Characters

Introduction to Computer Science
Iris Howley
TODAY’S LESSON

Unicode

(Characters have numerical representations)
Sorting Characters

```python
>>> word = 'Hello, hi!

>>> sorted(word)
[' ', '!', ',', 'H', 'e', 'h', 'i', 'l', 'l', 'o']
```

Why does 'H' come before 'e'? Why does '   ' come before any letters?
Sorting Characters

- Python uses Unicode code points to sort characters
- Each character is represented by a number
- Python uses this number to sort
<table>
<thead>
<tr>
<th>Dec</th>
<th>Hx Oct</th>
<th>Chr</th>
<th>Dec</th>
<th>Hx Oct</th>
<th>Chr</th>
<th>Dec</th>
<th>Hx Oct</th>
<th>Chr</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>NUL (null)</td>
<td>32</td>
<td>0040</td>
<td>Space</td>
<td>64</td>
<td>100</td>
<td>@#32;</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>SOH (start of heading)</td>
<td>33</td>
<td>0041</td>
<td>!</td>
<td>0</td>
<td>65</td>
<td>@#65;</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>STX (start of text)</td>
<td>34</td>
<td>0042</td>
<td>”</td>
<td>66</td>
<td>102</td>
<td>@#66;</td>
</tr>
<tr>
<td>3</td>
<td>003</td>
<td>ETX (end of text)</td>
<td>35</td>
<td>0043</td>
<td>#</td>
<td>67</td>
<td>103</td>
<td>@#67;</td>
</tr>
<tr>
<td>4</td>
<td>004</td>
<td>EOT (end of transmission)</td>
<td>36</td>
<td>0044</td>
<td>$</td>
<td>68</td>
<td>104</td>
<td>@#68;</td>
</tr>
<tr>
<td>5</td>
<td>005</td>
<td>ENQ (enquiry)</td>
<td>37</td>
<td>0045</td>
<td>%</td>
<td>69</td>
<td>105</td>
<td>@#69;</td>
</tr>
<tr>
<td>6</td>
<td>006</td>
<td>ACK (acknowledge)</td>
<td>38</td>
<td>0046</td>
<td>&amp;</td>
<td>70</td>
<td>106</td>
<td>@#70;</td>
</tr>
<tr>
<td>7</td>
<td>007</td>
<td>BEL (bell)</td>
<td>39</td>
<td>0047</td>
<td>'</td>
<td>71</td>
<td>107</td>
<td>@#71;</td>
</tr>
<tr>
<td>8</td>
<td>010</td>
<td>BS (backspace)</td>
<td>40</td>
<td>0050</td>
<td>(</td>
<td>72</td>
<td>110</td>
<td>@#72;</td>
</tr>
<tr>
<td>9</td>
<td>011</td>
<td>TAB (horizontal tab)</td>
<td>41</td>
<td>0051</td>
<td>)</td>
<td>73</td>
<td>111</td>
<td>@#73;</td>
</tr>
<tr>
<td>10</td>
<td>A01</td>
<td>LF (NL line feed, new line)</td>
<td>42</td>
<td>0052</td>
<td>*</td>
<td>74</td>
<td>112</td>
<td>@#74;</td>
</tr>
<tr>
<td>11</td>
<td>B01</td>
<td>VT (vertical tab)</td>
<td>43</td>
<td>0053</td>
<td>+</td>
<td>75</td>
<td>113</td>
<td>@#75;</td>
</tr>
<tr>
<td>12</td>
<td>C01</td>
<td>FF (NP form feed, new page)</td>
<td>44</td>
<td>0054</td>
<td>,</td>
<td>76</td>
<td>114</td>
<td>@#76;</td>
</tr>
<tr>
<td>13</td>
<td>D01</td>
<td>CR (carriage return)</td>
<td>45</td>
<td>0055</td>
<td>-</td>
<td>77</td>
<td>115</td>
