CS0007 Recitation

THURSDAYS 12:00-12:50PM

MICHAEL BARTLETT

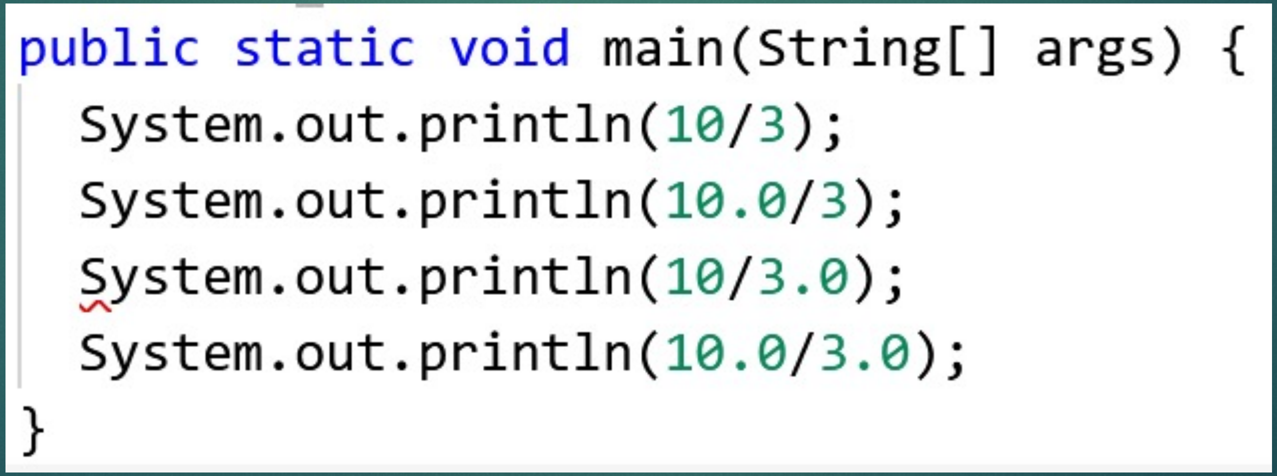
TODAY’S SLIDES ARE ADOPTED FROM LIN ROJTAS, ANOTHER CS0007 TA

Today’s Agenda

- Variables and arithmetic, pt 2
  - Integer division
  - Casting
  - The final keyword
- Strings!
  - Review of primitives and objects
- Lab 3 hints

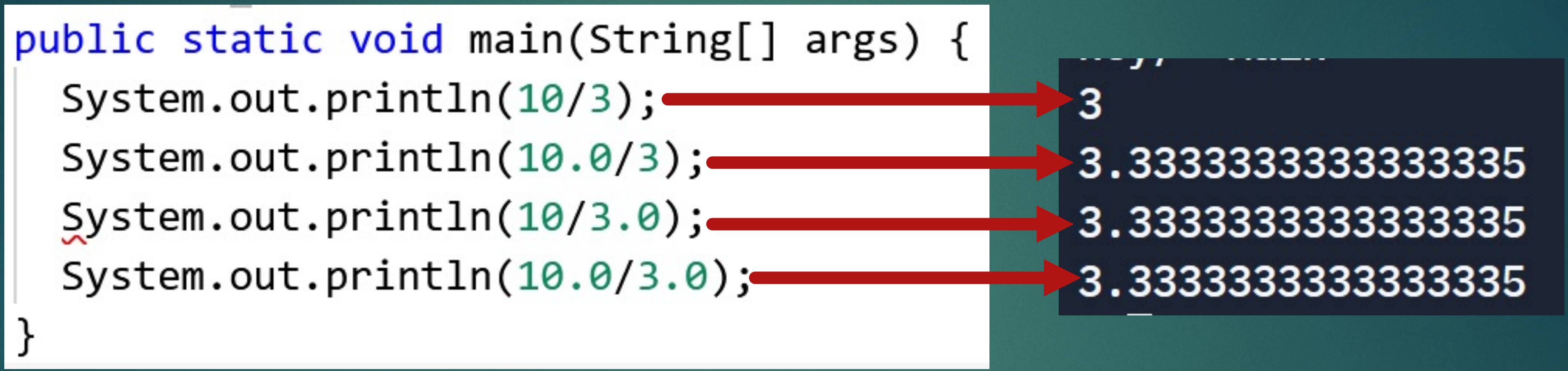
Integer Division

- Integer division and floating point (decimal) division are different!
- If you're dividing two integers, you will end up with a whole number (`int`). Otherwise, you'll end up with a decimal (`double`)



```
public static void main(String[] args) {
    System.out.println(10/3);
    System.out.println(10.0/3);
    System.out.println(10/3.0);
    System.out.println(10.0/3.0);
}
```

