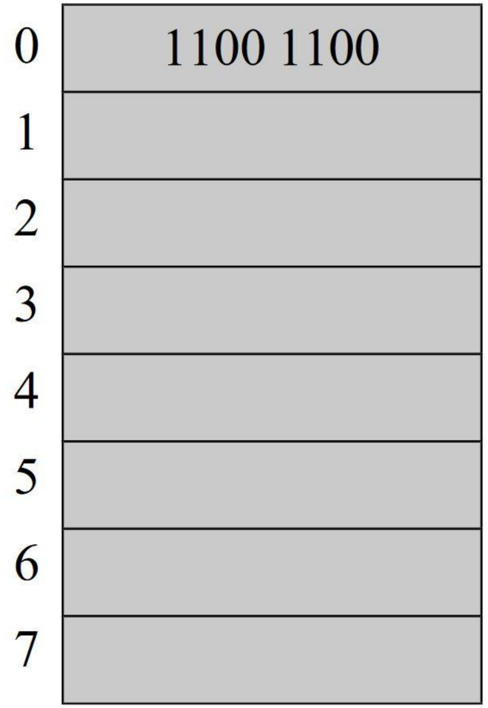
WEEK TWO

Acknowledgements: Slides created based off material provided by Dr. Travis Doom

MEMORY AND DATA REPRESENTATION

- Computers are made up of many switches
  - Switches can either be on or off
  - Binary: 1 or 0
  - Bit: a single binary digit
  - Byte: 8 bits
- Main memory
  - RAM: Random Access Memory
  - Contains running programs and their data
  - Divided into addresses
  - Address holds bits of information



BASE TEN DECIMAL SYSTEM VS BINARY

Base Ten

- Ten digits (0,1,2,3,4,5,6,7,8,9)
- What do we do when we run out of digits?

$9_{10} \rightarrow 10_{10}$

$99_{10} \rightarrow 100_{10}$

Binary

- Two digits (0,1)
- When we run out of digits, we add a 1 to the left and reset the right to 0

$0_{10} = 0_2$

$1_{10} = 1_2$

$2_{10} = 10_2$

$3_{10} = 11_2$

$4_{10} = 100_2$

$5_{10} = 101_2$

MORE ON BINARY

Base Ten	1B-Binary	Base Ten	1B-Binary
0	0000 0000	9	0000 1001
1	0000 0001	10	0000 1010
2	0000 0010	11	0000 1011
3	0000 0011	12	0000 1100
4	0000 0100	13	0000 1101
5	0000 0101	14	0000 1110
6	0000 0110	15	0000 1111
7	0000 0111	16	0001 0000
8	0000 1000	17	0001 0001