<td>@#77;</td>
</tr>
<tr>
<td>14</td>
<td>E01</td>
<td>SO (shift out)</td>
<td>46</td>
<td>0056</td>
<td>.</td>
<td>78</td>
<td>116</td>
<td>@#78;</td>
</tr>
<tr>
<td>15</td>
<td>F01</td>
<td>SI (shift in)</td>
<td>47</td>
<td>0057</td>
<td>/</td>
<td>79</td>
<td>117</td>
<td>@#79;</td>
</tr>
<tr>
<td>16</td>
<td>020</td>
<td>DLE (data link escape)</td>
<td>48</td>
<td>0060</td>
<td>0</td>
<td>80</td>
<td>120</td>
<td>@#80;</td>
</tr>
<tr>
<td>17</td>
<td>021</td>
<td>DC1 (device control 1)</td>
<td>49</td>
<td>0061</td>
<td>1</td>
<td>81</td>
<td>121</td>
<td>@#81;</td>
</tr>
<tr>
<td>18</td>
<td>022</td>
<td>DC2 (device control 2)</td>
<td>50</td>
<td>0062</td>
<td>2</td>
<td>82</td>
<td>122</td>
<td>@#82;</td>
</tr>
<tr>
<td>19</td>
<td>023</td>
<td>DC3 (device control 3)</td>
<td>51</td>
<td>0063</td>
<td>3</td>
<td>83</td>
<td>123</td>
<td>@#83;</td>
</tr>
<tr>
<td>20</td>
<td>024</td>
<td>DC4 (device control 4)</td>
<td>52</td>
<td>0064</td>
<td>4</td>
<td>84</td>
<td>124</td>
<td>@#84;</td>
</tr>
<tr>
<td>21</td>
<td>025</td>
<td>NAK (negative acknowledge)</td>
<td>53</td>
<td>0065</td>
<td>5</td>
<td>85</td>
<td>125</td>
<td>@#85;</td>
</tr>
<tr>
<td>22</td>
<td>026</td>
<td>SYN (synchronous idle)</td>
<td>54</td>
<td>0066</td>
<td>6</td>
<td>86</td>
<td>126</td>
<td>@#86;</td>
</tr>
<tr>
<td>23</td>
<td>027</td>
<td>ETB (end of trans. block)</td>
<td>55</td>
<td>0067</td>
<td>7</td>
<td>87</td>
<td>127</td>
<td>@#87;</td>
</tr>
<tr>
<td>24</td>
<td>030</td>
<td>CAN (cancel)</td>
<td>56</td>
<td>0070</td>
<td>8</td>
<td>88</td>
<td>130</td>
<td>@#88;</td>
</tr>
<tr>
<td>25</td>
<td>031</td>
<td>EM (end of medium)</td>
<td>57</td>
<td>0071</td>
<td>9</td>
<td>89</td>
<td>131</td>
<td>@#89;</td>
</tr>
<tr>
<td>26</td>
<td>032</td>
<td>SUB (substitute)</td>
<td>58</td>
<td>0072</td>
<td>;</td>
<td>90</td>
<td>132</td>
<td>@#90;</td>
</tr>
<tr>
<td>27</td>
<td>033</td>
<td>ESC (escape)</td>
<td>59</td>
<td>0073</td>
<td>:</td>
<td>91</td>
<td>133</td>
<td>@#91;</td>
</tr>
<tr>
<td>28</td>
<td>034</td>
<td>FS (file separator)</td>
<td>60</td>
<td>0074</td>
<td>&lt;</td>
<td>92</td>
<td>134</td>
<td>@#92;</td>
</tr>
<tr>
<td>29</td>
<td>035</td>
<td>GS (group separator)</td>
<td>61</td>
<td>0075</td>
<td>=</td>
<td>93</td>
<td>135</td>
<td>@#93;</td>
</tr>
<tr>
<td>30</td>
<td>036</td>
<td>RS (record separator)</td>
<td>62</td>
<td>0076</td>
<td>&gt;</td>
<td>94</td>
<td>136</td>
<td>@#94;</td>
</tr>
<tr>
<td>31</td>
<td>037</td>
<td>US (unit separator)</td>
<td>63</td>
<td>0077</td>
<td>?</td>
<td>95</td>
<td>137</td>
<td>@#95;</td>
</tr>
</tbody>
</table>