Integer division



```java
public static void main(String[] args) {
  System.out.println(10/3);
  System.out.println(10.0/3);
  System.out.println(10/3.0);
  System.out.println(10.0/3.0);
}
```

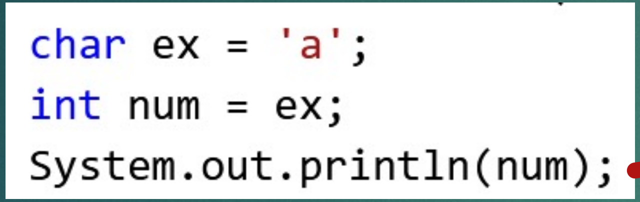
```
3
3.3333333333333335
3.3333333333333335
3.3333333333333335
```

Casting

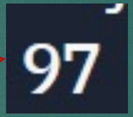
- When working with primitives, you can change certain data types into other data types!
- Widening casting (small to large)
  - byte → short → char → int → long → float → double
- Narrowing casting (large to small)
  - double → float → long → int → char → short → byte

Casting

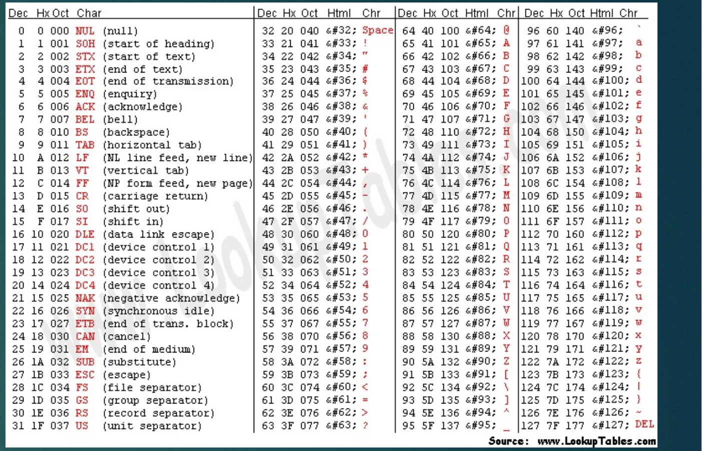
- Widening casting is done automatically…



```java
char ex = 'a';
int num = ex;
System.out.println(num);
```



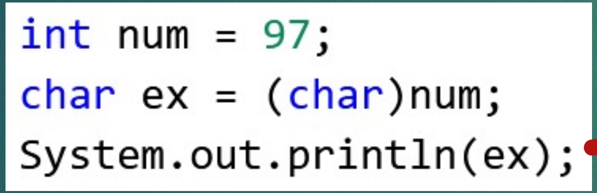
```
97
```



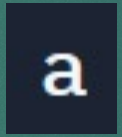
Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

- While narrowing casting has to be done manually…

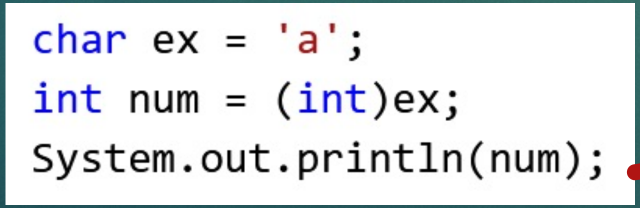


```java
int num = 97;
char ex = (char)num;
System.out.println(ex);
```

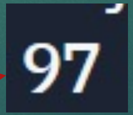


```
a
```

- In general, though, it’s good practice to do all casting manually



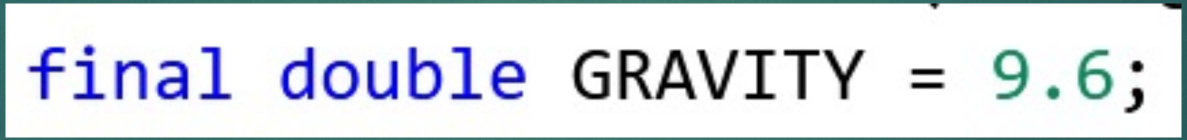
```java
char ex = 'a';
int num = (int)ex;
System.out.println(num);
```



```
97
```

