Primitive Types

How to Store Data

Lecture Contents

- How to store different data
  - Computers only store zeros and ones!
- Java Primitive Types

How to store numbers?

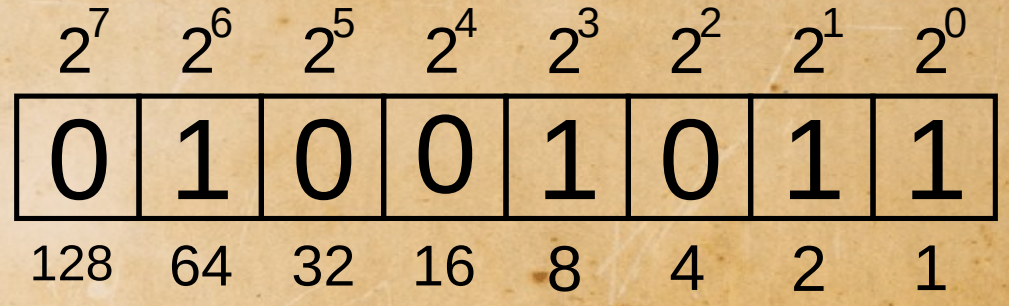
- Whole numbers
  - Just store as a binary string...

$75 - 64 = 11$

$11 - 8 = 3$

$3 - 2 = 1$

$1 - 1 = 0$



How to store numbers?

- Whole numbers
  - Just store as a binary string…
  - How many binary digits to use?

$75 - 64 = 11$

$11 - 8 = 3$

$3 - 2 = 1$

$1 - 1 = 0$

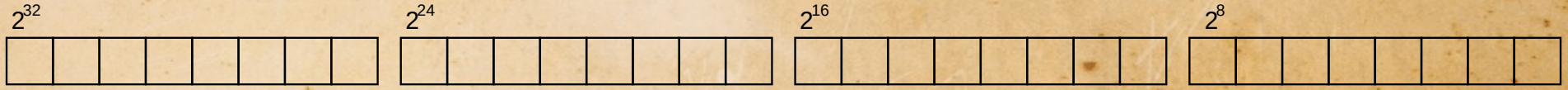
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	0	0	1	0	1	1
128	64	32	16	8	4	2	1

How to store numbers?

- Whole numbers
  - Just store as a binary string…
  - How many binary digits to use?

How to store numbers?

- Whole numbers
  - Just store as a binary string…
  - How many binary digits to use?
    - 8 bits → $2^8 = 256$
    - 16 bits → $2^{16} = 65536$
    - 32 bits → $2^{32} = 4{,}294{,}967{,}296$

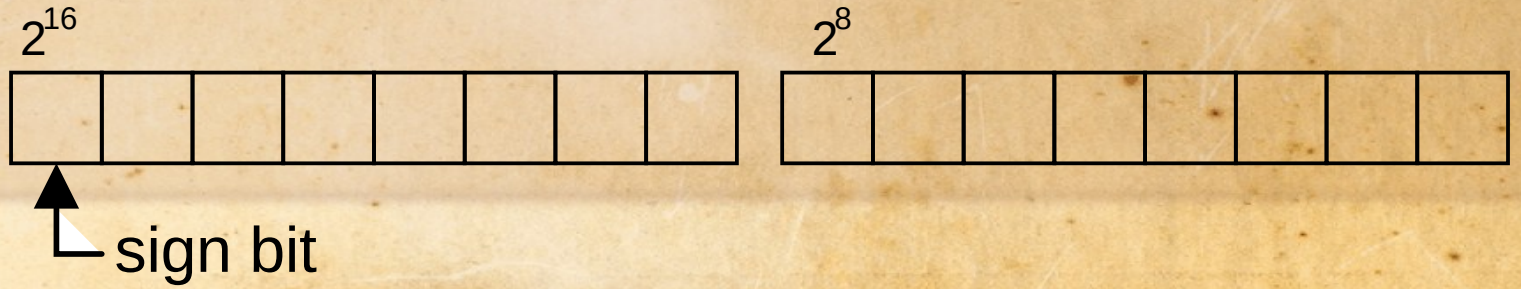


How to store numbers?

- Whole numbers
  - Just store as a binary string…
  - How many binary digits to use?
    - 8 bits → $2^8 = 256$
    - 16 bits → $2^{16} = 65536$
    - 32 bits → $2^{32} = 4{,}294{,}967{,}296$
  - ...but what about negative numbers?

How to store numbers?

