APS105
Winter 2012

Jonathan Deber
jdeber -at- cs -dot- toronto -dot- edu

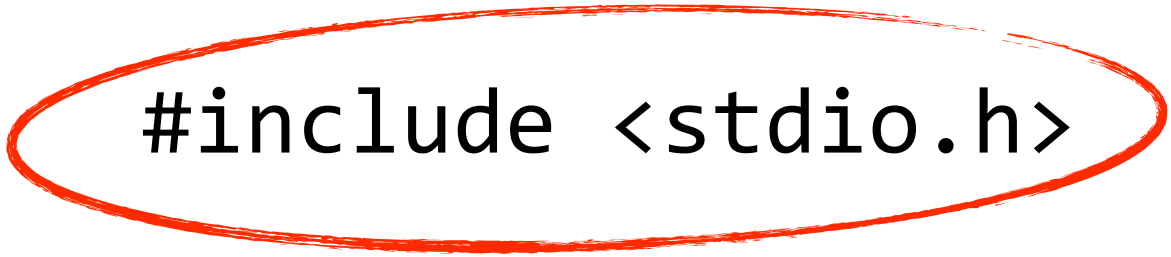
Lecture 6
January 27, 2012

Today

- Memory Representation
- Booleans
- `if` Statements

- Tutorial room changed to WB 219

#include vs. #import

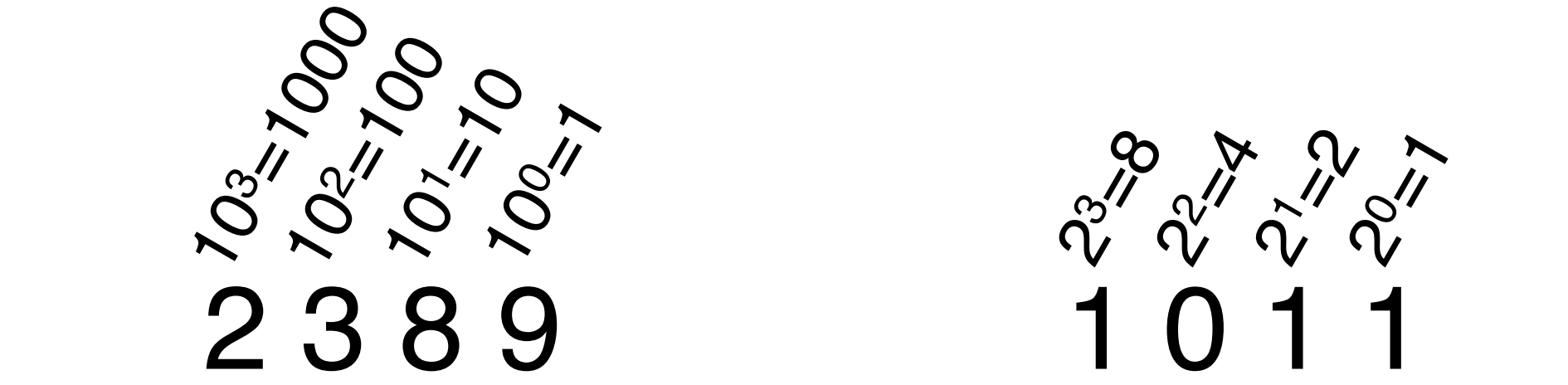



```
#import <stdio.h>
```

Bits and Bytes

Bits

- Basic unit of information is a bit (binary integer)
- 0 or 1, easy mapping to various physical states
- This is a base 2 number system