The final Keyword

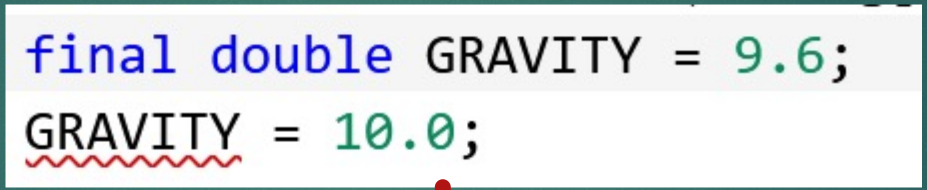
- The `final` keyword is used when you're declaring a constant
  - (a variable that cannot be changed or reassigned!)
- Convention: variable names with the `final` keyword are given names in all caps



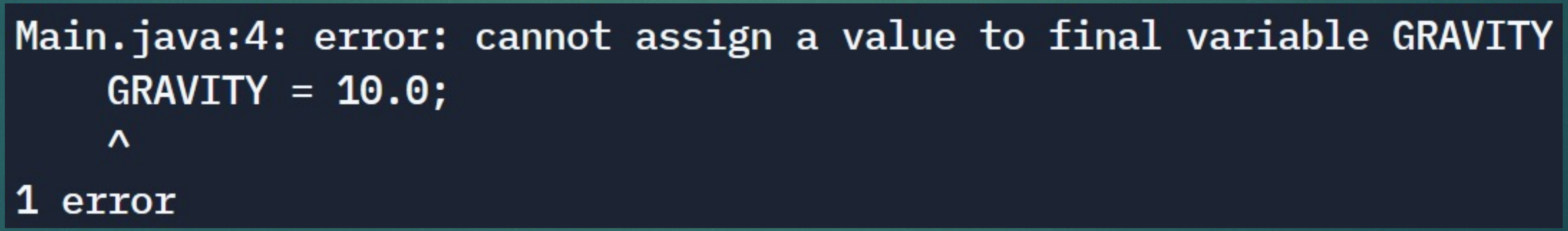
```
final double GRAVITY = 9.6;
```

What happens if we try to change a final variable?

- Well…



```java
final double GRAVITY = 9.6;
GRAVITY = 10.0;
```

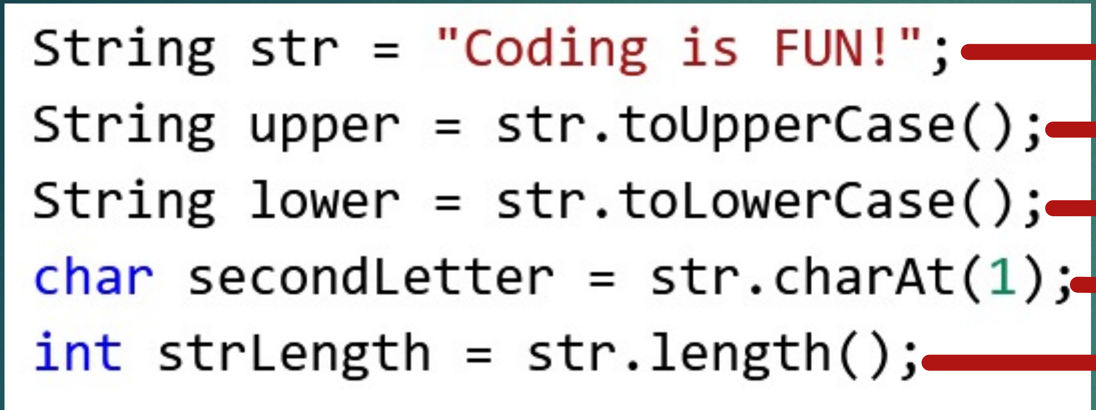


```
Main.java:4: error: cannot assign a value to final variable GRAVITY
    GRAVITY = 10.0;
    ^
1 error
```

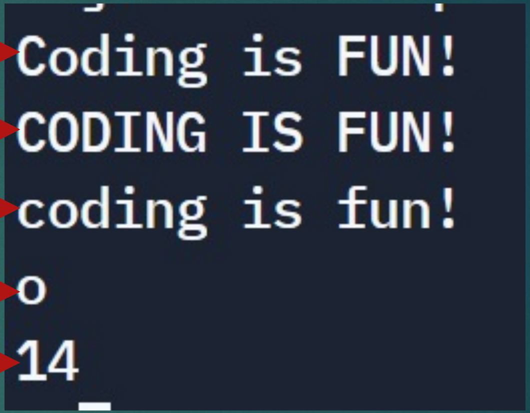
Strings

- Recall: Strings are objects, meaning they come with their own special operations
  - Instead of calling Strings “variables”, we typically say “objects” or “instances of”
    - Variable `num` versus object `name`
- Say we have a String object named `str`. Common operations are:
  - `str.toUpperCase();` → returns `str` in all uppercase letters
  - `str.toLowerCase();` → returns `str` in all lowercase letters
  - `str.charAt(int num);` → returns a `char` at index `num` in `str`
  - `str.length();` → returns an `int` giving the length of `str`

