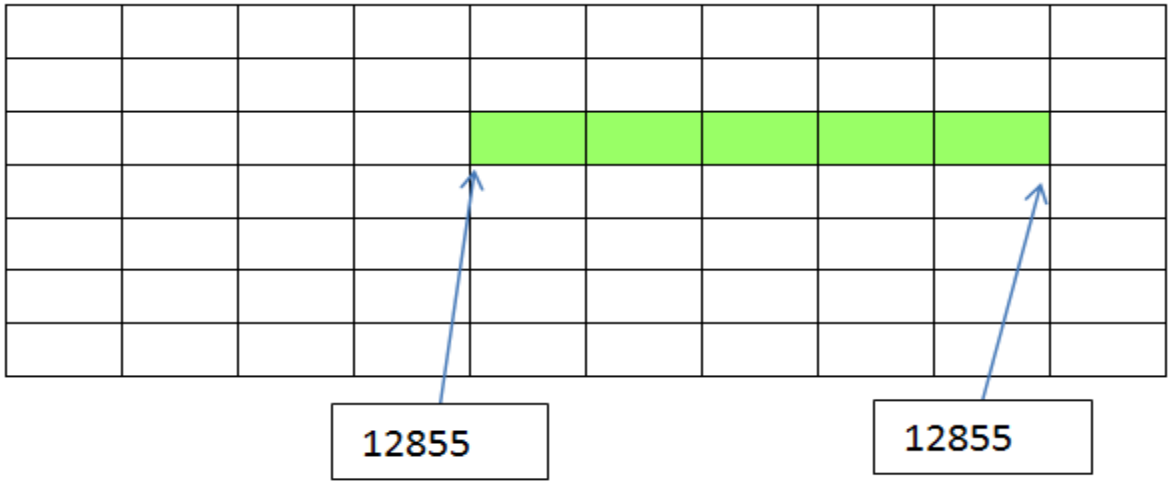


Integer Literal

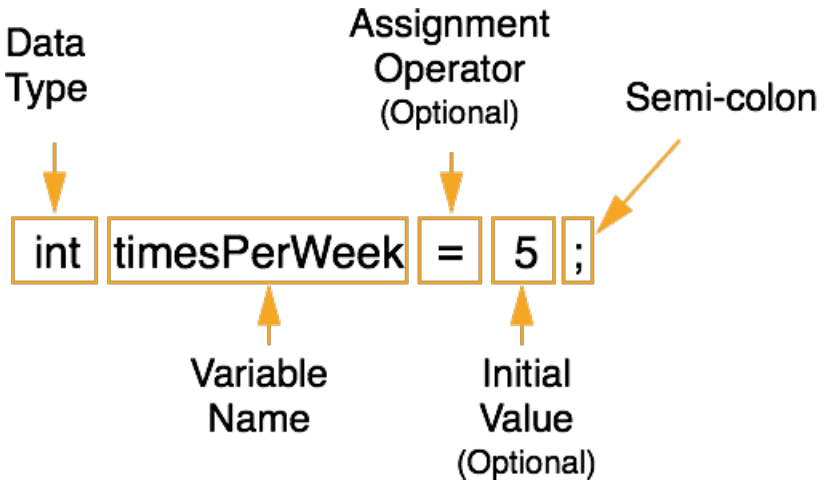
```
cout << 5 << endl;
```

Variables

- int x; // variable definition



Variables


Data
Type
Assignment
Operator
(Optional)
Semi-colon
int
timesPerWeek
=
5
;
Variable
Name
Initial
Value
(Optional)

