CMSC201
Computer Science I for Majors
Lecture 22 – Data Representation
Last Class We Covered

• Sorting
  – Bubble
  – Selection
  – Quick

• Searching
  – Linear
  – Binary
Any Questions from Last Time?
Today’s Objectives

• To understand how data is represented and stored in memory
  – Binary numbers
  – Hexadecimal numbers
  – Converting
    • Binary to Decimal
    • Decimal to Binary
  – ASCII
Binary Numbers
Binary Numbers

• Computers store all information (code, text, images, sound,) as a binary representation
  – “Binary” means only two parts: 0 and 1

• Specific formats for each file help the computer know what type of item/object it is

• But why use binary?
Decimal vs Binary

• Why do we use decimal numbers?
  – Ones, tens, hundreds, thousands, etc.

• But computers don’t have fingers...
  – What do they have instead?

• They only have two states: “on” and “off”
### Decimal Example

- How do we represent a number like 50,932?

<table>
<thead>
<tr>
<th>Place</th>
<th>Value</th>
<th>Digit</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ten thousands</td>
<td>$10^4$</td>
<td>5</td>
<td>$2 \times 10^0$</td>
</tr>
<tr>
<td>thousands</td>
<td>$10^3$</td>
<td>0</td>
<td>$3 \times 10^1$</td>
</tr>
<tr>
<td>hundreds</td>
<td>$10^2$</td>
<td>9</td>
<td>$9 \times 10^2$</td>
</tr>
<tr>
<td>tens</td>
<td>$10^1$</td>
<td>3</td>
<td>$0 \times 10^3$</td>
</tr>
<tr>
<td>ones</td>
<td>$10^0$</td>
<td>2</td>
<td>$5 \times 10^4$</td>
</tr>
</tbody>
</table>

\[ \begin{align*}
2 \times 10^0 &= 2 \\
3 \times 10^1 &= 30 \\
9 \times 10^2 &= 900 \\
0 \times 10^3 &= 0000 \\
5 \times 10^4 &= 50000 \\
\end{align*} \]

---

Total: 50932

Decimal uses 10 digits, so…
Another Decimal Example

<table>
<thead>
<tr>
<th></th>
<th>6</th>
<th>7</th>
<th>4</th>
<th>9</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^4$</td>
<td>10000</td>
<td>$10^3$</td>
<td>1000</td>
<td>$10^2$</td>
<td>100</td>
</tr>
<tr>
<td>$10^1$</td>
<td>1000</td>
<td>$10^0$</td>
<td>100</td>
<td>1000</td>
<td>1</td>
</tr>
<tr>
<td>$10^0$</td>
<td>60000</td>
<td>400</td>
<td>90</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

$60000+7000+400+90+3 = 67493$
**Binary Example**

- Let’s do the same with 10110 in binary

<table>
<thead>
<tr>
<th>sixteens</th>
<th>eights</th>
<th>fours</th>
<th>twos</th>
<th>ones</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>(2^4)</th>
<th>(2^3)</th>
<th>(2^2)</th>
<th>(2^1)</th>
<th>(2^0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 (\times 2^0) = 0</td>
<td>1 (\times 2^1) = 2</td>
<td>1 (\times 2^2) = 4</td>
<td>0 (\times 2^3) = 0</td>
<td>1 (\times 2^4) = 16</td>
</tr>
</tbody>
</table>

**Total:** 22

Binary uses 2 digits, so our base isn’t 10, but...
Binary to Decimal Conversion

- Step 1: Draw Conversion Box
- Step 2: Enter Binary Number
- Step 3: Multiply
- Step 4: Add

\[
\begin{array}{cccccccc}
1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\
128 & 0 & 0 & 0 & 8 & 4 & 0 & 1 \\
\end{array}
\]

\[128 + 0 + 0 + 0 + 8 + 4 + 0 + 1 = 141\]
Exercise: Converting From Binary

• What are the decimals equivalents of...
  101
  1111
  100000
  101010
  0010 1010
  1000 0000

Longer binary numbers are often broken into blocks of four digits for the sake of readability.
Exercise: Converting From Binary

• What are the decimals equivalents of...

101       = 4+0+1       = 5
1111      = 8+4+2+1     = 15
100000    = 32+0+0+0+0+0= 32
101010    = 32+0+8+0+2+0= 42
0010 1010  = 32+0+8+0+2+0= 42
1000 0000  = 128+ . . .+0+0 = 128
Decimal to Binary Conversion

- Step 1: Draw Conversion Box
- Step 2: Compare decimal to highest binary value
- Step 3: If binary value is smaller, put a 1 there and subtract the value from the decimal number
- Step 4: Repeat until 0

Convert 163 to binary

\[
\begin{array}{cccccccc}
2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
\end{array}
\]

163 - 128 = 35  
35 - 32 = 3  
3 - 2 = 1  
1 - 1 = 0
Converting to Binary

• What are the binary equivalents of...

  9
  27
  68
  216
  255
Converting to Binary

• What are the binary equivalents of...

9    = 1001 (or 8+1)
27   = 0001 1011 (or 16+8+2+1)
68   = 0100 0100 (or 64+4)
216  = 1101 1000
    (or 128+64+16+8)
255  = 1111 1111
    (or 128+64+32+16+8+4+2+1)
Binary Tips and Tricks

• Some “sanity checking” rules for conversions:
  1. Binary can only be 1 or 0
     – If you get “2” of something, it’s wrong
  2. Odd numbers must have a 1 in the ones column
     – Why? (And what’s the rule for even numbers?)
  3. Each column’s value is the sum of all of the previous columns (to the right) plus one
     – In decimal, what column comes after 999?
Hexadecimal Numbers
Decimal Representation

• Decimal uses 10 digits
  – Decimal, *deci* = 10
  – The digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9

<table>
<thead>
<tr>
<th>ten millions</th>
<th>millions</th>
<th>hundred thousands</th>
<th>ten thousands</th>
<th>thousands</th>
<th>hundreds</th>
<th>tens</th>
<th>ones</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>8</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$10^7$</td>
<td>$10^6$</td>
<td>$10^5$</td>
<td>$10^4$</td>
<td>$10^3$</td>
<td>$10^2$</td>
<td>$10^1$</td>
<td>$10^0$</td>
</tr>
</tbody>
</table>
Binary Representation

- Binary uses 2 digits
  - Binary, $b_i = 2$
  - The digits used are 0 and 1

\[
\begin{array}{cccccccc}
\text{one hundred and} & \text{twenty-eights} & \text{sixty-fours} & \text{thirty-twos} & \text{sixteens} & \text{eights} & \text{fours} & \text{twos} & \text{ones} \\
2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
1 & 0 & 0 & 1 & 1 & 1 & 1 & 0
\end{array}
\]
Hexadecimal Representation

- Hexadecimal (or just “hex”) uses 16 digits
  - Hexadecimal, hex = 6 plus deci = 10 → 16
  - The digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9
- Any letters A (10), B (11), C (12), D (13), E (14), and F (15)

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>8</th>
<th>6</th>
<th>3</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>16&lt;sup&gt;7&lt;/sup&gt;</td>
<td>16&lt;sup&gt;6&lt;/sup&gt;</td>
<td>16&lt;sup&gt;5&lt;/sup&gt;</td>
<td>16&lt;sup&gt;4&lt;/sup&gt;</td>
<td>16&lt;sup&gt;3&lt;/sup&gt;</td>
<td>16&lt;sup&gt;2&lt;/sup&gt;</td>
<td>16&lt;sup&gt;1&lt;/sup&gt;</td>
<td>16&lt;sup&gt;0&lt;/sup&gt;</td>
<td></td>
</tr>
</tbody>
</table>
Hexadecimal Representation

- Hexadecimal (or just “hex”) uses 16 digits
  - Hexadecimal, hex = 6 plus deci = 10 → 16
  - The digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9
    - And letters A (10), B (11), C (12), D (13), E (14), and F (15)
Hex to Binary Conversion

• A hexadecimal digit can be easily represented as four digits of binary (with leading zeros)