- Whole numbers
  - Just store as a binary string…
  - How many binary digits to use?
    - 8 bits → $2^8 = 256$
    - 16 bits → $2^{16} = 65536$
    - 32 bits → $2^{32} = 4{,}294{,}967{,}296$
  - ...but what about negative numbers?



How to store numbers?

- Java integers

	bits	range
byte	8	-128 to +127
short	16	-32768 to +32767
int	32	-2,147,483,648 to +2,147,483,647
long	64	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

How to store numbers?

- In most cases, just use **`int`**

	bits	range
byte	8	-128 to +127
short	16	-32768 to +32767
int	**32**	**-2,147,483,648 to +2,147,483,647**
long	64	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

  - In Java, `Integer.MIN_VALUE` = -2,147,483,648 and `Integer.MAX_VALUE` = +2,147,483,647 (no need to type these numbers!)

How to store numbers?

REMEMBER:

Unless you have a particular need, to keep thing simple, when storing an *integer*, just use the primitive type:

```
int
```

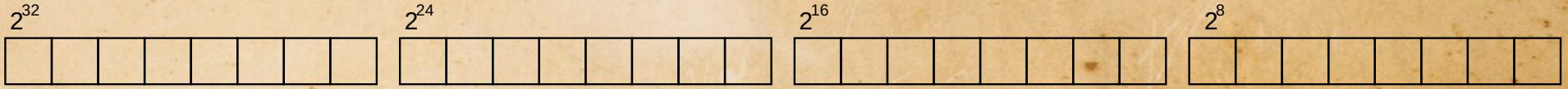
How to store numbers?

- What about numbers with decimals?

$$\pi = 3.14159265358979323846264 33\ldots$$

- What about huge numbers?
  - Avogadro's Number:

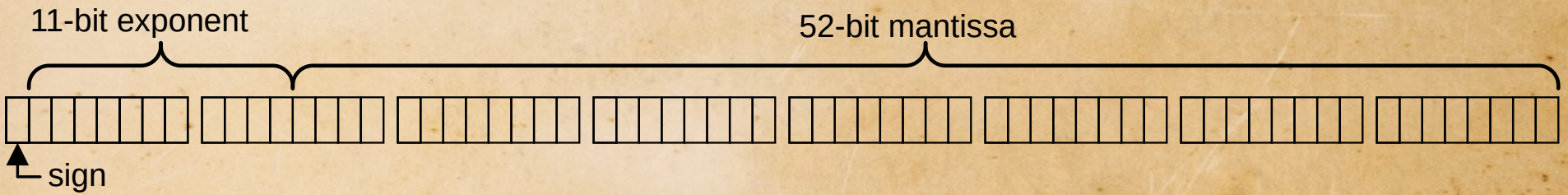
$$N_A = 6.02214076 \times 10^{23}$$



How to store numbers?

- What about huge numbers?
  - Scientific notation

$$\pm\, mantissa \times 2^{exponent}$$



How to store numbers?

- Java floating point numbers

	sign	exponent	mantissa
float	1	8	23
double	1	11	52

  - Specification: *IEEE 754*

How to store numbers?

REMEMBER:

Unless you have a particular need, to keep thing simple,
to store an *real number*, use the primitive type:

`double`

How to store text?

- How might we store: “Hello World!”
  - Character by character
    - Previously ***ASCII*** *– American Standard Code for Information Interchange* (1963)
    - Java uses ***Unicode***
      - 16 bits, so 65536 different characters
      - Lower characters match with ASCII

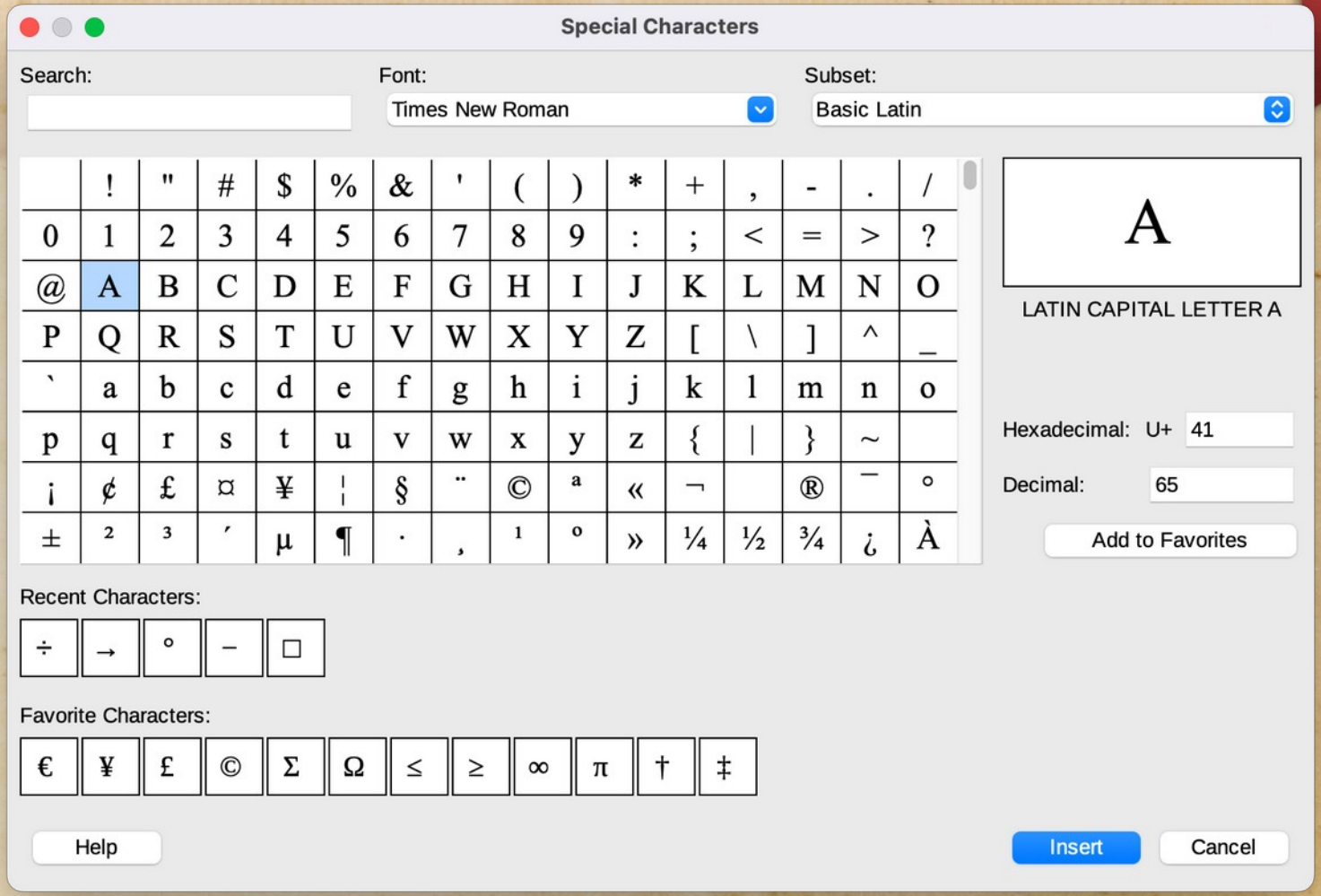
ASCII Character Codes in Hexadecimal

ASCII Character Set (0x20-0x7F)

hex	char	hex	char	hex	char	hex	char	hex	char	hex	char
20	*space*	30	0	40	@	50	P	60	`	70	p
21	!	31	1	41	A	51	Q	61	a	71	q
22	“	32	2	42	B	52	R	62	b	72	r
23	#	33	3	43	C	53	S	63	c	73	s
24	$	34	4	44	D	54	T	64	d	74	t
25	%	35	5	45	E	55	U	65	e	75	u
26	&	36	6	46	F	56	V	66	f	76	v
27	'	37	7	47	G	57	W	67	g	77	w
28	(	38	8	48	H	58	X	68	h	78	x
29	)	39	9	49	I	59	Y	69	i	79	y
2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
2F	/	3F	?	4F	O	5F	_	6F	o	7F	*delete*

USASCII code chart

Bits $b_7 b_6 b_5$ →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 b_3 b_2 b_1 ↓ / Row ↓ Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL


Special Characters
Search:
Font:
Times New Roman
Subset:
Basic Latin
A
LATIN CAPITAL LETTER A
Hexadecimal: U+ 41
Decimal: 65
Add to Favorites
Recent Characters:
Favorite Characters:
Help
Insert
Cancel

Java Primitive Types

- Integers
  - byte (1)
  - short (2)
  - int (4)
  - long (8)
- Real
  - float (4)
  - double (8)
- True/False
  - boolean (1 bit?)
- Letters
  - char (2)

AP Java Subset Primitive Types

- Integers
  - byte (1)
  - short (2)
  - **int** (4)
  - long (8)
- Real
  - float (4)
  - **double** (8)
- True/False
  - **boolean** (1 bit?)
- Letters
  - **char** (2)

Primitive Types

How to Store Data