C++ Data Types

Built-in

- Fundamental Data Types
- Non Fundamental Data Types

Programmer Defined

- Abstract Data Types

Built In Data Types

Integer Data Types

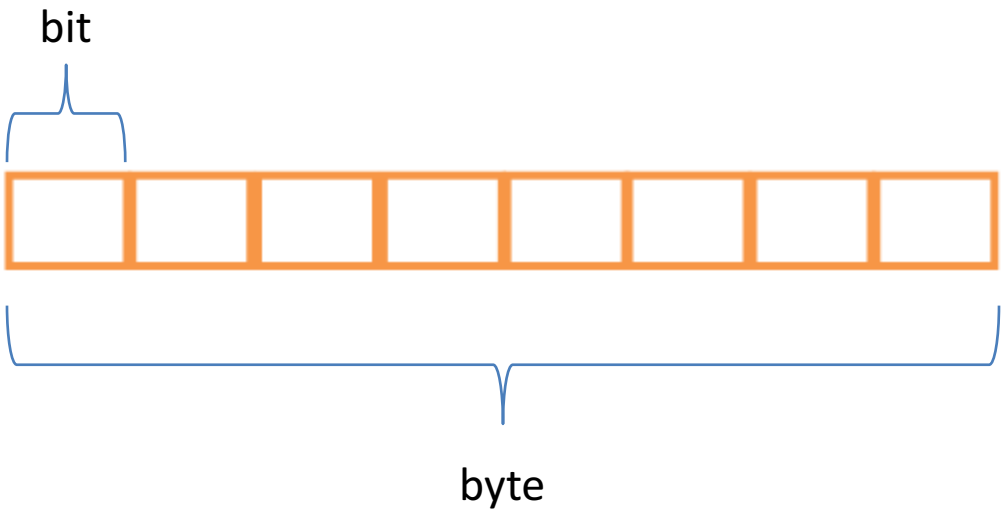
- bool
- char
- short int (also short)
- int
- long int (also long)
- unsigned char
- unsigned short int
- unsigned int
- unsigned long

Floating Point Data Types

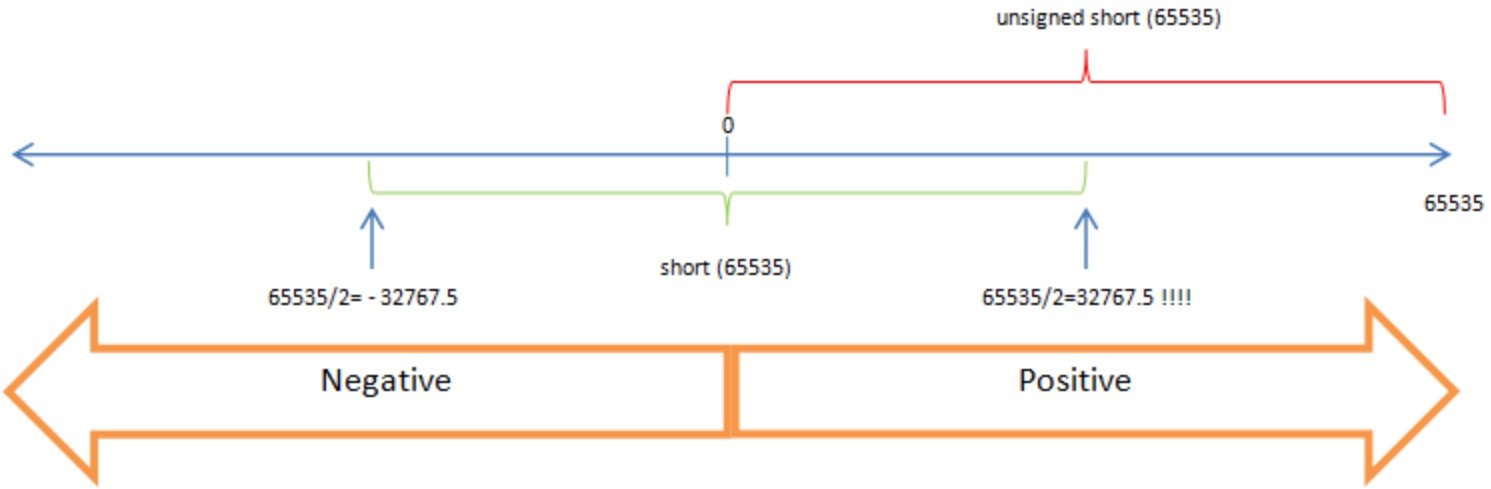
- Number with decimal point

What is the difference between signed and unsigned?

