News…

TA office hours

Piazza

Look at survey results

Talk about Packages and VSCode and –classpath
Survey:

- Prefer live lectures to flipped (will do only a few flipped)
- Difficulty about right, but labs are difficult
- Have labs due Monday night, not Sunday (sure)

CS112 –
Java
Programming

Spring 2024

Number Formats and Binary

Integer Variables and Two's complement

What is…?

What is "decimal"?

- 735,624

What is "binary"?

- 0111 1010

Is there a "unary"? Yes!

- $1111111_1 = 1^6 + 1^5 + 1^4 + 1^3 + 1^2 + 1^1 + 1^0 = 7_{10}$

WALK THRU ALL EXAMPLES SLOWLY

Integer Data Types – "Two's Complement" Format

Represented in **binary**

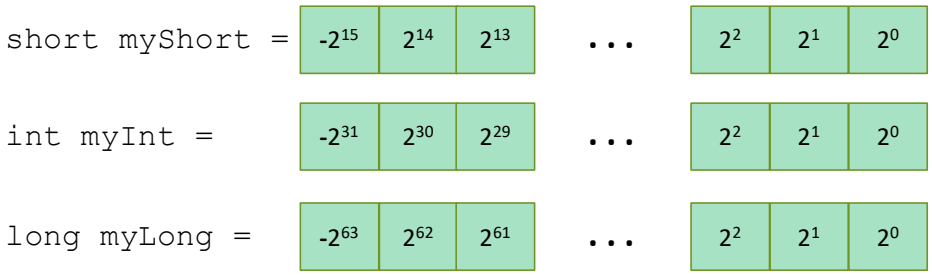
`byte var1 =`

Bit Pattern	Value
0000 0000	0
0000 0001	1
0000 0011	2 + 1 = 3
0100 0000	2^6 = 64
1000 0000	-2^7 = -128
0111 1111	127
1111 1111	-128 + 127 = -1

This shows a **byte** : 8 bits
This structure lets us use all familiar rules of addition, subtraction, multiplication, and division.
CPUs have transistor circuits built in to perform these operations directly in HW

Integer Data Types – shorts, ints, longs

Always only the most significant bit is negative



How about **`char`**?

Considered an "integer type"

It is unsigned

Can do all the same operations as with other types

Can cast to/from other types, automatically to int and long

(Of course), printing chars is different—we get the Unicode character whose UTF-16 code is the variable's value…not the value itself

```
byte x = (byte) 65;
char c = (char) x;
System.out.println(x + "  " + c);
```

Arithmetic Operations

Some gotchas!

Arithmetic underflow

```
byte b = 127;
b++;
System.out.println(b);

byte d = -128;
d--;
System.out.println(d);
```

No warning from computer—if a risk in your use, you must check!

Binary Operators

Binary Data Manipulation

Numbers (and in fact everything) stored in computers are stored in binary format

Our programming language manages the binary data so we can see Strings, Arrays, etc

Can we manipulate the binary directly?

Binary Data Manipulation

We can manipulate the bits in "integer" data types: "bit by bit" operations

What operations?
1) Same as for boolean
- AND: &
- OR: |
- NOT: ~
- XOR: ^

2) Left- and right-shifts of bit positions
- <<, >>, <<=, >>=
- Sign extension
- '>>>' is non-sign-extended right shift

Byte, short, int, long

Bit-by-bit. Try it out! 1 & 1, 2 & 1, 3 & 1, 7 & 3.

What is left shift equivalent to? How about right-shift?
SIGN EXTENSION!!

Binary Data Manipulation

Ok, why would anyone ever want to do this?

```
int  s_state =  0xace1;

int ShiftRegister() {
   int bit = ((s_state >>> 0) ^ (s_state >>> 2)^ (s_state >>> 3) ^ (s_state >>> 5)) & 0x1;
   s_state = ((s_state >>> 1) | (bit << 15));
   return s_state;
}

...
for(int i = 0; i < 1000; i++) { System.out.println("" + ShiftRegister()); }
```

Not often a reason. Some activities require it. Error correction coding, scrambling and encryption.
Communication protocols for audio and video.
Random number generation!