NEGATIVE NUMBERS

- Two's complement
  - Flip all the bits in the number
  - Add 1 to the result

$-3_{10} = ?$

$3_{10} = 0011_2$

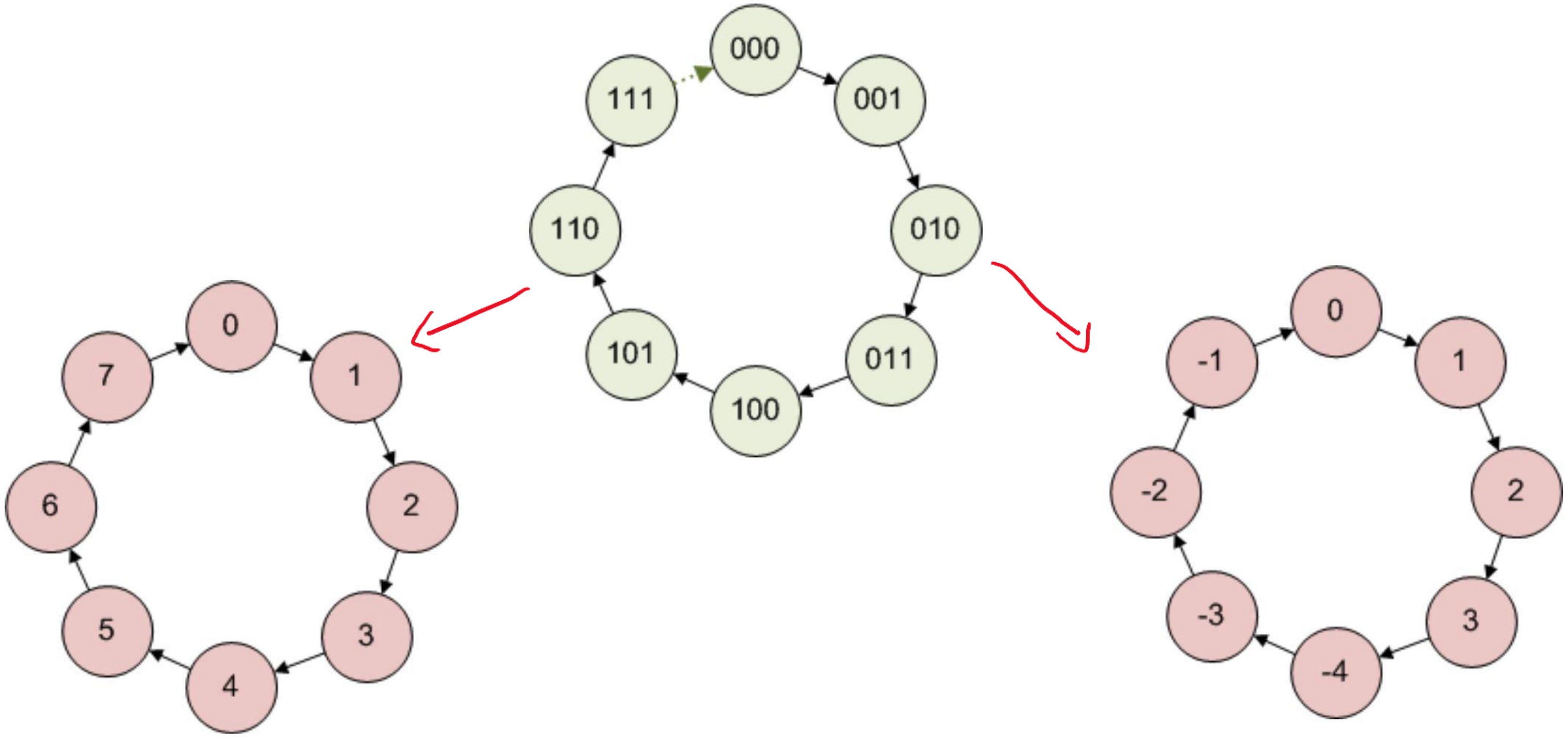
Flip all bits 1100

Add 1 1101

$-3_{10} = 1101_2$

Base Ten	4b-Binary	Base Ten	4b-Binary
+0	0000	-8	1000
+1	0001	-7	1001
+2	0010	-6	1010
+3	0011	-5	1011
+4	0100	-4	1100
+5	0101	-3	1101
+6	0110	-2	1110
+7	0111	-1	1111

