|  |  |
| --- | --- |
|  | Pythonic thing |
| 0 | **print**  **python 2.7**  print <variable>  **python 3+**  print (<variable>) |
| 1 | **stdin input:**  python 2.7:  raw\_input()  use s = raw\_input()  **python 3+**  module “fleinput”  use: import fileinput  <https://docs.python.org/3/library/fileinput.html> |
| 2 | **String manipulation:**  remove last char from a string called a  a = a[:-1]  What does a = a[:-5] do?  what if a = “1” to start with?  What does a[::-1] do? |
| 3 | **Check if a string s is an integer:**  if the string is indeed an integer, then   * int(s)   would return the integer that s represents, else would throw an error.   * s.isdigit() would return boolean value only if s is a positive integer |
| 4. | **Prevent code from stopping to run when error is thrown**  try:  <code that may throw error>  except:  <what do you want to do if the code fails> |
| 5. | **some operators**   * ==   a == b compares two variables a and b by value, return boolean   * != (not equal to)   compares if a and b are not equal to each other by value, returns boolean   * % (modulo)   a % b  return the reminder when b divides a.  a is divisible by b if and only if a % b == 0 |
| 6. | **if - else**  if <logical condition>:  <code block>  (else is optional)  else:  <code block> |
| 7. | **Logical operators**  <boolean> and <another boolean>  <boolean> or <another boolean> |
| 8. | **Looping through**   * **a range of number**   for i in range(<start number> , <stop right before this end number>):  <code block>  Loops from gegin to end with 1 increment   * multiple ranges of numbers   for i in range(<start number> , <stop right before this end number for i>):  for j in range(<start number> , <stop right before this end number for j>):  <code block>  for i in range(<start number> , <stop right before this number, -1):  <code block>  this accesses the range in reverse.  Eg:  print(list(range(2,-1,-1)))  >>[2,1,0] |
| 9. | **convert an integer to a string**  if a is an integer, then   * s = str(a)   would give string s that represents a as a string |
| 10. | **dictionary**  Python has dictionaries storing key value pairs.  A dictionary has a unique key.  A dictionary is used to get the value of a key very fast.  Eg:  d = {1:”one”,2:”two”,3:”three”}  print(d[1])  >>”one” |

**Leap year logic:**

integer a is a leap year if the following happens:

* 4 divides a ( meaning the remainder is 0) and 100 does not divide a ( meaning the remainder is not 0)
* 4 divides a and 100 divides a and 400 divides a

**palindrome logic:**

an integer a is a palindrome if it reads the same back and reverse. Eg:12321

Hence, a number a is a palindrome if

* the string representing a is same the reverse string of a

**Largest palindrome logic:**

* what is the first and last three-digit number? (that is what is the starting and ending point of tour range?)
* How many for loops you need to write? Are they independent or nested?
* Since a\*b == b\*a, do you need to go through all of the range? Or you can shorten it for the second for loop?
* write a separate method to determine if the given number is a palindrome. What is the most appropriate place to compare if a number is a palindrome or not in the for loops?
* Have a separate variable to determine the largest number, after you know it is a palindrome.
* I will give you extra points if you can write a condition that stops the for loops when the largest palindrome is found :)

**Python ide for in-class practice:**

Python is an indented language. So, the indents matter. It is very hard to do this in vi, unless you got a tone of practice for it and you really like vi (I don’t) and impossible to do it with nano in the given lab time. So, for the lab purposes, if you do not have a python environment setup, we will use online ide

<https://repl.it/languages/python>

This ide **DOES NOT SAVE YOUR WORK,** but it can run python scripts. It is extremely easy to type python in here since the ide auto indents the right blocks. However, you need to un-indent something once you are done writing a method to get out of that method.