Hexadecimal!

Decimal, Binary, Hexadecimal…

Decimal means ??

```
  999
+   1
-----
 1000
```

Ones place, tens place, hundreds place, …

Decimal, Binary, Hexadecimal…

Binary means ??

```
  111
+   1
-----
 1000
```

Ones place, twos place, fours place, eights, place, 16s place, …

Decimal, Binary, Hexadecimal…

Hexadecimal means ??

Base 16. Digits are 0123456789abcdef

- a has value 'ten'
- b has value 'eleven'
- …
- f has value 'fifteen'
- '10' has value ??
- Instead of a "tens column" and "hundreds column", we have "sixteens column" and "256's column"

Write down some examples on board
Ones place, 16s place, 256s place, (16^3) place, …

Hexadecimal

- Instead of a "tens column" and "hundreds column", we have "sixteens column" and "256's column"

```
  fff
+   1
-----
 1000
```

Hexadecimal

- Instead of a "tens column" and "hundreds column", we have "sixteens column" and "256's column"

What is value of 1f ?

- Sixteen + fifteen = thirty-one (decimal)

How do we pronounce '10' if it is hexadecimal? "Ten hex"

- 100 is "one hundred hex"

What is decimal value of following hexadecimal expressions?

```
9          100       10 + 20
a          200       100 + 10
10         101       100 + 100
1a         10f
30         fff
```

Hexadecimal

How do we show that a variable in a Java program is hexadecimal?

- Variables are not decimal or hexadecimal or anything else. They have a value
- They are stored in binary, but that does not influence their value

But we can indicate that a constant value in Java is hexadecimal: use '0x' prefix, e.g.

```
int myVariable = 0x10; // value is sixteen
```

Why do we care about hexadecimal?

- Sometimes we want to know or express the binary representation of a variable's value. Binary expressions have too many darn characters: 32 for an int. Each hexadecimal character represents exactly 4 binary characters. It is very easy to convert between binary and hex.

Hexadecimal and Binary

Binary	Hex
0000 0000	0x00
1000 0000	0x80
0000 1111	0x0f
1010 0101	0xa5
0001 0000	0x10
0010 0000	0x20

Binary	Hex
1000 0000 0000 0000	
0001 0001 0001 1111	
1001 0110 0001 0111	
1000 1001 1010 1011	
1100 1101 1110 1111	
0001 0010 0100 1000	

CALL ON STUDENTS TO ANSWER

printf()

Formatted printing

printf()

An alternative to println() and print() that gives formatting control

- Used in many languages: C/C++, Python (sort of), Matlab, Ruby, etc
- Formatting string:

```
System.out.printf("String with format codes", variable1, variable2, …);
```

For example

```
int id = 4;
double cost = 2.5;
System.out.printf("Cost for item %d: %f\n", id, cost);
```

Cost for item 4: 2.5

Wasn't that exciting?

We can customize the formatting of the printing

```
int id = 4;
int id2 = 5678;
double cost = 2.5;
printf("Cost for item %4d: $%.02f\n", id, cost);
printf("Cost for item %4d: $%.02f\n", id2, 2*cost);
```

Cost for item 4: $2.50

Cost for item 5678: $5.00

printf() variable type codes

%d: integer
%x: integer printed in hexadecimal
%f: double
%e: double in Scientific Notation (6.02e23)
%c: char
%s: string

printf() special character printing

\n: newline
\t: tab
\\: backslash
\": double-quote i.e. "
%%: percent symbol i.e. %

printf() formatting codes

% d:	leave a space for a minus sign
%4d:	take up at least 4 spaces
%04d:	take up at least 4 spaces and use leading '0's instead of spaces
%,d:	put a comma between every 3 digits
%6.3f:	take up at least 6 characters, with at least 3 to the right of the decimal point

```
double val = 2.5;
System.out.printf("Cost:%7.3f\n", val);
```

Cost: 2.500

2 extra spaces