**Total Classes conducted:** 14

**Topics not covered at all**

1. Pickle
2. Custom exception class
3. Special Methods
4. Type identification
5. Package basics
6. Special Methods
7. Debugging
8. re.sub()
9. re.findall()
10. XLRD package
11. Socket Programming
12. Basic debugging
13. Subclass hook

**Topics started but not completed:**

1. Classes: Discussed only in one class with basics
2. Decorators
3. Regular expressions

**Open questions:**

1. Data Encapsulation:

http://www.pythoncourse.eu/object\_oriented\_programming.php

2. Unicode in socket programing ------------------ ftplib, telnetlib

f = io.open("abc.txt", "wt", encoding="utf-8")

f.write(u"Imagine non-English language here")

f.close()

3. Open question on below program why one instance variable is applied to another instance also

http://www.swaroopch.com/notes/python/#oop\_Need explanation of

4. finding multiple search items in given string and how to print them

| **Method/Attribute** | **Purpose** |
| --- | --- |
| span() | Return a tuple containing the (start, end) positions of the match |

| **Method/Attribute** | **Purpose** |
| --- | --- |
| match() | Determine if the RE matches at the beginning of the string. |
| search() | Scan through a string, looking for any location where this RE matches. |
| findall() | Find all substrings where the RE matches, and returns them as a list. |
| finditer() | Find all substrings where the RE matches, and returns them as an[*iterator*](https://docs.python.org/2/glossary.html#term-iterator). |

| **Flag** | **Meaning** |
| --- | --- |
| **DOTALL**, **S** | Make . match any character, including newlines (Covered) |
| **IGNORECASE**, **I** | Do case-insensitive matches (Covered) |
| **LOCALE**, **L** | Do a locale-aware match |
| **MULTILINE**, **M** | Multi-line matching, affecting ^ and $ |
| **VERBOSE**, **X** | Enable verbose REs, which can be organized more cleanly and understandably. |
| **UNICODE**, **U** | Makes several escapes like \w, \b, \s and \d dependent on the Unicode character database. |

| **Method/Attribute** | **Purpose** |
| --- | --- |
| split() | Split the string into a list, splitting it wherever the RE matches |
| sub() | Find all substrings where the RE matches, and replace them with a different string |
| subn() | Does the same thing as **sub()**, but returns the new string and the number of replacements |

**Below open question on regular expression (topic covered last day):**

In [23]: re.compile(str1)

Out[23]: re.compile(r"[1, 2, 3, 'udhay', 'prak']")

In [24]: reg=re.compile(str1)

In [25]: print reg.match('prakash')

<\_sre.SRE\_Match object at 0x02FAC870>

abababababab

In [107]: string="aaaaaa"

In [113]: re.match('a{3}\*',string).group()

In [123]: re.match("h[e,E]llo",string).group()

In [151]: str1="1234 12 34"

In [152]: re.match("[0-5][3-6].\*",str1).group()

In [186]: re.match('X', 'A\nB\nX', re.MULTILINE) # No match

In [187]: print 'A\nB\nX'

A

B

X

In [188]: re.search('^X', 'A\nB\nX', re.MULTILINE)

Out[188]: <\_sre.SRE\_Match at 0x300a368>

In [189]: re.match('X', 'A\nB\nX') # No match

In [190]: re.match('X', 'A\nB\nX',re.DoTALL) # No match

---------------------------------------------------------------------------

AttributeError Traceback (most recent call last)

<ipython-input-190-b868c67a3a40> in <module>()

----> 1 re.match('X', 'A\nB\nX',re.DoTALL) # No match

AttributeError: 'module' object has no attribute 'DoTALL'

In [191]: re.match('X', 'A\nB\nX',re.DOTALL) # No match

In [192]: re.search('^X', 'A\nB\nX', re.MULTILINE)

Out[192]: <\_sre.SRE\_Match at 0x300a480>

In [193]: re.search('^X', 'A\nB\nX')

In [194]: re.search('^X', 'A\nB\nX're.DOTALL)

File "<ipython-input-194-efe3a54b8206>", line 1

re.search('^X', 'A\nB\nX're.DOTALL)

^

SyntaxError: invalid syntax

In [195]: re.search('^X', 'A\nB\nX',re.DOTALL)

In [196]: re.search('X', 'A\nB\nX',re.DOTALL)

Out[196]: <\_sre.SRE\_Match at 0x300a640>

In [197]: re.search('X', 'A\nB\nX')

Out[197]: <\_sre.SRE\_Match at 0x300a870>

In [268]: re.search('ki',str1).groups()