CMPT 165
INTRODUCTION TO THE
INTERNET AND THE
WORLD WIDE WEB

Unit 7
Introduction to Programming
Learning Objectives

In this unit you will learn the following.

• **Explain** what programming is and what Python is.
• **Learn** Python basics.
• **Create** simple Python programs.
Topics

1. Computer Programming  
2. Python for Programming  
3. Python Basics…  
4. …more Python Basics  
5. …and even more Python Basics  
6. Demos and Examples  
7. Code Tracing

Lectures 1 & 2
Lectures 2, 3, & 4
Lectures 5 & 6
Computer Science

def. is the scientific and practical approach to computation and its applications (a.k.a algorithms).

- **Theory** - *is it possible to compute…*
- **Analysis** - *how well did that compute…*
- **Design** - *how can we compute…*
- **Optimization** - *how fast can we compute…*
- **Implementation** - *let’s create a program to compute…*
- **Application** - *let’s run program to compute this data…*

**Computer** def. a general purpose programmable information processor with input and output.

von Neumann Machine (1943)

Copyright © 2014 by Stephen Makonin
Computer Program

• *def* a list of instructions a computer follows to perform a given task.
• **Programming** is the act of creating that list of instructions.
• You need to tell the computer what you want it to do.
• Sometimes this requires specifying a lot of details.
• Instructions have to be specified in a way the computer can understand (a.k.a. machine code)

```
A4 00 F5 34 22 BA BB 75
00 9F 67 6E EE 80 EF 21
```
• We **do not** understand machine code.
• Computers **do not** understand natural language.
• **Compromise:** so we create something in between
  • a **programming language:**
Programming Approaches

Structured Programming
• Came about the 1960s
• Programming broken down into functions and modules
• Clear structure, easy to test, debug, and modify
• Design more complex program

Object Oriented Programming
• **Object** def.: entity characterized by state and behaviours.
• State is the data and **behaviours** are methods/functions.

Can do both, we will only do structured.
Markup vs. Programming

- We have used HTML to markup content.
- Use **markup** to **create structure** for content.
- Use **programming** to **perform tasks**.
  - Take and input and produce an output.
  - Process and manipulate input data.
  - Create a list of instructions to process data.
- Programs use markup to **understand** content for processing.
- HTML markup is **not** programming.
- Rendering HTML in its visual form is **programming**.
Why learn to program?

• So you probably never make a program before…
• And you might ask yourself why start now? 😏
• Some might reply because it is fun! 😃
• Your instructor would reply because you have to!!! 😣
• A real reason: because it helps solve really hard problems, fast. Problems that would take us a long time to solve.
• An example: what are the first 10,000 prime numbers?
  • A computer can do this easily in under 2 seconds.
  • A human? …days? …weeks?
Why Python?

...why not C++, C#, or Java?

• Python is simple to learn and use.
  • You can solve problems quickly with little overhead.
  • No need to worry about: memory allocation, numerical limits, the structure of data types, etc.

• Python is a good first programming language to learn.

• Python is well suited to web programming.

• The focus is not to making you into a programmer, rather for you to understand the general concept of programming.
Python 2.7.x

Now would be a good time to install Python!
Computers ❤️ Binary

- **Decimal** is base-10, symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- **Binary** is base-2, symbols 0, 1 (or OFF, ON)
  - e.g. 1010001010101111000 (can be very long)
- **Hex** is base-16 ($2^4$) (shorter then binary!),
  - symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Computers love whole numbers.
- Rational numbers need special computation
  - e.g. 4.5 or $\frac{1}{2}$
- What about characters — our alphabet?
## Characters are Numbers

