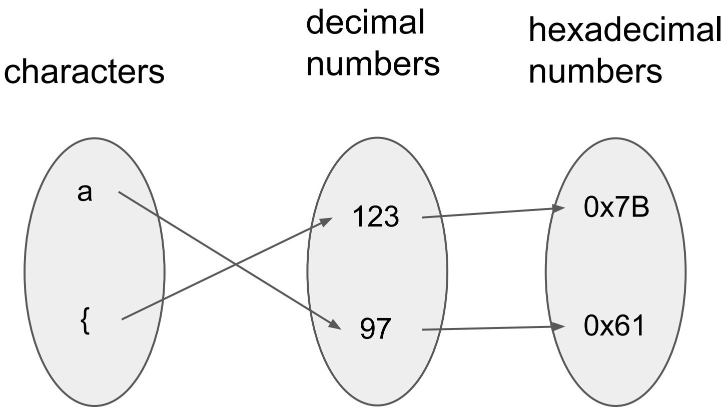
Introduction to Encodings

- Today's Plan
  - Discussions on Functions and Mappings
  - Introduce the ASCII Character Encoding
  - Introduce the UTF-8 Character Encoding

Mappings and Functions

- Mapping: assigning a relation between
- Function: a binary relation between two sets
    - Encode: input -> output
    - Decode: output -> input
- A table can represent a function

INPUT	OUTPUT
5	8
2	5
4	7
7	? 10
h	? k



0 1 2 3 4 5 6 7 8 9

a b c d e f g h i j k l

An Encoding for the keyboard

- Look at your keyboard.
  - a-z, A-Z, 0-9, !@#$%^&*()_+-~`,./<>?;':"[]\{}|
  - don't forget: space, tab, return, and delete key
  - plus we need other stuff:
  - All total, we we have 128 things to encode
- We need to devise an encoding that maps everything to numbers
- How many bits do we need? How many things do we bits in a byte?
- An example of a fixed-width encoding!
- Let's build a table! Keyboard Table

ASCII

- ASCII, abbreviated from American Standard Code for Information Interchange, is a character encoding standard for electronic communication.

- $ man ascii

- gdb: a debugger -- but I want a GUI
  - print /d 'a'
  - print /t 'a'
  - print /c 100
  - print /t 100
  - print /t (unsigned char) 10 -100
  - print /t (unsigned char) (10 -100)

Formats:
o - octal
x - hexadecimal
u - unsigned decimal
t - binary
f - floating point
a - address
c - char
s - string

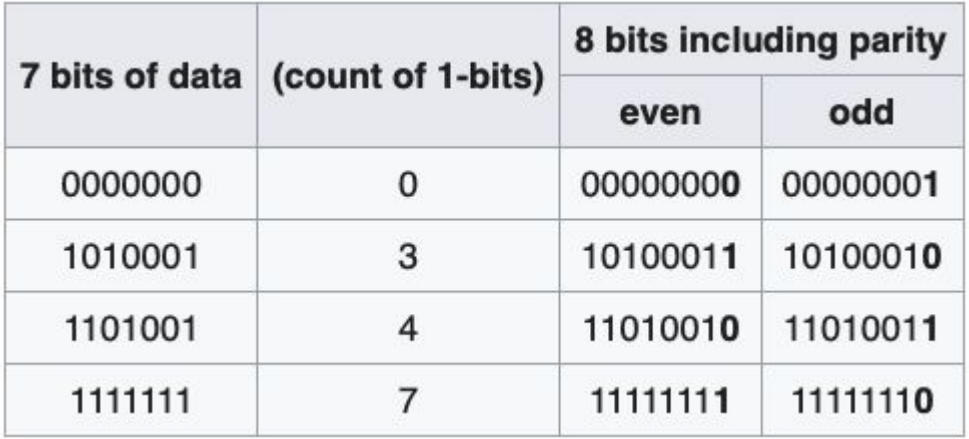
Parity Bit (or Check Bit)

- We are only using 7 of the 8 bits, what shall we do with it.

```
(gdb) print /t 'a'
$29 = 11101001
```

- Algorithm (odd)
  a. count the number of 1's
  b. add a 1 to make odd
  c. transmit
  d. receive
  e. count the number of 1's
  f. off even, ask for the data to be reser

- Checksum.. no need



7 bits of data	(count of 1-bits)	8 bits including parity	
		even	odd
0000000	0	00000000	00000001
1010001	3	10100011	10100010
1101001	4	11010010	11010011
1111111	7	11111111	11111110

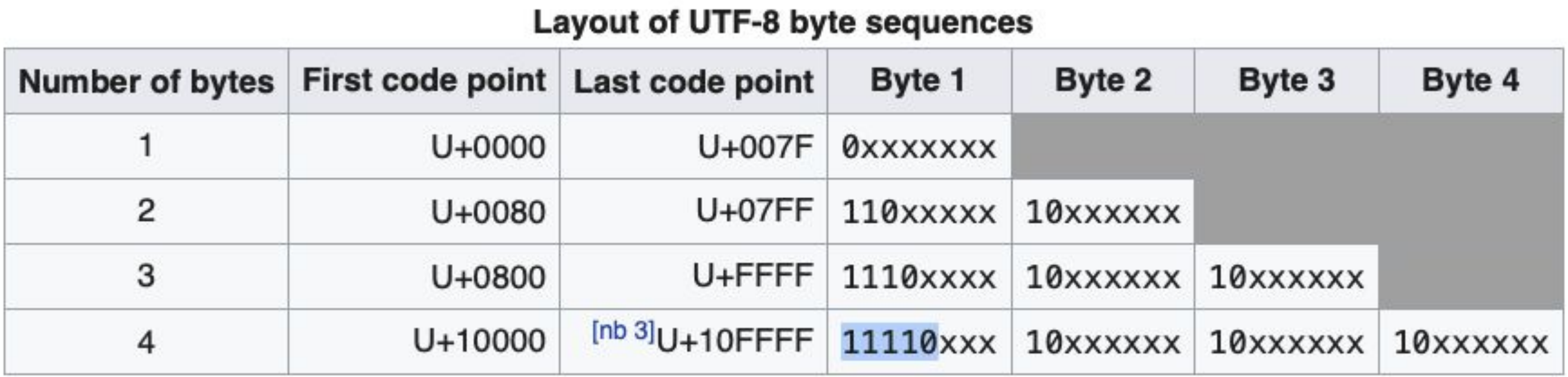
Extended ASCII and UTF-8 (unicode)

- We could use that bit to encode more stuff: 0..255
- But we have even more stuff. Let's use 16 bits to encode: 0..64K
- But now we have doubled what we need to send..
- Enter variable-length encoding.
  - Send only a byte for the most common symbols
  - Use the msb to indicate a variable length encoding
- UTF-8: encodes >2,000,000 (2^21) values, using a maximum of 4 bytes
- Defines four type of bytes:
  - ASCII byte: begins with a 0 (1-byte indicator)
  - Continuation byte: begins with a 10
  - 2-byte Indicator: begins with a 110
  - 3-byte Indicator: begins with a 1110
  - 4-byte Indicator: begins with a 11110

Extended ASCII and UTF-8

1110 0011   10101110   10 101010
1110 0011   10101110   10101010

- The list of UTF-8 characters:
- Layout of the bits:
- Example on how to lay it out:



Layout of UTF-8 byte sequences

Number of bytes	First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
1	U+0000	U+007F	0xxxxxxx			
2	U+0080	U+07FF	110xxxxx	10xxxxxx		
3	U+0800	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
4	U+10000	[nb 3]U+10FFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx