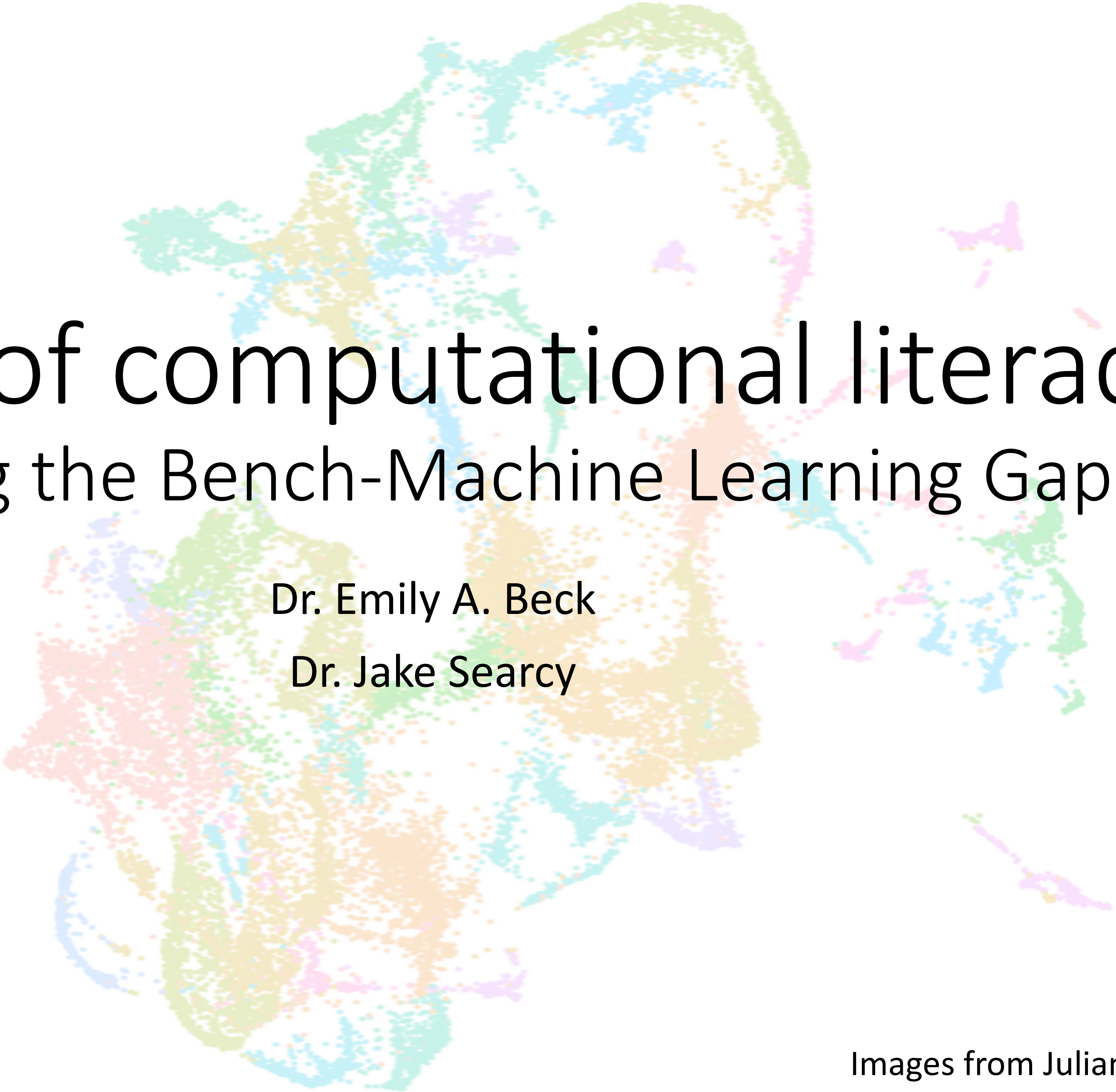


Basics of computational literacy

Bridging the Bench-Machine Learning Gap

Dr. Emily A. Beck

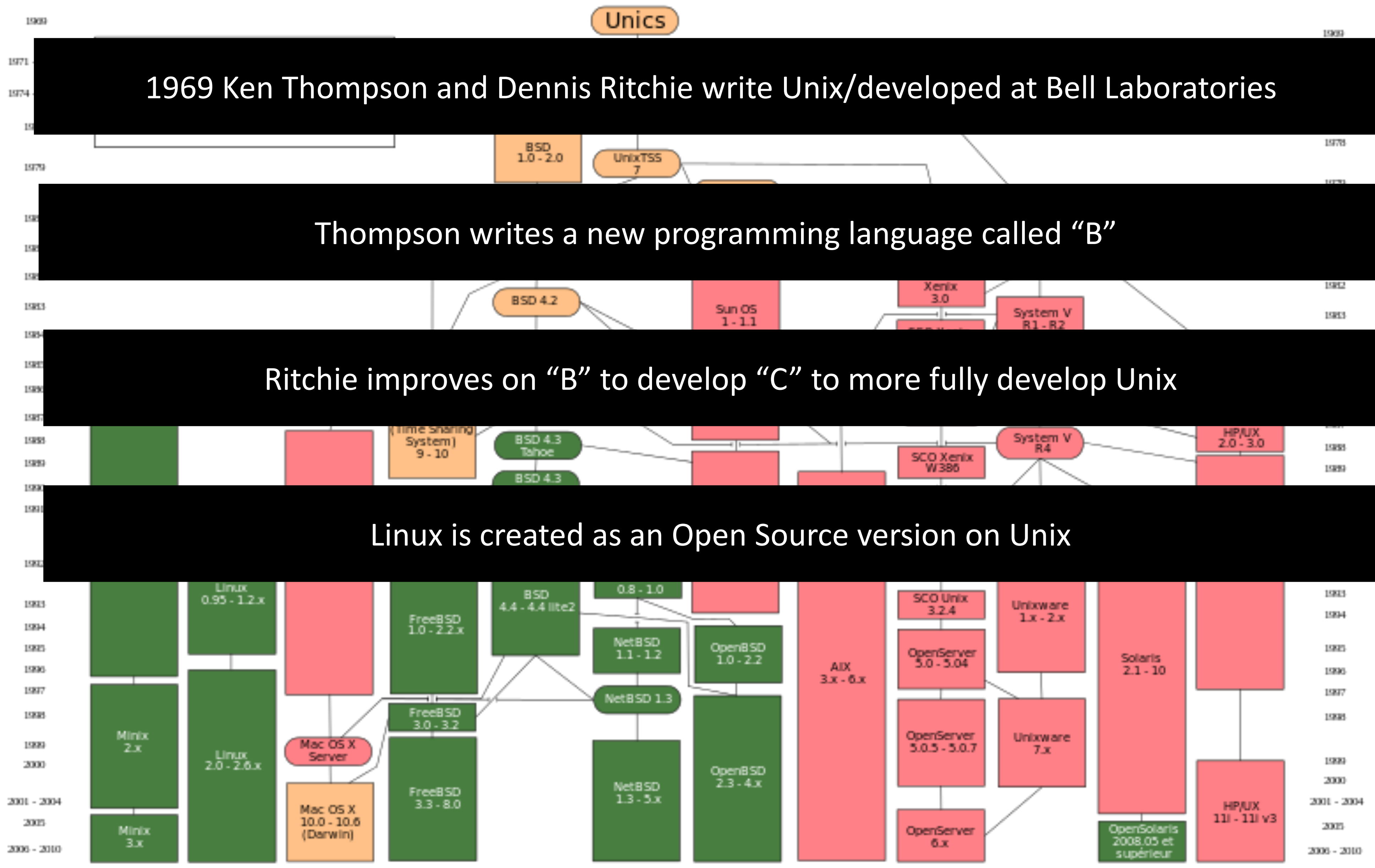
Dr. Jake Searcy

Images from Julian Catchen-Evomics

How many of you are familiar with Unix?

What about Linux?

Unix has a long complicated history



1969 Ken Thompson and Dennis Ritchie write Unix/developed at Bell Laboratories

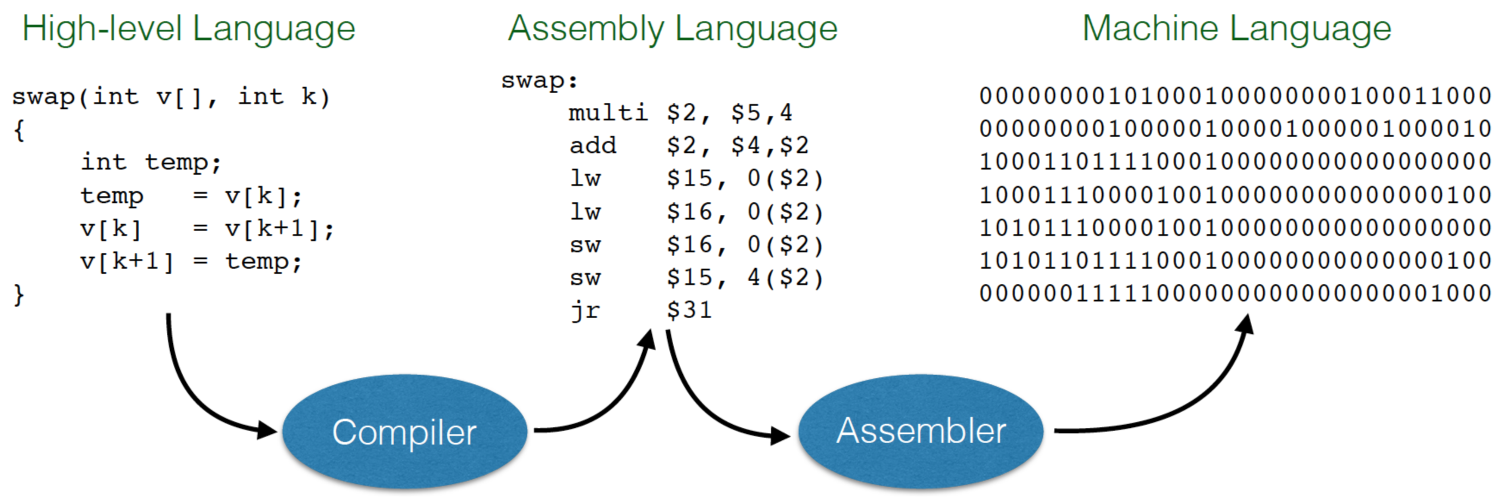
Thompson writes a new programming language called “B”

Ritchie improves on “B” to develop “C” to more fully develop Unix

Linux is created as an Open Source version on Unix

There are two “flavors” of code.
High-level **Source code** and **Machine code**.
How do these types of code work together?

Compilers, Linkers, Assemblers



```
swap(int v[], int k)
{
    int temp;
    temp   = v[k];
    v[k]   = v[k+1];
    v[k+1] = temp;
}
```

```
swap:
    multi $2, $5,4
    add   $2, $4,$2
    lw    $15, 0($2)
    lw    $16, 0($2)
    sw    $16, 0($2)
    sw    $15, 4($2)
    jr    $31
```

```
00000000101000100000000100011000
00000000100000100001000001000010
10001101111000100000000000000000
10001110000100100000000000000100
10101110000100100000000000000000
10101101111000100000000000000100
00000011111000000000000000001000
```

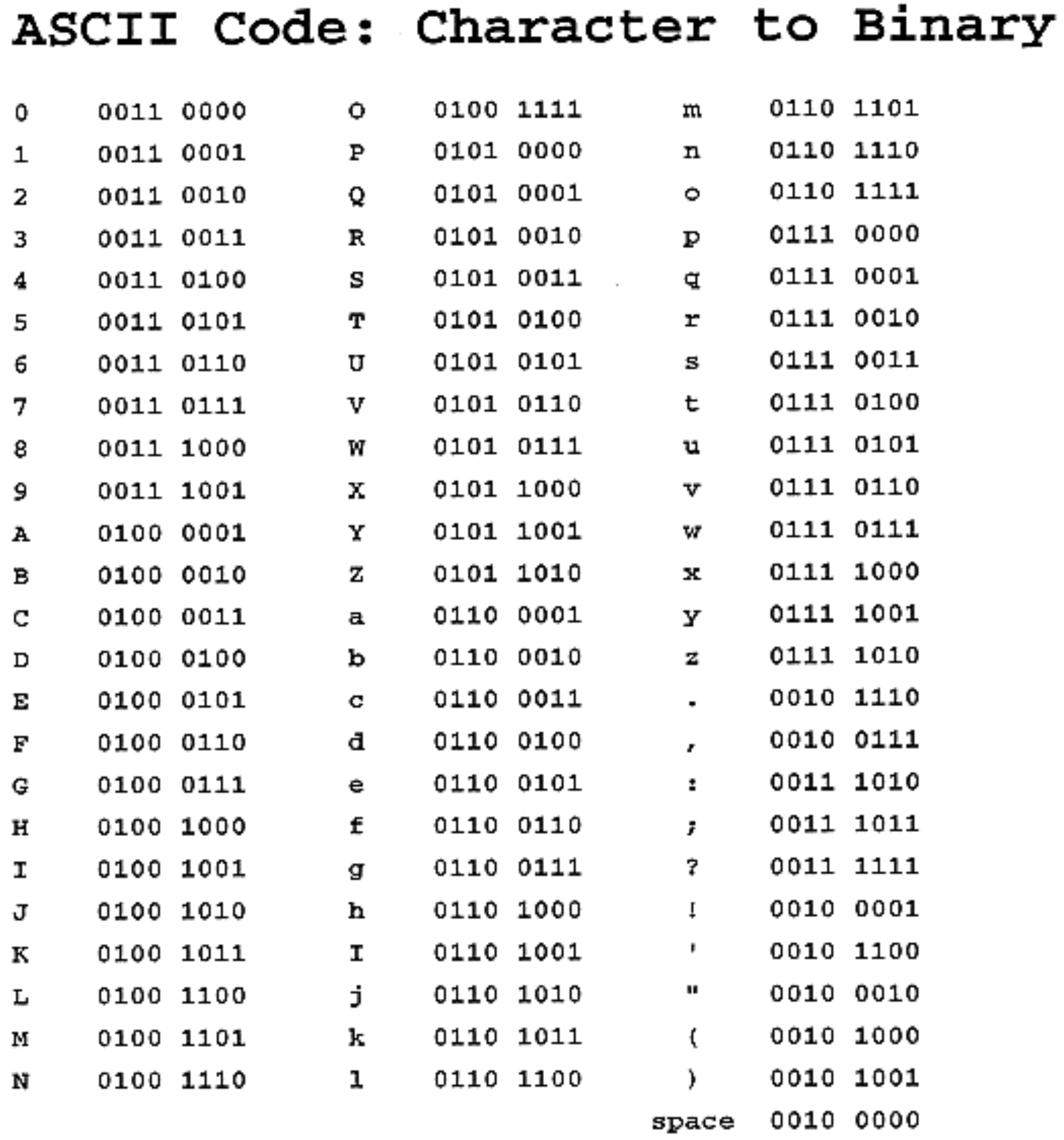
Machine code (Binary) is made of 0s and 1s and is used to represent all types of characters.

This is important later when we talk about Bam (Binary Alignment Map) files.

Base 10 (decimal) versus Base 2 (binary)

0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
17	10001
18	10010
19	10011
20	10100
...	...
99	1100011
100	1100100
101	1100101

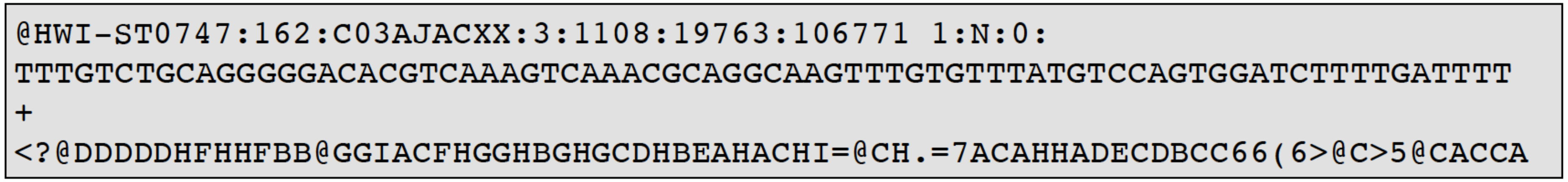
ASCII is used to represent text and can be easily converted into binary