Source: www.LookupTables.com
Unicode Values

```python
>>> ord(' ')  # ord(' ') converts to unicode
32
>>> ord('H')  # ord('H') converts to unicode
72
>>> ord('d')  # ord('d') converts to unicode
100
>>> ord('e')  # ord('e') converts to unicode
101
>>> ord('l')  # ord('l') converts to unicode
108

>>> chr(32)  # chr(32) converts back to a string
' '
>>> chr(72)  # chr(72) converts back to a string
'H'
>>> chr(100)  # chr(100) converts back to a string
'd'
>>> chr(101)  # chr(101) converts back to a string
'e'
>>> chr(108)  # chr(108) converts back to a string
'l'
```

ord('s') converts to unicode

chr(115) converts back to a string
QUESTIONS?
Please contact me!
Ciphers

Introduction to Computer Science
Iris Howley
TODAY’S LESSON

Ciphers

(Encrypting and decrypting data for security)
End-to-end encryption

Privacy and security is in our DNA, which is why we have end-to-end encryption. When end-to-end encrypted, your messages, photos, videos, voice messages, documents, status updates and calls are secured from falling into the wrong hands.
What is encryption?

"the process of converting messages in ordinary language, or other information into a secret coded form that cannot be interpreted without knowing the secret method for interpretation, called the key."
I send private communication via secret messages, where I shift each letter of the alphabet by a certain number of characters.
Simple Encryption: Caesar Ciphers

With a shift of 1, 'a' would become 'b', 'b' would be 'c', all the way to 'z' which would wrap around and become 'a'.
Simple Encryption: Caesar Ciphers

With a shift of 2, 'a' would become 'c', 'b' would be 'd', all the way to 'y' which would wrap around and become 'a'.
Can YOU encrypt my messages?!
How do we rotate a character by 1?

- $a \rightarrow b$, $b \rightarrow c$, ..., $y \rightarrow z$, $z \rightarrow a$, etc

Any ideas?

```python
>>> rotated = ord('a') + 1
>>> chr(rotated)
'b'
```
Rotating Letters

• `chr(ord('b') + 1)`
  ▪ `'c'`
• `chr(ord('z') + 1)`
  ▪ `'{'`

What happened here?

How do we wrap around back to ‘a’ after ‘z’?

v w x y z {

118 119 120 121 122 123
Rotating Letters

• What we want is for 123 to wrap around back to 97...
• Let’s solve a simpler problem:

Any number over 25 should wrap back around to 0, 1, 2, etc.
• What might we use to do that?
### Rotating Letters

<table>
<thead>
<tr>
<th>a</th>
<th>...</th>
<th>v</th>
<th>w</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>{</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>...</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
</tbody>
</table>

- 24 ______ = 24
- 25 ______ = 25
- 26 ______ = 0
- 27 ______ = 1
- 28 ______ = 2

- 24 % 26 = 24
- 25 % 26 = 25
- 26 % 26 = 0
- 27 % 26 = 1
- 28 % 26 = 2
Rotating Letters

\[
\begin{array}{cccccccc}
0 & \ldots & 21 & 22 & 23 & 24 & 25 & 26 \\
\end{array}
\]

- Our formula is: \( \text{rotatedValue} = \text{value} \mod 26 \)
- But that’s for 0+, how do we convert to 97+?

\[
\begin{array}{cccccccc}
97 & \ldots & 118 & 119 & 120 & 121 & 122 & 123 \\
\end{array}
\]

- Subtract 97 to adjust to 'a' starting at 0:
  - \( \text{rotatedValue} = (\text{originalValue} - 97) \mod 26 \)
- And then add 97 back so we can convert to real unicode value
  - \( \text{rotatedValue} = 97 + (\text{originalValue} - 97) \mod 26 \)
Rotating Numbers

- \( \text{rotatedValue} = 97 + (\text{originalValue} - 97) \mod 26 \)
- Generalizes to:
  - \( \text{rotatedVal} = \text{ord}(\text{givChar}) + (\text{ord(\text{givChar})} - \text{ord}(\text{'a'})) \mod 26 \)
- Convert from Unicode to character:
  - \( \text{rotatedChr} = \text{chr}(\text{ord(\text{'a'})} + (\text{ord(\text{givChar})} - \text{ord(\text{'a'})}) \mod 26) \)
- ...Are we missing anything?
Rotating Numbers

- We need to rotate by 'n':
  \[ \text{rotCh} = \text{chr}(\text{ord('a')} + (((\text{ord(givChar)} - \text{ord('a')}) + n) \mod 26)) \]

- ...Are we missing anything?

- What about...

  \[
  \begin{array}{ccccccc}
  A & B & \ldots & Y & Z & [ & \backslash \\
  65 & 66 & \ldots & 89 & 90 & 91 & 92 \\
  \end{array}
  \]

- \[ \text{start} = \text{ord('A')} \]
- \[ \text{rotCh} = \text{chr(}\text{start} + (((\text{ord(givChar)} - \text{start}) + n) \mod 26)) \]
def encrypt(astr, shift):
    """ Encrypt code, similar to p.81 of textbook """
    result = ''
    for letter in astr:
        if letter.isalpha():
            if letter.isupper():
                start = ord('A')
            elif letter.islower():
                start = ord('a')
            rotateVal = (ord(letter) - start + shift) % 26 + start
            result += chr(rotateVal)
        else:
            result += letter
    return result

if __name__ == '__main__':
    #print("shift by 1", encrypt('ab','c', 1))
    #print("shift by 2", encrypt('ab','c', 2))
    orig = "Hello, my name is Iris!"
    enc = encrypt(orig, 15)
    dec = encrypt(enc, -15)
    print("encrypted: ", enc)
    print("decrypted: ", dec)
QUESTIONS?
Please contact me!
Lab 8