<table>
<thead>
<tr>
<th>Hex</th>
<th>Binary</th>
<th>Hex</th>
<th>Binary</th>
<th>Hex</th>
<th>Binary</th>
<th>Hex</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>4</td>
<td>0100</td>
<td>8</td>
<td>1000</td>
<td>C</td>
<td>1100</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>5</td>
<td>0101</td>
<td>9</td>
<td>1001</td>
<td>D</td>
<td>1101</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>6</td>
<td>0110</td>
<td>A</td>
<td>1010</td>
<td>E</td>
<td>1110</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>7</td>
<td>0111</td>
<td>B</td>
<td>1011</td>
<td>F</td>
<td>1111</td>
</tr>
</tbody>
</table>

• This makes conversion very simple
  – 7A0F becomes 0111 1010 0000 1111
  – 1100 0010 0110 1001 becomes C269
Hex to Decimal Conversion

• Possible to convert between decimal and hex
  ‒ But it requires calculating out multiples of 16
• Simpler to make a “side trip” to binary as an in-between step when converting
  ‒ 240 becomes 1111 0000 becomes F0
    • F0 is equal to (15 * 16^1) + (0 * 16^0) = 240 + 0 = 240
  ‒ 7D becomes 0111 1101 becomes 125
    • 7D is equal to (7 * 16^1) + (13 * 16^0) = 112 + 13 = 125
Number System Notation

• Because number systems share a subset of the same digits, it may be confusing which is which
  – For example, what is the value of 10?
    • In decimal it’s 10, in binary it’s 2, and in hex it’s 16

• To prevent this, numbers may often be prefixed with 0b, 0d, or 0x (binary, decimal, hex):
  – 0b1100 is binary, and has a value of 12
  – 0x15 is hexadecimlal, and has a value of 21
ASCII Values
ASCII Values

• ASCII is how text is represented in computers
  – Just like binary is how numbers are represented

• In ASCII, every character has a unique, individual numerical code
  – Lowercase and uppercase characters are separate
  – Codes go from 0 to 127

