|  |  |
| --- | --- |
| Precedence  \*\*, \*, /, //, %, +, - , not, and , or  (6.2 + 8) \* (3-10/5) == 14.2 | BIF  Abs, float, format, int, round,  range(10) - 10 times (0 through 9)  range(5,10) - 5 times: 5 through 9 |
| Lists:--- Slicing  thelist = [5,3,‘p’,9,‘e’]  thelist[0] returns 5  thelist[1:3] returns [3, ‘p’]  thelist[2:] returns [‘p’, 9, ‘e’]  thelist[:2] returns [5, 3]  thelist[2:-1] returns [‘p’, 9]  thelist.pop() -- 'e'  ----------  thelist + [0] returns [‘z’,9,’p’,0]  9 in thelist returns True  **for** i in range(1,len(s)+1):  rStr += s[-1 \* i]  print (s[-1 \* i], i)  ------------  banned\_users = ['ann', 'chad', 'dee']  user = 'erin'  if user not in banned\_users:  print("You can play!")  **-------------WHILE**  msg = ''  while msg != 'quit':  msg = input("What's your message? ")  print(msg)  -----  while True:  n = input("Enter 'hello':")  if n == 'hello':  break | --- Dictionary  table = {'Sjoerd': 4127, 'Jack': 4098}  for name, phone in table.items():  print('{0:10} ==> {1:10d}'.format(name, phone))  ---  d.items() - view of the (key,value)  d.keys() -Returns a view of the keys  d.pop(key) -Removes the (key, value) pair  d.update(d2) -Adds the (key, value) pairs  d.values()-view of the values of d  --  def startup(fileName):  accountDict = {}  # open and read the file only once; use readline()  try:  file = open (fileName, 'r')  lines = file.readlines()  file.close()  except:  return (False, accountDict)  # read throuugh and store it in a Dictionary  for line in lines:  acctData = line.split(",")  # Layout: key, [fName, lName, Balance]  accountDict[acctData[0]] = [acctData[1]]  accountDict[acctData[0]].append(acctData[2])  accountDict[acctData[0]].append(float(acctData[3][:-1])) ## convert balance to float  return (True, accountDict) |
| o Random Module  Random numbers  import random  dice1 = random.randrange(1, 6)  Random choices  **random.choice(seq)**  Return a random element from the non-empty sequence seq. | **Files:**  thisfile.read() #reads entire file into one string  thisfile.readline() #reads one line of a file  thisfile.readlines() #reads entire file into a list of strings, one per line  for eachline in thisfile: #steps through lines in a file  Sample  infile = open(fname, 'r')  content = infile.read()  infile.close()  m = ['Monday', 'Tuesday',… ]  outfile = open('NumberOfDays.txt', 'w')  for i in range(len(m)):  s = content.count(m[i])  print(m[i],s)  outfile.write(m[i]+ ' is mentioned '+ str(s)+ ' times in '+ fname + '\n')  return |

|  |  |
| --- | --- |
| Scope and namespaces  Encapsulation in functions  Global and local variables  Passing parameters  Returning data  Program stack  User-defined functions | o Exceptions  Try/except blocks  prompt = "How many tickets do you need? "  num\_tickets = input(prompt)  try:  num\_tickets = int(num\_tickets)  except ValueError:  print("Please try again.") |
| class Point:  'class that represents a point in the plane’  def \_\_init\_\_(self, xcoord, ycoord):  'initialize coordinates to (xcoord, ycoord)'  self.x = xcoord  self.y = ycoord  def setx(self, xcoord):  'set x coordinate of point to xcoord'  self.x = xcoord  def sety(self, ycoord):  'set y coordinate of point to ycoord'  self.y = ycoord  def get(self):  'return coordinates of the point as a tuple'  return (self.x, self.y)  def move(self, dx, dy):  'change the x and y coordinates by dx and dy'  self.x += dx  self.y += dy  **a** = Point()  a.setx(3)  a.getx()  **a- is namespace** | **o Recursion**  def prompt():  passw = input("Enter Password:")  if len(passw) ==0:  return prompt()  savpass = ''  else:  savpass = passw  return savpass  -------------  defsuml(lst):  'sum all elements in sublists of lst'  s = 0  for row in lst:  for elemin row:  s += elem  return s |

class Animal:

def setSpecies(self, species):

self.spec = species

def setLanguage(self, language):

self.lang = language

def speak(self):

print('I am a {} and I {}.'.format(self.spec, self.lang))

>>> snoopy = Animal()

>>> snoopy.setpecies('dog')

>>> snoopy.setLanguage('bark')

>>> snoopy.speak() I am a dog and I bark.