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Char</th>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Html</th>
<th>Chr</th>
<th>Dec</th>
<th>Hx</th>
<th>Oct</th>
<th>Html</th>
<th>Chr</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>0</td>
<td>NUL (null)</td>
<td>32</td>
<td>20</td>
<td>040</td>
<td>¦#32;</td>
<td>Space</td>
<td>64</td>
<td>40</td>
<td>100</td>
<td>¦#64;</td>
<td>Ø</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>1</td>
<td>SOH (start of heading)</td>
<td>33</td>
<td>21</td>
<td>041</td>
<td>¦#33;</td>
<td></td>
<td>65</td>
<td>41</td>
<td>101</td>
<td>¦#65;</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>2</td>
<td>STX (start of text)</td>
<td>34</td>
<td>22</td>
<td>042</td>
<td>¦#34;</td>
<td></td>
<td>66</td>
<td>42</td>
<td>102</td>
<td>¦#66;</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>003</td>
<td>3</td>
<td>ETX (end of text)</td>
<td>35</td>
<td>23</td>
<td>043</td>
<td>¦#35;</td>
<td></td>
<td>67</td>
<td>43</td>
<td>103</td>
<td>¦#67;</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>004</td>
<td>4</td>
<td>EOT (end of transmission)</td>
<td>36</td>
<td>24</td>
<td>044</td>
<td>¦#36;</td>
<td></td>
<td>68</td>
<td>44</td>
<td>104</td>
<td>¦#68;</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>005</td>
<td>5</td>
<td>ENQ (enquiry)</td>
<td>37</td>
<td>25</td>
<td>045</td>
<td>¦#37;</td>
<td></td>
<td>69</td>
<td>45</td>
<td>105</td>
<td>¦#69;</td>
<td>E</td>
</tr>
<tr>
<td>6</td>
<td>006</td>
<td>6</td>
<td>ACK (acknowledge)</td>
<td>38</td>
<td>26</td>
<td>046</td>
<td>¦#38;</td>
<td></td>
<td>70</td>
<td>46</td>
<td>106</td>
<td>¦#70;</td>
<td>F</td>
</tr>
<tr>
<td>7</td>
<td>007</td>
<td>7</td>
<td>BEL (bell)</td>
<td>39</td>
<td>27</td>
<td>047</td>
<td>¦#39;</td>
<td></td>
<td>71</td>
<td>47</td>
<td>107</td>
<td>¦#71;</td>
<td>G</td>
</tr>
<tr>
<td>8</td>
<td>010</td>
<td>8</td>
<td>BS (backspace)</td>
<td>40</td>
<td>28</td>
<td>050</td>
<td>¦#40;</td>
<td></td>
<td>72</td>
<td>48</td>
<td>110</td>
<td>¦#72;</td>
<td>H</td>
</tr>
<tr>
<td>9</td>
<td>011</td>
<td>9</td>
<td>TAB (horizontal tab)</td>
<td>41</td>
<td>29</td>
<td>051</td>
<td>¦#41;</td>
<td></td>
<td>73</td>
<td>49</td>
<td>111</td>
<td>¦#73;</td>
<td>I</td>
</tr>
<tr>
<td>10</td>
<td>A012LF</td>
<td>(NL line feed, new line)</td>
<td>42</td>
<td>2A</td>
<td>052</td>
<td>¦#42;</td>
<td></td>
<td>74</td>
<td>4A</td>
<td>112</td>
<td>¦#74;</td>
<td>J</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B013VT</td>
<td>(vertical tab)</td>
<td>43</td>
<td>2B</td>
<td>053</td>
<td>¦#43;</td>
<td></td>
<td>75</td>
<td>4B</td>
<td>113</td>
<td>¦#75;</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>C014FF</td>
<td>(NP form feed, new page)</td>
<td>44</td>
<td>2C</td>
<td>054</td>
<td>¦#44;</td>
<td></td>
<td>76</td>
<td>4C</td>
<td>114</td>
<td>¦#76;</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>D015CR</td>
<td>(carriage return)</td>
<td>45</td>
<td>2D</td>
<td>055</td>
<td>¦#45;</td>
<td></td>
<td>77</td>
<td>4D</td>
<td>115</td>
<td>¦#77;</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>E016SO</td>
<td>(shift out)</td>
<td>46</td>
<td>2E</td>
<td>056</td>
<td>¦#46;</td>
<td></td>
<td>78</td>
<td>4E</td>
<td>116</td>
<td>¦#78;</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F017SI</td>
<td>(shift in)</td>
<td>47</td>
<td>2F</td>
<td>057</td>
<td>¦#47;</td>
<td></td>
<td>79</td>
<td>4F</td>