• Why 127?
# ASCII Table

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Hex</th>
<th>Char</th>
<th>Decimal</th>
<th>Hex</th>
<th>Char</th>
<th>Decimal</th>
<th>Hex</th>
<th>Char</th>
<th>Decimal</th>
<th>Hex</th>
<th>Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>[NULL]</td>
<td>32</td>
<td>20</td>
<td>[SPACE]</td>
<td>64</td>
<td>40</td>
<td>@</td>
<td>96</td>
<td>60</td>
<td>'</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>[START OF HEADING]</td>
<td>33</td>
<td>21</td>
<td>!</td>
<td>65</td>
<td>41</td>
<td>A</td>
<td>97</td>
<td>61</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>[START OF TEXT]</td>
<td>34</td>
<td>22</td>
<td>&quot;</td>
<td>66</td>
<td>42</td>
<td>B</td>
<td>98</td>
<td>62</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>[END OF TEXT]</td>
<td>35</td>
<td>23</td>
<td>#</td>
<td>67</td>
<td>43</td>
<td>C</td>
<td>99</td>
<td>63</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>[END OF TRANSMISSION]</td>
<td>36</td>
<td>24</td>
<td>$</td>
<td>68</td>
<td>44</td>
<td>D</td>
<td>100</td>
<td>64</td>
<td>d</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>[ENQUIRY]</td>
<td>37</td>
<td>25</td>
<td>%</td>
<td>69</td>
<td>45</td>
<td>E</td>
<td>101</td>
<td>65</td>
<td>e</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>[ACKNOWLEDGE]</td>
<td>38</td>
<td>26</td>
<td>&amp;</td>
<td>70</td>
<td>46</td>
<td>F</td>
<td>102</td>
<td>66</td>
<td>f</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>[BEL]</td>
<td>39</td>
<td>27</td>
<td>'</td>
<td>71</td>
<td>47</td>
<td>G</td>
<td>103</td>
<td>67</td>
<td>g</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>[BACKSPACE]</td>
<td>40</td>
<td>28</td>
<td>(</td>
<td>72</td>
<td>48</td>
<td>H</td>
<td>104</td>
<td>68</td>
<td>h</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>[HORIZONTAL TAB]</td>
<td>41</td>
<td>29</td>
<td>)</td>
<td>73</td>
<td>49</td>
<td>I</td>
<td>105</td>
<td>69</td>
<td>i</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>[VERTICAL TAB]</td>
<td>43</td>
<td>2B</td>
<td>+</td>
<td>75</td>
<td>4B</td>
<td>K</td>
<td>107</td>
<td>6B</td>
<td>k</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>[FORM FEED]</td>
<td>44</td>
<td>2C</td>
<td>,</td>
<td>76</td>
<td>4C</td>
<td>L</td>
<td>108</td>
<td>6C</td>
<td>l</td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>[CARRIAGE RETURN]</td>
<td>45</td>
<td>2D</td>
<td>-</td>
<td>77</td>
<td>4D</td>
<td>M</td>
<td>109</td>
<td>6D</td>
<td>m</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>[SHIFT OUT]</td>
<td>46</td>
<td>2E</td>
<td>.</td>
<td>78</td>
<td>4E</td>
<td>N</td>
<td>110</td>
<td>6E</td>
<td>n</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>[SHIFT IN]</td>
<td>47</td>
<td>2F</td>
<td>/</td>
<td>79</td>
<td>4F</td>
<td>O</td>
<td>111</td>
<td>6F</td>
<td>o</td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>[DATA LINK ESCAPE]</td>
<td>48</td>
<td>30</td>
<td>0</td>
<td>80</td>
<td>50</td>
<td>P</td>
<td>112</td>
<td>70</td>
<td>p</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>[DEVICE CONTROL 1]</td>
<td>49</td>
<td>31</td>
<td>1</td>
<td>81</td>
<td>51</td>
<td>Q</td>
<td>113</td>
<td>71</td>
<td>q</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>[DEVICE CONTROL 2]</td>
<td>50</td>
<td>32</td>
<td>2</td>
<td>82</td>
<td>52</td>
<td>R</td>
<td>114</td>
<td>72</td>
<td>r</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>[DEVICE CONTROL 3]</td>
<td>51</td>
<td>33</td>
<td>3</td>
<td>83</td>
<td>53</td>
<td>S</td>
<td>115</td>
<td>73</td>
<td>s</td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>[DEVICE CONTROL 4]</td>
<td>52</td>
<td>34</td>
<td>4</td>
<td>84</td>
<td>54</td>
<td>T</td>
<td>116</td>
<td>74</td>
<td>t</td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>[NEGATIVE ACKNOWLEDGE]</td>
<td>53</td>
<td>35</td>
<td>5</td>
<td>85</td>
<td>55</td>
<td>U</td>
<td>117</td>
<td>75</td>
<td>u</td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>[SYNCHRONOUS IDLE]</td>
<td>54</td>
<td>36</td>
<td>6</td>
<td>86</td>
<td>56</td>
<td>V</td>
<td>118</td>
<td>76</td>
<td>v</td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>[END OF TRANS. BLOCK]</td>
<td>55</td>
<td>37</td>
<td>7</td>
<td>87</td>
<td>57</td>
<td>W</td>
<td>119</td>
<td>77</td>
<td>w</td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>[CANCEL]</td>
<td>56</td>
<td>38</td>
<td>8</td>
<td>88</td>
<td>58</td>
<td>X</td>
<td>120</td>
<td>78</td>
<td>x</td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>[END OF MEDIUM]</td>
<td>57</td>
<td>39</td>
<td>9</td>
<td>89</td>
<td>59</td>
<td>Y</td>
<td>121</td>
<td>79</td>
<td>y</td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>[SUBSTITUTE]</td>
<td>58</td>
<td>3A</td>
<td>:</td>
<td>90</td>
<td>5A</td>
<td>Z</td>
<td>122</td>
<td>7A</td>
<td>z</td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>[ESCAPE]</td>
<td>59</td>
<td>3B</td>
<td>;</td>
<td>91</td>
<td>5B</td>
<td>[</td>
<td>123</td>
<td>7B</td>
<td>{</td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>[FILE SEPARATOR]</td>
<td>60</td>
<td>3C</td>
<td>&lt;</td>
<td>92</td>
<td>5C</td>
<td>\</td>
<td>124</td>
<td>7C</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>[GROUP SEPARATOR]</td>