Introduction to Computer Science
Iris Howley
CONNECT TO CAMPUS VIA VPN BEFORE git clone
FROM GITLAB AND BEFORE PUSHING TO GITLAB

TODAY’S LESSON
Building our own ciphers

(An overview of Lab 8)
WHAT IS A CIPHER?
See Lecture 21: Ciphers

WHAT IS ENCRYPTION?
See Lecture 21: Ciphers

HOW TO ROTATE LETTERS?
See Lecture 21: Ciphers
Class Diagram: Lab08

Message represents an object with a text attribute

Message

(and some other properties & methods), See Lab document + starter code)

What's an attribute?
See Lecture 16 & 17

What's a method?
See Lecture 16 & 17

What's a property?
See Lecture 16 & 17
Message is Plaintext's super-class

Plaintext represents a plain text message that can be encrypted.

We can call Message's initializer from Plaintext using:

```
super().__init__(text)
```

What's super/sub-classes?

See Lecture 19 & 20
Ciphertext is a sub-class of Message. Message is Ciphertext's super-class. Ciphertext represents an encrypted text message that can be decrypted. We can call Message's initializer from Ciphertext similar to how we do so with Plaintext. See Lecture 19 & 20.
Lab 08: Helper Methods

• These helper functions are useful and fully implemented:
  ▪ canonical
  ▪ loadWords
  ▪ isWord
  ▪ loadStory
  ▪ Use them! Don't make redundant code!

• rotateLetter is a helper function that you need to implement
  ▪ See Lecture 21!
  ▪ Look at page 81 of the textbook!
QUESTIONS?

Please contact me!
Lab 8: Starter Code

Introduction to Computer Science
Iris Howley
Lab 8: PDF Document

Introduction to Computer Science
Iris Howley
Leftover Slides
TODAY’S LESSON

super()  

(Calling super methods implicitly)
class Robot:
    __slots__ = ['name']
def __init__(self, nm):
    self.name = nm

class EvilRobot(Robot):
    __slots__ = ['mission']
def __init__(self, nm, misn):
    Robot.__init__(self, nm)
    self.mission = misn
Our Simple Robot Class

What happens if our class structure changes?

```python
class Robot:
    __slots__ = ['name']
def __init__(self, nm):
    self.name = nm

class EvilRobot(Robot):
    __slots__ = ['mission']
def __init__(self, nm, misn):
    Robot.__init__(self, nm)
    self.mission = misn
```
Our Simple Robot Class

What happens if our class structure changes?

```python
class Robot:
    __slots__ = ['name']
    def __init__(self, nm):
        self.name = nm

class EvilRobot(PurposeRobot):
    __slots__ = ['mission']
    def __init__(self, nm, misn):
        Robot.__init__(self, nm)
        self.mission = misn
```
class Robot:
    __slots__ = ['name']
def __init__(self, nm):
    self.name = nm

class EvilRobot(Robot):
    __slots__ = ['mission']
def __init__(self, nm, misn):
    super().__init__(nm)  
    self.mission = misn

Calling Super Methods with super()

super().__init__(nm) is similar to:
Robot.__init__(self, nm)

super() lets us call the super-class implicitly...

...and we no longer need to pass self
Ciphers Example

Introduction to Computer Science
Iris Howley
TODAY’S LESSON
Ciphers Example

(Using our cipher.py)
if __name__ == '__main__':
    with open('crypt.py', 'r') as fin:
        codeFile = fin.read()
    with open('encrypted', 'w') as fout:
        fout.write(encrypt(codeFile, 13))
    with open('encrypted', 'r') as fin:
        print(encrypt(fin.read(), -13))
**Stdin**

< for stdin > for pushing output to a file

- **python3 crypt.py < crypt.py > whatever**
  - Encrypts stdin into whatever file
  - Rotates by 13 (by default, see program)

- **python3 crypt.py < whatever > second.py**
  - Rotates 'whatever' by 13, stores in second.py
  - What happens when you rotate by 13 twice?

- **python3 crypt.py < crypt.py | python3 crypt.py**
  - Rotates by 13, then rotates by 13 again
QUESTIONS?

Please contact me!