<td>117</td>
<td>¦#79;</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>020DLE</td>
<td>(data link escape)</td>
<td>48</td>
<td>30</td>
<td>060</td>
<td>¦#48;</td>
<td></td>
<td>80</td>
<td>50</td>
<td>120</td>
<td>¦#80;</td>
<td>P</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>021DC1</td>
<td>(device control 1)</td>
<td>49</td>
<td>31</td>
<td>061</td>
<td>¦#49;</td>
<td></td>
<td>81</td>
<td>51</td>
<td>121</td>
<td>¦#81;</td>
<td>Q</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>022DC2</td>
<td>(device control 2)</td>
<td>50</td>
<td>32</td>
<td>062</td>
<td>¦#50;</td>
<td></td>
<td>82</td>
<td>52</td>
<td>122</td>
<td>¦#82;</td>
<td>R</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>023DC3</td>
<td>(device control 3)</td>
<td>51</td>
<td>33</td>
<td>063</td>
<td>¦#51;</td>
<td></td>
<td>83</td>
<td>53</td>
<td>123</td>
<td>¦#83;</td>
<td>S</td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>024DC4</td>
<td>(device control 4)</td>
<td>52</td>
<td>34</td>
<td>064</td>
<td>¦#52;</td>
<td></td>
<td>84</td>
<td>54</td>
<td>124</td>
<td>¦#84;</td>
<td>T</td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>025MAK</td>
<td>(negative acknowledge)</td>
<td>53</td>
<td>35</td>
<td>065</td>
<td>¦#53;</td>
<td></td>
<td>85</td>
<td>55</td>
<td>125</td>
<td>¦#85;</td>
<td>U</td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>026SYN</td>
<td>(synchronous idle)</td>
<td>54</td>
<td>36</td>
<td>066</td>
<td>¦#54;</td>
<td></td>
<td>86</td>
<td>56</td>
<td>126</td>
<td>¦#86;</td>
<td>V</td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>027ETB</td>
<td>(end of trans. block)</td>
<td>55</td>
<td>37</td>
<td>067</td>
<td>¦#55;</td>
<td></td>
<td>87</td>
<td>57</td>
<td>127</td>
<td>¦#87;</td>
<td>W</td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>030CAN</td>
<td>(cancel)</td>
<td>56</td>
<td>38</td>
<td>070</td>
<td>¦#56;</td>
<td></td>
<td>88</td>
<td>58</td>
<td>130</td>
<td>¦#88;</td>
<td>X</td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>031EM</td>
<td>(end of medium)</td>
<td>57</td>
<td>39</td>
<td>071</td>
<td>¦#57;</td>
<td></td>
<td>89</td>
<td>59</td>
<td>131</td>
<td>¦#89;</td>
<td>Y</td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>032SUB</td>
<td>(substitute)</td>
<td>58</td>
<td>3A</td>
<td>072</td>
<td>¦#58;</td>
<td></td>
<td>90</td>
<td>5A</td>
<td>132</td>
<td>¦#90;</td>
<td>Z</td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>033ESC</td>
<td>(escape)</td>
<td>59</td>
<td>3B</td>
<td>073</td>
<td>¦#59;</td>
<td></td>
<td>91</td>
<td>5B</td>
<td>133</td>
<td>¦#91;</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>034FS</td>
<td>(file separator)</td>
<td>60</td>
<td>3C</td>
<td>074</td>
<td>¦#60;</td>
<td></td>
<td>92</td>
<td>5C</td>
<td>134</td>
<td>¦#92;</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>035GS</td>
<td>(group separator)</td>
<td>61</td>
<td>3D</td>
<td>075</td>
<td>¦#61;</td>
<td></td>
<td>93</td>
<td>5D</td>
<td>135</td>
<td>¦#93;</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>036RS</td>
<td>(record separator)</td>
<td>62</td>
<td>3E</td>
<td>076</td>
<td>¦#62;</td>
<td></td>
<td>94</td>
<td>5E</td>
<td>136</td>
<td>¦#94;</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>037US</td>
<td>(unit separator)</td>
<td>63</td>
<td>3F</td>
<td>077</td>
<td>¦#63;</td>
<td></td>
<td>95</td>
<td>5F</td>
<td>137</td>
<td>¦#95;</td>
<td></td>
</tr>
</tbody>
</table>