DIFFERENT BINARY SYSTEMS

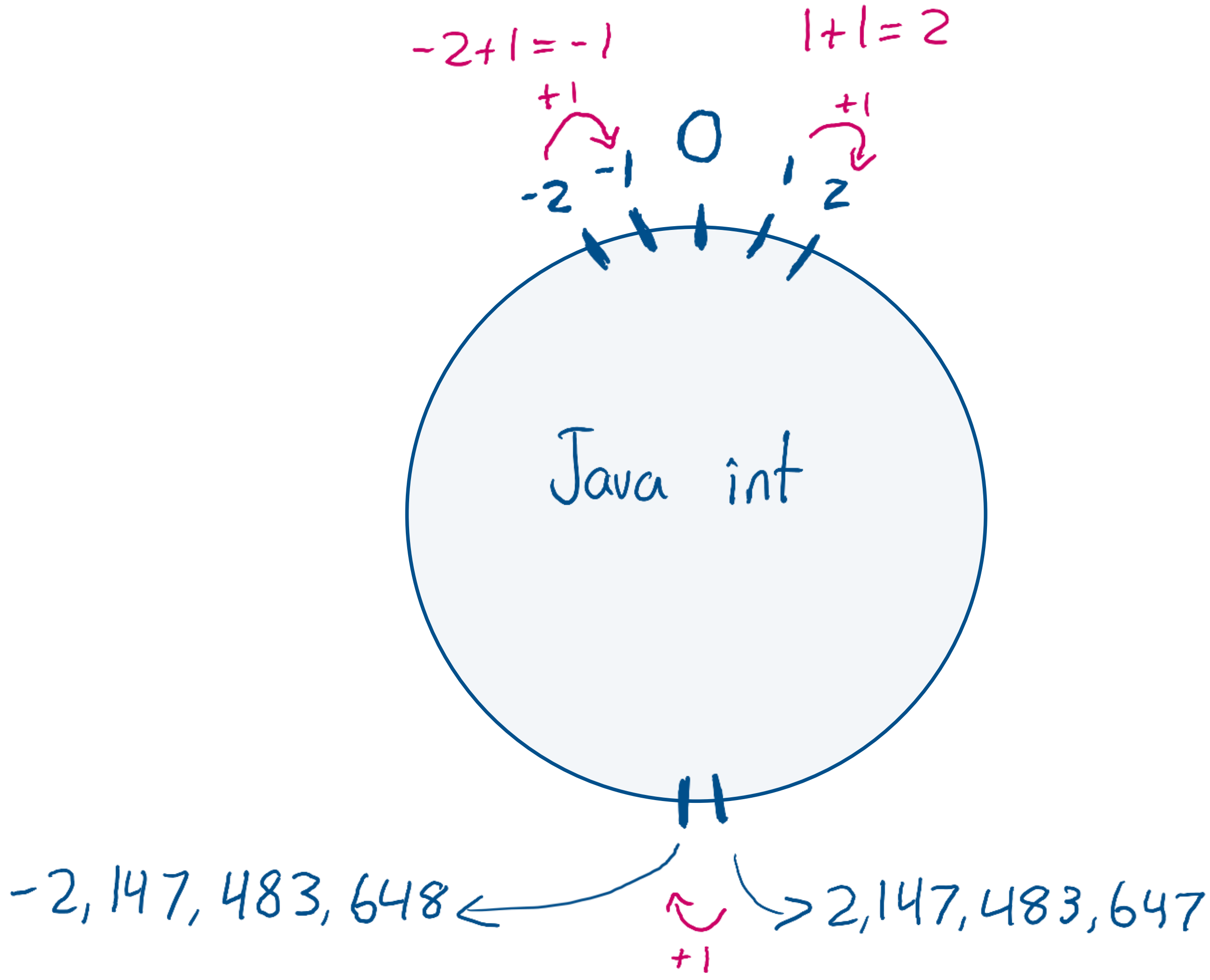


MEMORY SIZE FOR DATATYPES

byte	1 byte	Integers in the range -128 to +127
short	2 bytes	Integers in the range of -32,768 to +32,767
int	4 bytes	Integers in the range of -2,147,483,648 to +2,147,483,647
long	8 bytes	Integers in the range of -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
float	4 bytes	Floating-point numbers in the range of ±3.410e-38 to ±3.410e+38, with 7 digits of accuracy
double	8 bytes	Floating-point numbers in the range of ±1.710e-308 to ±1.710e+308, with 15 digits of accuracy

OVERFLOW

- There are real constraints on the size of number the memory can hold
- What happens when we exceed those constraints?
  - In Java, overflow occurs
  - No errors
  - Every numeric datatype affected



2,147,483,647 + 1 = -2,147,483,648

TYPE-CASTING AND INTEGER DIVISION

- What's the result of the following?
  - `double answer = 9 / 3;`
  - `answer = 9 / 2;`
- Must make our numbers into doubles
  - Add a .0 on either
  - Cast to a double
- Casting allows us to change the type of a variable
  - `answer = (double) 9 / 2;`
- Java will automatically cast types of lower precision to types of higher precision
- We must use type-casting if we wish to go from higher precision to lower precision
  - `int num = (int) 6.543;`

USEFUL STRING METHODS

- `char        charAt(int index)`
- `boolean     contains(CharSequence s)`
- `int         indexOf(int ch)`
- `int         lastIndexOf(int ch)`
- `boolean     isEmpty()`
- `int         length()`
- `String      replace(char oldChar, char newChar)`
- `String      substring(int beginIndex, int endIndex)`
- `String      toLowerCase()`
- `String      toUpperCase()`
- `String      valueOf(boolean b)`