String Methods



```
String str = "Coding is FUN!";
String upper = str.toUpperCase();
String lower = str.toLowerCase();
char secondLetter = str.charAt(1);
int strLength = str.length();
```



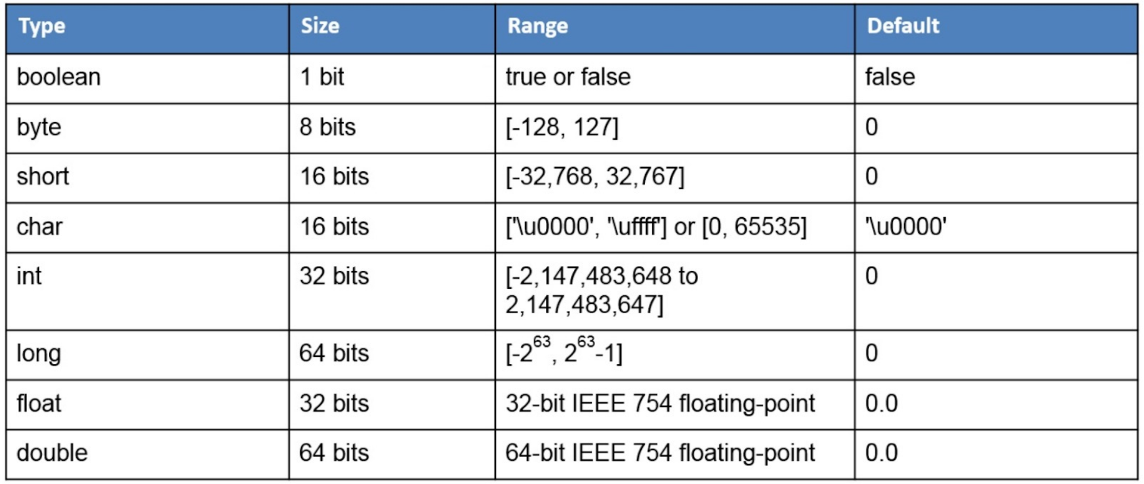
```
Coding is FUN!
CODING IS FUN!
coding is fun!
o
14
```

- Note: string indexing starts at 0!
  - `str.charAt(0);` would return ‘C’ while `str.charAt(14);` gives an error

Lab 3 hints

- Activity 1: Temperature Conversion
  - Tip: USE THE FORMULA GIVEN TO YOU!! Think about how you would do this problem on paper, and then do your best to translate that into code.
- Activity 2: Variable Types
  - Tip: You don’t need to print anything; we’re just looking to see that you’re using the appropriate variable type and naming conventions!

Helpful Table for Activity 2



Type	Size	Range	Default
boolean	1 bit	true or false	false
byte	8 bits	[-128, 127]	0
short	16 bits	[-32,768, 32,767]	0
char	16 bits	['\u0000', '\uffff'] or [0, 65535]	'\u0000'
int	32 bits	[-2,147,483,648 to 2,147,483,647]	0
long	64 bits	$[-2^{63}, 2^{63}-1]$	0
float	32 bits	32-bit IEEE 754 floating-point	0.0
double	64 bits	64-bit IEEE 754 floating-point	0.0

Lab 3 hints

- Activity 3: Game Playtime
  - Tip: make sure you're using integer division and the modulus (%) operation! This is another instance in which it would be helpful to do this on paper first and then do your best to convert it into code
- Activity 4: Degrees to Radians
  - Tip: reference the Java Math API! Don't be intimidated by the long equation; the computer does all of the heavy lifting for you. Just think about which methods in the Math class help with exponents and keep the order of operations in mind. Parentheses are your friend, too!
  - There is a sin Method in the Java API, but the point of this activity is to implement the formula using other Methods.

Other general lab tips

- Coding is hard! It’s okay if you don’t get it on the first go around.
- Make sure to check for common syntax errors (meaning, sometimes you’ll forget to add a semicolon at the end of a line; it happens!)
- Before you come to me asking for help, make sure you try working on your own first! There’s only so much that I can help you with and you already having some code we can work with helps a great deal.
- Again, don’t cheat. Cheating is for losers.

For next week

- Lab 3 is out!
    - I gave some tips in my slides, use them! It’s your first actual coding assignment, so I expect you guys to have a little trouble. That’s what my office hours are for, though. I will answer questions to the best of my ability.
    - The way you all submit assignments may change; keep an eye on Canvas for new submission instructions.
- Next week: Coding style, inputting data, and maybe if-structures!