Source: www.LookupTables.com

Copyright © 2014 by Stephen Makonin
The letter **A** (uppercase) has the integer value **65** (hex **0x41**) where as lowercase **a** has the integer value **97** (hex **0x61**).
The letter A (uppercase) has the integer value 65 (hex 0x41)
where as lowercase a has the integer value 97 (hex 0x61).

※ Is this correct? 'A' + 0x20 = 'a'
The letter A (uppercase) has the integer value 65 (hex 0x41) where as lowercase a has the integer value 97 (hex 0x61).

Is this correct? 'A' + 32 = 'a'

We need 2 Python functions: \texttt{ord()} and \texttt{chr()}

\texttt{ord()} turns a character into an integer.

\texttt{chr()} turns an integer into a character.
Is this correct? 'A' + 32 = 'a'

We need 2 Python functions: `ord()` and `chr()`

- `ord()` turns a character (chr) into an integer/ordinal.
- `chr()` turns an integer/ordinal into a character (chr).

In Python 2.7.x IDLE execute:

```python
chr(ord('A') + 32)
```

Is the question on the first line correct?
QUESTIONS?
Program Execution

def. performing the instructions of a computer program.

For simplicity, we can assume that the computer execute each line of code in a program from top to bottom, from outwards to inwards to outwards.

```python
i = 10
total = 0
text = 'the total is:'

for j in range(i):
    total = total + j

print text, j

print 'the end'
```

code output:
```
the total is: 0
the total is: 1
the total is: 2
the total is: 3
the total is: 4
the total is: 5
the total is: 6
the total is: 7
the total is: 8
the total is: 9
the end
```
**Literals & Variables**

**Literal**s are fixed values.

For example: boolean, integer, float, character.

**Variables** (for simplicity) are a labelled place in computer memory that store literals.
Assignment & Evaluation

For simplicity, each line of the program is executed sequentially.

Evaluation performs the instructions on the right side of the Assignment Operator which results in a literal.

Assignment takes the results from evaluation and stores them in a variable on the left side of the Assignment Operator.
GO TO THE PYTHON BASICS SLIDES
• data that is organized in key-value pairs:

```python
dict = {key: value, ...}
```

```
>>> person = {'name': 'John Doe', 'age': 32}
>>> person['name']
'John Doe'
>>> person['age']
32
>>> 'age' in person
True
>>> 'gender' in person
False
```
Dictionary Lists

• And we can have lists of dictionaries:

```python
>>> person = [{'name': 'John Doe', 'age': 32},
             {'name': 'Jane Doe', 'age': 76}]

>>> person[0]['name']
'John Doe'

>>> person[1]['name']
'Jane Doe'

>>> for p in person:
   print "%s is %d years old!" % (p['name'], p['age'])

John Doe is 32 years old!
Jane Doe is 76 years old!
```
QUESTIONS?
DEMOS & EXAMPLES
Coding: Max

Define a function `max()` that takes two numbers as arguments and returns the largest of them. Use the if-then-else construct available in Python.

It is true that Python has the `max()` function built in, but writing it yourself is nevertheless a good exercise.
Coding: Max of 3

Define a function `max_of_three()` that takes three numbers as arguments and returns the largest of them.

Source: [http://www.ling.gu.se/~lager/python_exercises.html](http://www.ling.gu.se/~lager/python_exercises.html)
Coding: Max of $n$

Define a function `max_of_n()` that takes a list of numbers as an argument and returns the largest of them.
Coding: Reverse

Define a function `reverse()` that computes the reversal of a string.

For example, `reverse("I am testing")` should return the string "gnitset ma I".
Coding: Repeat

Define a function `repeat_chars()` that takes an integer `n` and a character `c` and returns a string, `n` characters long.

For example, `repeat_chars(5, "x")` should return the string "xxxxx".

Python is unusual in that you can actually write an expression `5 * "x"` that will evaluate to "xxxxx". For the sake of the exercise you should ignore that the problem can be solved in this manner.
Coding: Histogram

Define a procedure `histogram()` that takes a list of integers and prints a histogram to the screen. For example, `histogram([4, 9, 7])` should print the following:

```
****
*********
*******
```

Source: http://www.ling.gu.se/~lager/python_exercises.html
Copyright © 2014 by Stephen Makonin
"99 Bottles of Beer" is a traditional song in the US and Canada. It is popular to sing on long trips, as it has a very repetitive format which is easy to memorize, and can take a long time to sing. The song's simple lyrics are as follows:

99 bottles of beer on the wall, 99 bottles of beer.
Take one down, pass it around, 98 bottles of beer on the wall.

The same verse is repeated, each time with one fewer bottle. The song is completed when the singer or singers reach zero.

Your task here is write a Python program capable of generating all the verses of the song.

Source: http://www.ling.gu.se/~lager/python_exercises.html
Copyright © 2014 by Stephen Makonin
How can determine how a program can run using pencil and paper, no computer?

```python
#EXAMPLE 0 - Trace the Exection.
for i in range(5):
    print i
```

<table>
<thead>
<tr>
<th>Line Number</th>
<th>range(5)</th>
<th>i</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-1</td>
<td>[0, 1, 2, 3, 4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-1</td>
<td>[0, 1, 2, 3, 4]</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4-1</td>
<td>[0, 1, 2, 3, 4]</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3-2</td>
<td>[0, 1, 2, 3, 4]</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4-2</td>
<td>[0, 1, 2, 3, 4]</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3-3</td>
<td>[0, 1, 2, 3, 4]</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4-3</td>
<td>[0, 1, 2, 3, 4]</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3-4</td>
<td>[0, 1, 2, 3, 4]</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4-4</td>
<td>[0, 1, 2, 3, 4]</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3-5</td>
<td>[0, 1, 2, 3, 4]</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4-5</td>
<td>[0, 1, 2, 3, 4]</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3-6</td>
<td>[]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Summary

• Learnt about computer programming.
• Learnt about the Python programming language.
• Performed a coding exercise.
• Experienced what it is like to program a specific project.

Next Unit: look at HTML forms and CGI.
QUESTIONS?