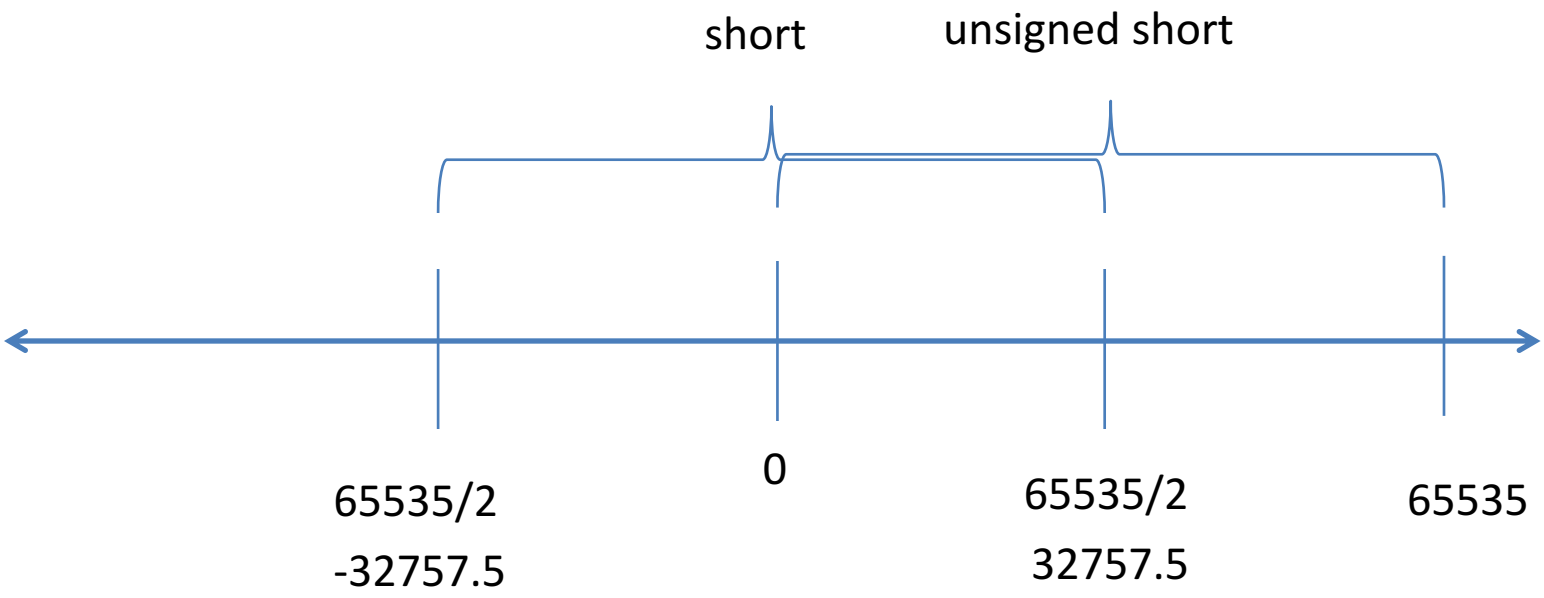
Bit vs. Byte



Signed vs. Unsigned



Signed vs. Unsigned



Source Code

```
#include <iostream>
#include <climits>

using namespace std;

int main()
{
    cout << "sizeof bool: " << sizeof(bool)<< endl;
    cout << "bool range: "
         << CHAR_MIN
         << " to "
         << CHAR_MAX << endl;
    return 0;
}
```

short vs. int

Fundamental Data Types cont.