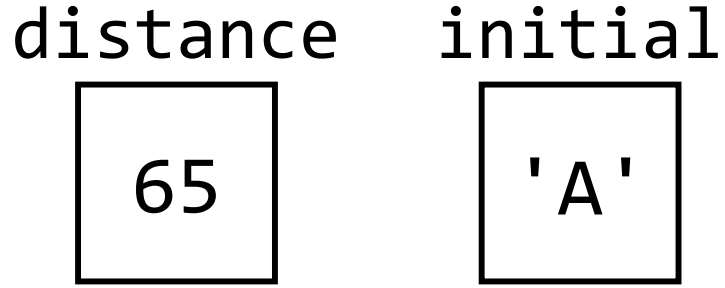


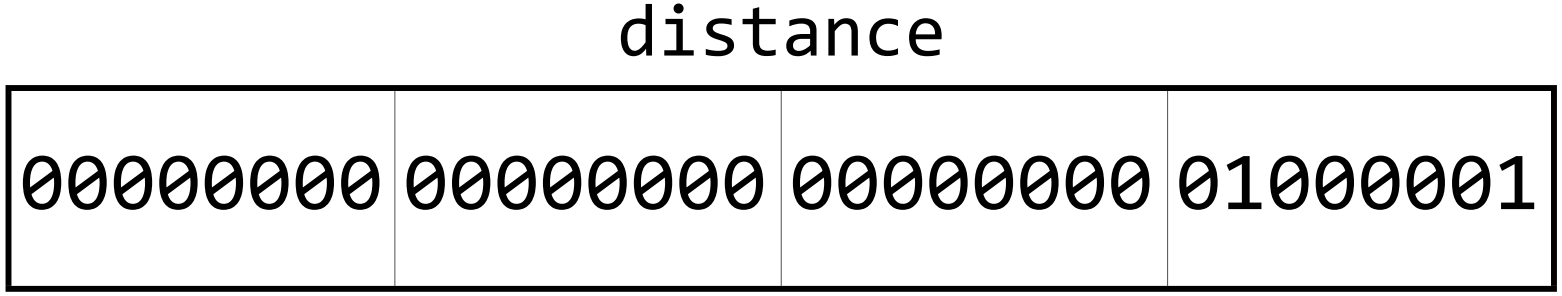
Bytes

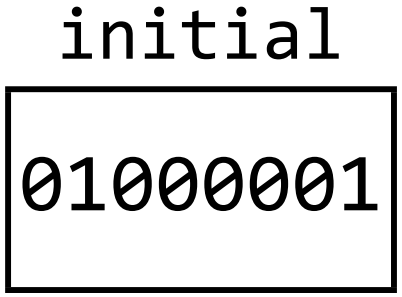
- Two numbers isn't enough for most things
- Group bits into larger chunks called bytes
- 1 byte = 8 bits = 2^8 = 256 possible values
- b is bit, B is byte
- 1KB = 1000 bytes, right?
- Unfortunately, no. It's actually 1024 bytes
- And 1MB = 1024 KB, 1GB = 1024 MB, *etc.*

Memory Representation

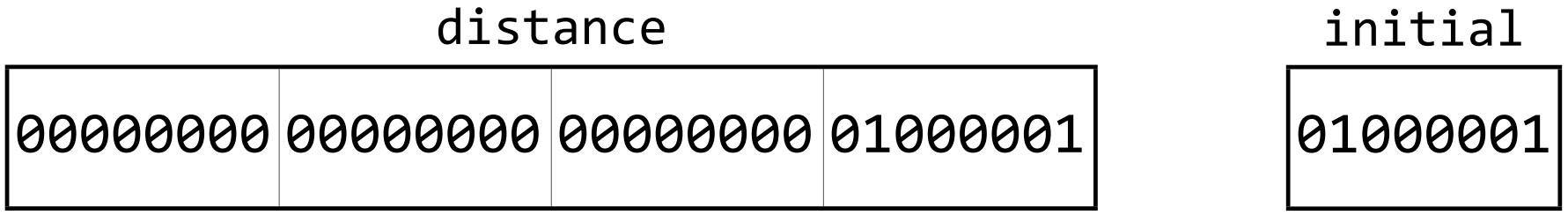
```
int distance = 65;
char initial = 'A';
```







Memory Representation




```
printf("distance as int: %d\n", distance);
printf("initial as char: %c\n", initial);

printf("distance as char: %c\n", distance);
printf("initial as int:   %d\n", initial);

printf("distance as double: %g\n", distance);
printf("initial as double:  %g\n", initial);
```

65

A

A

65

?