<td>61</td>
<td>3D</td>
<td>=</td>
<td>93</td>
<td>5D</td>
<td>]</td>
<td>125</td>
<td>7D</td>
<td>}</td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>[RECORD SEPARATOR]</td>
<td>62</td>
<td>3E</td>
<td>&gt;</td>
<td>94</td>
<td>5E</td>
<td>^</td>
<td>126</td>
<td>7E</td>
<td>~</td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>[UNIT SEPARATOR]</td>
<td>63</td>
<td>3F</td>
<td>?</td>
<td>95</td>
<td>5F</td>
<td>_</td>
<td>127</td>
<td>7F</td>
<td>[DEL]</td>
</tr>
<tr>
<td>Decimal</td>
<td>Hex</td>
<td>Char</td>
<td>Decimal</td>
<td>Hex</td>
<td>Char</td>
<td>Decimal</td>
<td>Hex</td>
<td>Char</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----</td>
<td>---------------</td>
<td>---------</td>
<td>-----</td>
<td>---------------</td>
<td>---------</td>
<td>-----</td>
<td>---------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>[NULL]</td>
<td>32</td>
<td>20</td>
<td>[SPACE]</td>
<td>64</td>
<td>40</td>
<td>@</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>[START OF HEADING]</td>
<td>33</td>
<td>21</td>
<td>!</td>
<td>65</td>
<td>41</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>[START OF TEXT]</td>
<td>34</td>
<td>22</td>
<td>&quot;</td>
<td>66</td>
<td>42</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>[END OF TEXT]</td>
<td>35</td>
<td>23</td>
<td>#</td>
<td>67</td>
<td>43</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>[END OF TRANSMISSION]</td>
<td>36</td>
<td>24</td>
<td>$</td>
<td>68</td>
<td>44</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>[ENQUIRY]</td>
<td>37</td>
<td>25</td>
<td>%</td>
<td>69</td>
<td>45</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>[ACKNOWLEDGE]</td>
<td>38</td>
<td>26</td>
<td>&amp;</td>
<td>70</td>
<td>46</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>[BELL]</td>
<td>39</td>
<td>27</td>
<td></td>
<td>71</td>
<td>47</td>
<td>G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>[BACKSPACE]</td>
<td>40</td>
<td>28</td>
<td>(</td>
<td>72</td>
<td>48</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>[HORIZONTAL TAB]</td>
<td>41</td>
<td>29</td>
<td>)</td>
<td>73</td>
<td>49</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>[LINE FEED]</td>
<td>42</td>
<td>2A</td>
<td>*</td>
<td>74</td>
<td>4A</td>
<td>J</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>[VERTICAL TAB]</td>
<td>43</td>
<td>2B</td>
<td>+</td>
<td>75</td>
<td>4B</td>
<td>K</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>[FORM FEED]</td>
<td>44</td>
<td>2C</td>
<td>,</td>
<td>76</td>
<td>4C</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>[CARRIAGE RETURN]</td>
<td>45</td>
<td>2D</td>
<td>-</td>
<td>77</td>
<td>4D</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>[SHIFT OUT]</td>
<td>46</td>
<td>2E</td>
<td>.</td>
<td>78</td>
<td>4E</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>[SHIFT IN]</td>
<td>47</td>
<td>2F</td>
<td>/</td>
<td>79</td>
<td>4F</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>[DATA LINK ESCAPE]</td>
<td>48</td>
<td>30</td>
<td>0</td>
<td>80</td>
<td>50</td>
<td>P</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>[DEVICE CONTROL 1]</td>
<td>49</td>
<td>31</td>
<td>1</td>
<td>81</td>
<td>51</td>
<td>Q</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>[DEVICE CONTROL 2]</td>
<td>50</td>
<td>32</td>
<td>2</td>
<td>82</td>
<td>52</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>[DEVICE CONTROL 3]</td>
<td>51</td>
<td>33</td>
<td>3</td>
<td>83</td>
<td>53</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>[DEVICE CONTROL 4]</td>
<td>52</td>
<td>34</td>
<td>4</td>
<td>84</td>
<td>54</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>[NEGATIVE ACKNOWLEDGE]</td>
<td>53</td>
<td>35</td>
<td>5</td>
<td>85</td>
<td>55</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>[SYNCHRONOUS IDLE]</td>
<td>54</td>
<td>36</td>
<td>6</td>
<td>86</td>
<td>56</td>
<td>V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>[ENG OF TRANS. BLOCK]</td>
<td>55</td>
<td>37</td>
<td>7</td>
<td>87</td>
<td>57</td>
<td>W</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>[CANCEL]</td>
<td>56</td>
<td>38</td>
<td>8</td>
<td>88</td>
<td>58</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>[END OF MEDIUM]</td>
<td>57</td>
<td>39</td>
<td>9</td>
<td>89</td>
<td>59</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>[SUBSTITUTE]</td>
<td>58</td>
<td>3A</td>
<td>:</td>
<td>90</td>
<td>5A</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>[ESCAPE]</td>
<td>59</td>
<td>3B</td>
<td>;</td>
<td>91</td>
<td>5B</td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>[FILE SEPARATOR]</td>
<td>60</td>
<td>3C</td>
<td>&lt;</td>
<td>92</td>
<td>5C</td>
<td>\</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>[GROUP SEPARATOR]</td>
<td>61</td>
<td>3D</td>
<td>=</td>
<td>93</td>
<td>5D</td>
<td>}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>[RECORD SEPARATOR]</td>
<td>62</td>
<td>3E</td>
<td>&gt;</td>
<td>94</td>
<td>5E</td>
<td>~</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>[UNIT SEPARATOR]</td>
<td>63</td>
<td>3F</td>
<td>?</td>
<td>95</td>
<td>5F</td>
<td>[DEL]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**“control” characters**