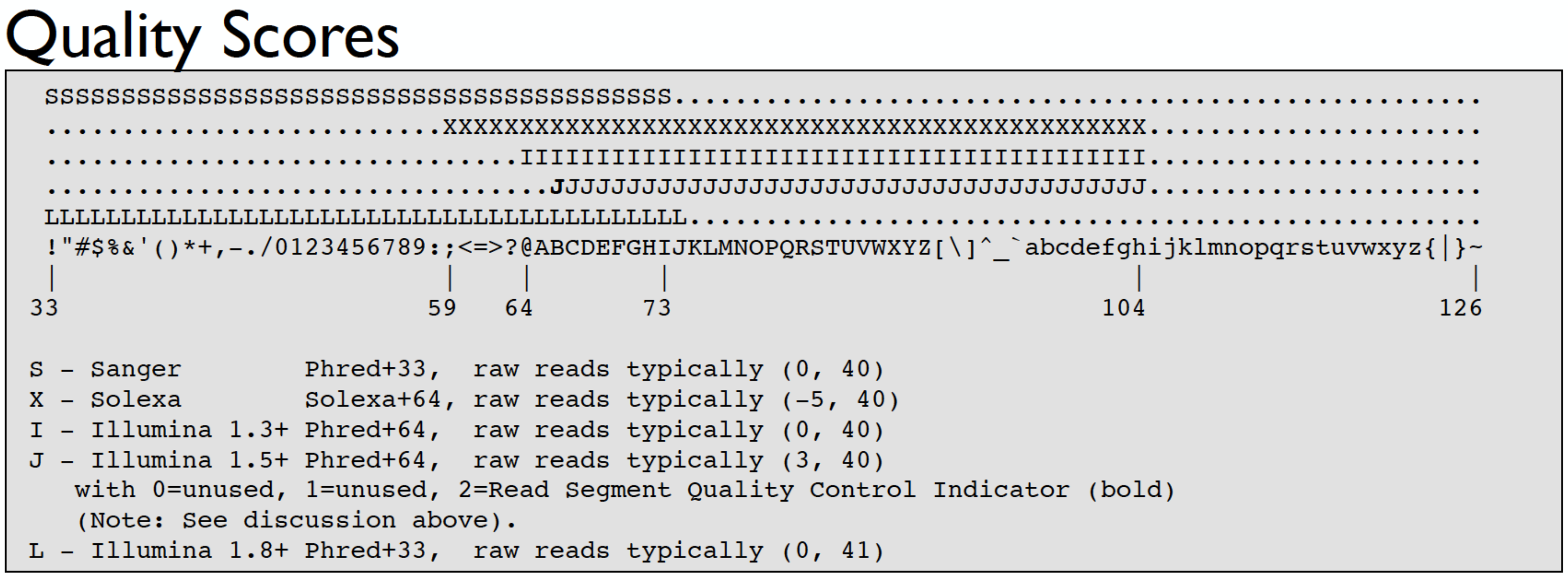
ASCII Code: Character to Binary

0	0011 0000	O	0100 1111	m	0110 1101
1	0011 0001	P	0101 0000	n	0110 1110
2	0011 0010	Q	0101 0001	o	0110 1111
3	0011 0011	R	0101 0010	p	0111 0000
4	0011 0100	S	0101 0011	q	0111 0001
5	0011 0101	T	0101 0100	r	0111 0010
6	0011 0110	U	0101 0101	s	0111 0011
7	0011 0111	V	0101 0110	t	0111 0100
8	0011 1000	W	0101 0111	u	0111 0101
9	0011 1001	X	0101 1000	v	0111 0110
A	0100 0001	Y	0101 1001	w	0111 0111
B	0100 0010	Z	0101 1010	x	0111 1000
C	0100 0011	a	0110 0001	y	0111 1001
D	0100 0100	b	0110 0010	z	0111 1010
E	0100 0101	c	0110 0011	.	0010 1110
F	0100 0110	d	0110 0100	,	0010 0111
G	0100 0111	e	0110 0101	:	0011 1010
H	0100 1000	f	0110 0110	;	0011 1011
I	0100 1001	g	0110 0111	?	0011 1111
J	0100 1010	h	0110 1000	!	0010 0001
K	0100 1011	I	0110 1001	'	0010 1100
L	0100 1100	j	0110 1010	"	0010 0010
M	0100 1101	k	0110 1011	(	0010 1000
N	0100 1110	l	0110 1100	)	0010 1001
				space	0010 0000

This will be important later because Fastq files use ASCII to encode quality scores!



```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```



Quality Scores

```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................................
  ..........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX......................
  ...............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII......................
  .................................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ......................
  LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL....................................................
  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
  |                         |    |        |                              |                     |
 33                        59   64       73                            104                   126

 S - Sanger        Phred+33,  raw reads typically (0, 40)
 X - Solexa        Solexa+64, raw reads typically (-5, 40)
 I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
 J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
    (Note: See discussion above).
 L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

For ”Gateway Week” we are going to focus on source code using **Bash (Shell)** and **R**

First, we will get oriented with the programs we will be using throughout the workshop series and then get started with Bash!