Booleans

Decisions

- Many (or most) algorithms have notion of decisions
  - "If the total is more than $100, then shipping is free"
  - "If the cake is golden brown, then it is ready to be taken out of the oven"

- If not, they do the same thing every time

True and False

- Implicit idea of "true" and "false"
  - If it is true that "the total is more than $100", then decide that "shipping is free"
  - If it is true that "the cake is golden brown", then decide that "it is ready to be taken out of the oven"
  - If it is false that "the total is more than $100", then decide that "shipping is not free"

Booleans

- George Boole (1815-1864)
- A variable that can hold two things
  - `true`
  - `false`
- `C` type is `bool`
- Need to use `stdbool` library
  - `#include <stdbool.h>`

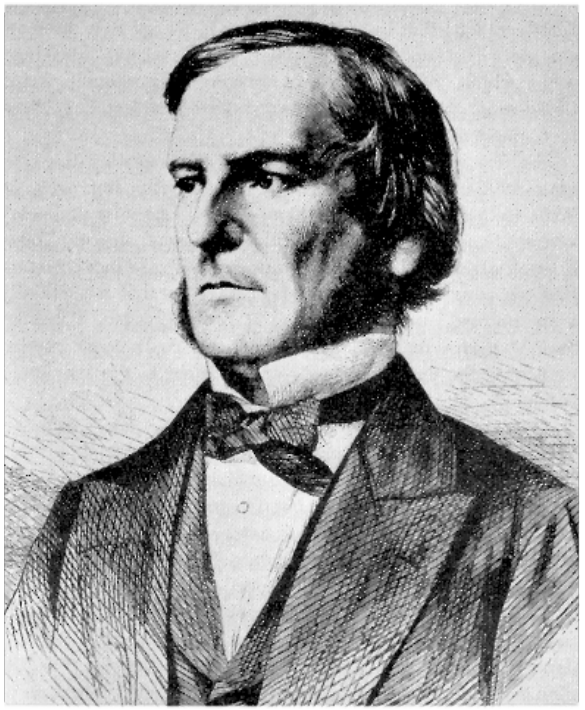


Image: Public Domain, via Wikimedia Commons

Literals

- `bool isSnowing = false;`
- `bool newDataWaiting = true;`

- Convention: "is" prefix often used
- Terminology: "flag"

Relational Operators

- Used to compare two values

Symbol	*Meaning*
<	less than
>	greater than
<=	less than or equal to (≤)
>=	greater than or equal to (≥)

Relational Operators

```
bool b = (7 < 2);
```
false

```
bool c = (9 >= 8);
```
true

```
bool d = (3.4 > 2);
```
true

```
int sum = 20;
bool isOver = (sum >= MAX_TOTAL);
```

Precedence

- Lower than arithmetic operators

```
i + 5 < j - 2
(i + 5) < (j - 2)
```

- Higher than assignment

```
bool b = 7 < 2;
bool b = (7 < 2);
```

Chaining

- This will not work:

```
int value = 12;
bool inRange = (4 <= value < 9);
```

Equality Operators

- Also used to compare two values

Warning!
= vs ==

Symbol	*Meaning*
==	equal to
!=	not equal to

```
bool b = 7 == 2;
bool c = (9 != 8);

bool d = 3.4 != 2;

int sum = 20;
bool isEqual = (sum == MAX_TOTAL);
```

false

true

true

Comparing chars

```
bool b = ('A' == 'A');
```
true (65 == 65)

```
bool c = ('D' >= 'F');
```
false (68 >= 70)

Not the best way to do this

```
bool d = ('0' == 0);
```
false (48 == 0)

```
bool e = ('\0' == 0);
```
true (0 == 0)

Logical Operators

- Used to reason about boolean values
- C has three operators

Symbol	*Meaning*
!	logical not
||	logical or
&&	logical and

Logical not (!)

- Unary operator that negates its operand

```
bool b = !true;
b = !(8 <= 4);

isOpen = !isOpen;
```

```
false
```

```
true
```

Formal Definitions

a && b is true iff (a is true) and (b is true)

a || b is true iff (a is true) or (b is true)

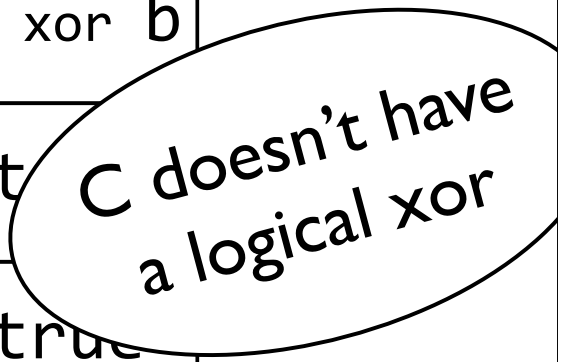
a	b	a && b	a || b	a xor b
false	true	false	true	true
true	false	false	true	true
true	true	true	true	false
false	false	false	false	false

Formal Definitions

a && b is true iff (a is true) and (b is true)

a || b is true iff (a is true) or (b is true)

a	b	a && b	a || b	a xor b
false	true	false	true	t
true	false	false	true	tru
true	true	true	true	false
false	false	false	false	false



Examples

```
bool b = true && false;
bool isBadWeather = isRaining || isSnowing;


int value = 12;
bool inRange = (4 <= value < 9);
```

Wrong

```
bool inRange = (4 <= value) && (value < 9);
```

DIY xor

a	b	a xor b
false	true	true
true	false	true
true	true	false
false	false	false

(!a && b)

"a is false" and "b is true"

"a is true" and "b is false"

(a && !b)

a xor b is the same as (!a && b) || (a && !b)

`if` statements

if statements

```
bool b = true;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
```

```
Is it true?
Yes it is.
```

```
b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
```

```
Is it true?
```

`if` statement

```
if (expression)
  statement
```

expression

```
if (true)
   statement
```

```
if (a && b)
   statement
```

```
if (weight == mass)
   statement
```

```
if ((i % 2) == 0))
   statement
```

```
if (orderTotal >= FREE_SHIPPING)
   statement
```

```
if( (4 <= value) && (value < 9) )
   statement
```

statement

```
bool b = true;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");


b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
```

statement

```
bool b = true;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
   printf("Definitely is.\n");

b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
   printf("Definitely is.\n");
```

```
Is it true?
Yes it is.
Definitely is.
```

```
Is it true?
Definitely is.
```

`if` statement

```
if (expression)
  statement
```

`if` statement

```
if (expression)
  statements
```

`if` statement

```
if (expression)
  statement
```

if statement

```
if (expression)
   statement
```

```
b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
   printf("Definitely is.\n");
```

`if` statement

```
if (expression)
   statement
```

```
b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
printf("Definitely is.\n");
```

Compound Statements

```
int main(void)
{
   ...
}
```

Compound Statements

```
b = false;

printf("Is it true?\n");
if (b)
   printf("Yes it is.\n");
   printf("Definitely is.\n");
```

Compound Statements

```
b = false;

printf("Is it true?\n");
if (b)
{
   printf("Yes it is.\n");
   printf("Definitely is.\n");
}
```

if/else

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);
}
printf("%d is odd\n", num);
```

if/else

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);
}
else
{
   printf("%d is odd\n", num);
}
```

Nested if

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);

   if ( (num % 3) == 0 )
   {
      printf("it's divisible by 3\n");
   }
}
```

num: 2

```
2 is even
```

num: 6

```
6 is even
it's divisible by 3
```

num: 9

Cascaded if

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);

}
if ( (num % 3) == 0 )
{
   printf("it's divisible by 3\n");
}
```

num: 2

```
2 is even
```

num: 6

```
6 is even
it's divisible by 3
```

num: 9

```
it's divisible by 3
```

Cascaded if

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);

}
else
{
   if ( (num % 3) == 0 )
   {
      printf("it's divisible by 3\n");
   }
}
```

Cascaded if

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);

}
else if ( (num % 3) == 0 )
{
   printf("it's divisible by 3\n");
}
```

Cascaded if

```
if ( (num % 2) == 0 )
{
   printf("%d is even\n", num);

}
else if ( (num % 3) == 0 )
{
   printf("it's divisible by 3\n");
}
```

num: 2

```
2 is even
```

num: 6

```
6 is even
```

num: 9

```
it's divisible by 3
```

```
if ( (num % 2) == 0 )
{
  printf("%d is even\n", num);
}
else if ( (num % 3) == 0 )
{
  printf("%d is divisible by 3\n", num);
}
else if ( (num % 5) == 0 )
{
  printf("%d is divisible by 5\n", num);
}
else
{
  printf("%d not divisible by 2,3,or 5\n", num);
}
```