- bool
- char
- Floating Point
  - float
  - double
  - long double

Non-Fundamental Data Types

```
#include <string>
string bestEver = “Michael Jordan”;
string myAddress = “123 Main Street”;
```

Arithmetic Operators

- Unary
  - Example: -10
- Binary

Operator	C++ Symbol	Example
Addition	+	5 + 3
Subtraction	-	5.4 – 3
Multiplication	*	5 * 3.14
Division	/	5.0 / 3.0
Modulus	%	5 % 3

- Ternary
  - Example: ((x+5) > 0) ? x = x + 1 : x = 10;

Integer Arithmetic

```
int main()
{
    cout << "5 / 3 = " << 5/3 << endl;
    return 0;
}
```

$\frac{5}{3} = 1.666666666666667$

Integer Arithmetic

```
int main()
{
    cout << "5 / 3 = " << 5/3.0 << endl;
    return 0;
}
```

$$\frac{5}{3} = 1.666666666666667$$

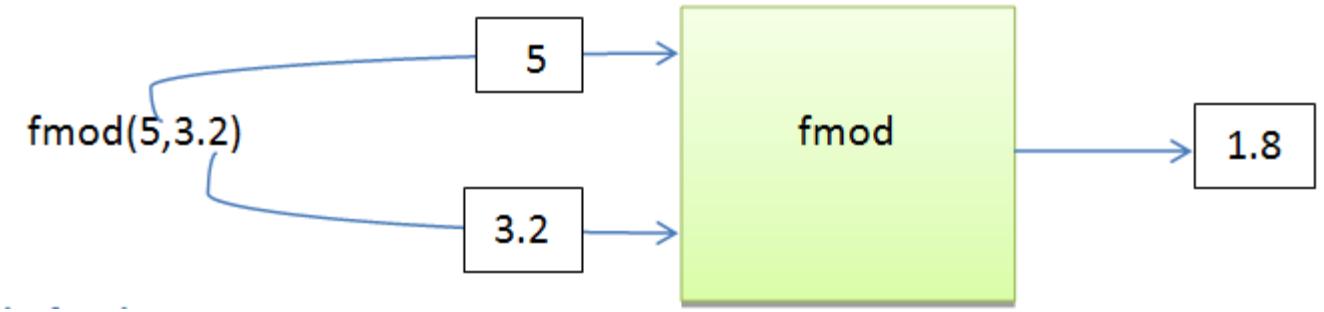
```
5 / 3 = 1.66667

Process returned 0 (0x0)   execution time : 0.032 s
Press any key to continue.
```

Integer Arithmetic

```
1   #include <iostream>
2   #include <cmath>                                  // required for fmod
3
4   using namespace std;
5
6   int main()
7   {
8       cout << "5 / 1 = " << 5/3 << endl;          // integer / integer = integer
9       cout << "5 / 3.0 = " << 5/3.0 << endl;      // integer / double  =  double
10
11      // modulus
12      cout << "5 % 3 = " << 5 % 3 << endl;        // integer % integer = integer
13      cout << "fmod(5,3.2) = " << fmod (5,3.2); // fmod(integer,double)  = double
14
15      return 0;
16  }
```

Integer Arithmetic



Operator Precedence

3 + 5 * 10 – 20 / 2 + 5 % 2 = 44

- How did I compute that?

3	+	5	*	10	-	20	/	2	+	5	%	2
3	+	50			-	10			+	1		
44												

Operator Precedence
Associativity

- (3 + 5) * (10 – 20) / 2 + 5 % 2

(3+5)	*	(10-20)	/	2	+	5	%	2
8	*	-10	/	2	+	1		
-80			/	2	+	1		
-40					+	1		
-39								

Named Constants

```
const dataType constantName = value;
```

Example:

```
const double QUIZ_CAT_WT = 0.60;
const double MIDTERM_CAT_WT = 0.10;
const double FINAL_CAT_WT = 0.20;
```

Named Constants
Legacy Directives

```
#define QUIZ_CAT_WT 0.60
```

```
#define QUIZ_CAT_WT 0.60;
#define MIDTERM_CAT_WT 0.10
#define FINAL_CAT_WT 0.20
```

```
score =  quizAvg * 0.60; + midTermExam *
0.10 + finalExam * FINAL_CAT_WT;
```

Overflow and Underflow

```
short myUpperNumber = 32767;
myUpperNumber = myUpperNumber + 1;
cout << myUpperNumber;
```