**symbols & numbers**

**uppercase letters**

**lowercase letters**
Comparing Strings

• The values of the ASCII characters are used when comparing strings together
  – Which can lead to some “weird” results

```python
>>> "cat" < "dog"
True
```

```python
>>> "cat" < "Dog"
False
```

```python
>>> "DOG" < "dog"
True
```
More on Comparing Strings

• Gets even more complex when you start adding in numbers and symbols
  >>> "2" < "one"
  True
  >>> "good?" < "good!"
  False
  >>> "UK" < "U.K."
  False
Rules for Comparisons

• To avoid (some) of these issues:

• Always use `lower()` for comparing strings

• Pay attention to symbols
  – *e.g.*, spaces, hyphens, punctuation, etc.
  – Either remove them, or keep them in mind as part of the order
ASCII Characters to ASCII Values

• We can convert between ASCII characters and their values using `ord()` and `chr()`

• The `ord()` function takes in a single character, and returns its ASCII value

• The `chr()` function takes in an integer, and returns its ASCII character
Using `chr()` and `ord()`

```python
>>> chr(65)
'A'
>>> chr(65+32)
'a'
>>> ord('?')
63
>>> ord('d')
100
>>> ord('e')
101
```
Project 3
Project 3 Tips

• Hopefully you have started by now!
  – Work on it a little everyday

• You have been given some solved puzzles
  – Which means you don’t need a working `solve()` to test the other parts of your project
    • Just load in the solution from the file

• Solve the puzzle once, and store the solved puzzle to use it later in your code
  – Don’t resolve it every time you need it

• Make your own puzzles to test!
Project 3 and Deep Copy

• You will need to make a deep copy of the 2D list used to hold your Sudoku board
  – Simply using `new = old[:]` will not work

• We recommend making a function to do this
  – Test that your function works before using it

• Do **NOT** use the built-in `deepcopy()` function, or you will lose major points!!!
Do Not Cheat on Project 3

• Yes, this project has solutions on the internet
  – Yes, we have copies of all of them
  – Yes, we will go looking for new ones after it’s due

• Yes, you could pay someone else to do it
  – Yes, we know of the sites where you can get this done
  – Yes, we will spot “elegant” code that you didn’t write

• Yes, there are libraries to deep copy in python
  – Yes, you will get points off for using them
  – You should not be importing anything for this project
Questions?
Announcements

• Project 3
  – Design is due Tuesday, December 4th
  – Project is due Tuesday, December 11th

• Final exam is when?
  – Friday, December 14th from 6 to 8 PM
  – Locations will be posted on the course website
  – Common final